OAK RIDGE National Laboratory

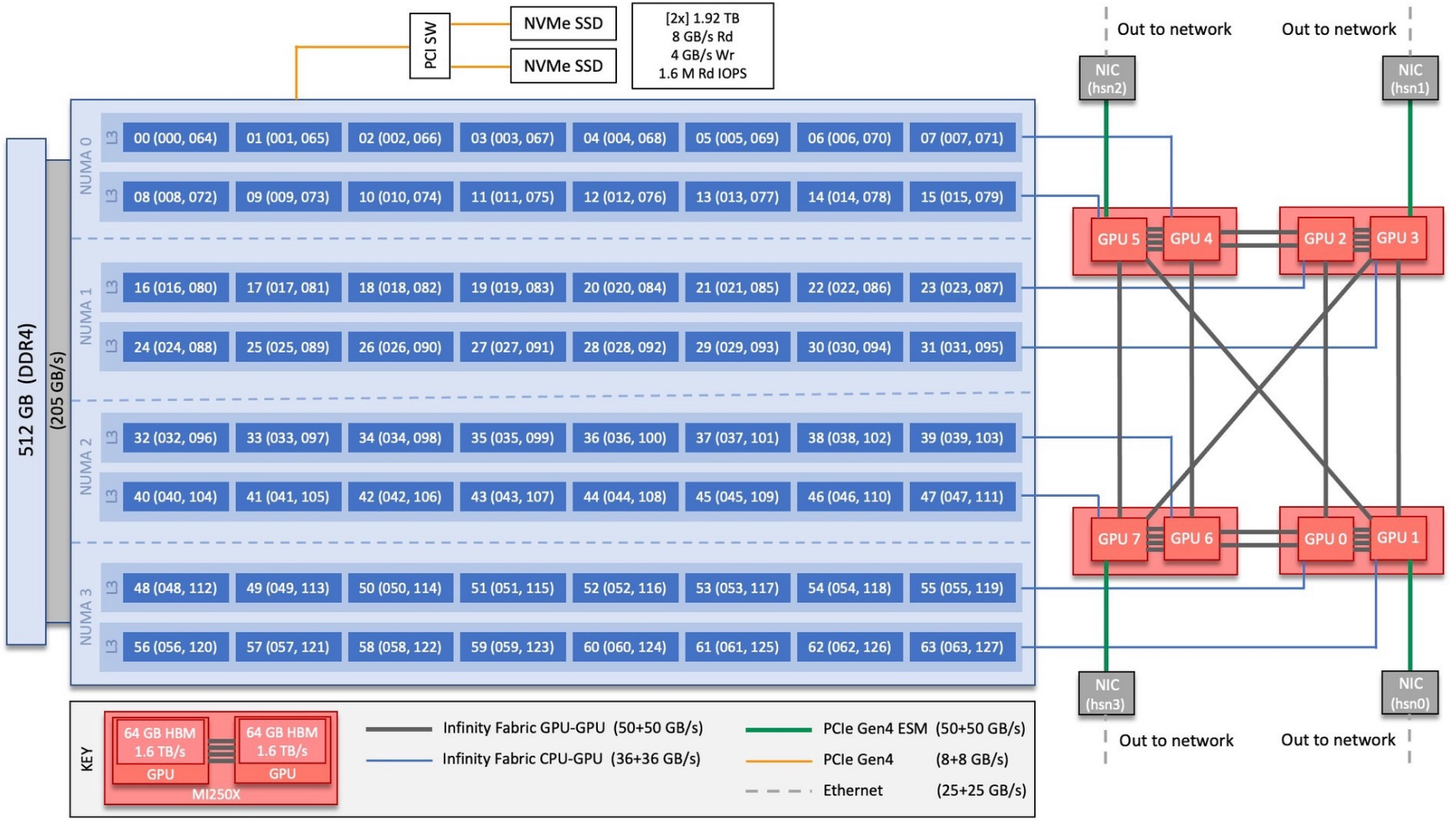
Frontier Node Health Checking and State Management

CUG2023 – Helsinki, Finland

Matt Ezell

ORNL is managed by UT-Battelle LLC for the US Department of Energy

U.S. DEPARTMENT OF ENERGY



Frontier Node

During Frontier Build -- the Chip Shortage Hit in Earnest!

- When HPE began ordering parts, suppliers said the lead time on orders was increasing an additional 6-12 months.

ORNL worked with ASCR to get DPAS rating for Frontier that helped prioritize USA part orders (DPAS was extended to Aurora and El Capitan)

- **60 Million parts** needed for Frontier
  - 685 Different part numbers used in Frontier
  - 167 Frontier part numbers affected by the chip shortage
    - (more than 2 million parts from dozens of suppliers worldwide)
  - 12 Part numbers blocked building the first compute cabinet
  - 15 Part numbers shortage for AMD building all the MI200 cards for Frontier

It wasn't exotic parts like CPUs or GPUs, rather parts needed by everyone – in cars, TVs, electronics, such as, voltage regulators, oscillators, power modules

Slide content courtesy Al Geist, ASCAC 2022

Supply Chain Remained a Constant Battle until Delivery

HPE saw commitments for parts deliveries from sub-contractors being broken weekly as the chip shortage got worse. Had to call every supplier every week (sometimes every day)

HPE had 15 people whose sole job was to try to find the needed parts or alternatives for Frontier. Using HPE's size to negotiate with suppliers, looking for handfuls of parts in warehouses or at other companies who were also stuck because of chip shortage.

April 30 – July 15: Initial shortage of 167 part numbers reduced down to 1 part #

- July 15th only found enough to build 63 of 74 cabinets (short 8,000)
- Took three more weeks to find all 8,000
- By that time had a couple more decommits on another part.

The final parts arrived the morning the last Frontier node was assembled

Slide content courtesy Al Geist, ASCAC 2022

Need for Node Health Checks

> "Mean time between failure on a system this size is hours, it's not days, so you need to make sure you understand what those failures are and that there's no pattern to those failures that you need to be concerned with."
>
> - Justin Whitt, OLCF Project Director

- Hardware and software faults can cause application failures – need to check for both
- Some issues are detectable in-band, others only out-of-band

Evaluation of Node Health Check Scripts

- The only viable open solution we found was NHC from LBL/mej
  - HPCM `cm health check` uses this under the hood
- We have an in-house script called `checknode` that we use on Summit, written in bash
- Prototyped a python-based custom replacement – did not develop a simpler interface

In the end, we decided to evolve `checknode` to work on Frontier

  - No need to specify host targets
  - Functions and tests in a single file

When checknode is run

Run…	By…
At boot	HPCM startup scripts
Between each job	The Slurm epilog
At request	A human attempting to clear a previous drain reason
Every 15 minutes	cron, on nodes previously drained by checknode

It is never run while a job is running

Special checks

- Lock to make sure only 1 copy of checknode is running at a time
- Flag file check to make sure the bootup procedure has finished before attempting any checks
  - Hardware techs tend to get impatient waiting on a node to boot
- Save “early” dmesg to avoid losing information that has rolled over
- Certain checks (long-running) only run at boot and cache their results for future runs

Checknode functions

- **logstdout** – log a message to standard output
- **logstderr** – log a message to standard error
- **diagerror** – mark a fatal error and log to standard error
- **compare** – ensure that the output of a command matches a certain string
- **compare2** – ensure that the output of a command matches one of two strings
- **compare_ne** – ensure that the output of a command does not equal a certain string
- **compare_le** – ensure that the output of a command is a number less than given
- **compare_ge** – ensure that the output of a command is greater than given
- **compare_re** – ensure that the output of a command matches a regular expression
- **compare_nre** – ensure that the output of a command does not match a regular expression
- **checkproc** – ensure a given process exists in the process table
- **run** – run a command ensure that the return code is 0
- **journalgrep** – check the system journal for a given match. Cache the result so subsequent checks only need to query messages since the previous check

Some Examples

```
checkproc munged

compare "$(/usr/sbin/dmidecode -s bios-version)" "1.6.2" "BIOS version incorrect"

compare "$(ps axo stat|grep -c D)" 0 "Processes stuck in IO Wait (D)"

compare_ge $(awk '/MemAvailable/ {print $2}' /proc/meminfo) 460000000 "Available memory”

[ -e /home/cxi_debug/trstest.py ] && run /home/cxi_debug/trstest.py

journalgrep sq_intr 'amdgpu: sq_intr' 'GPU sq_intr - put in HBM sandbox’

for nvme in nvme0 nvme1 ; do
        [ -e /dev/${nvme}n1 ] || diagerror "NVME namespace ${nvme}n1 does not exist”
        SL=$(/usr/sbin/nvme smart-log /dev/${nvme} -o json)
        compare "$(echo $SL | jq .critical_warning)" 0 "${nvme} critical warning”
done

for gpuid in {0..7}; do
        compare_re "$(cat /sys/class/drm/${gpu}/device/current_link_speed)”
                "16(\.0)? GT/s( PCIe)?$" "GPU ${gpu} link speed”
done
```

Differences between machines is expressed in tttax

```
verbose Checking Firmware
<% if @variables.include?("recipe") and
       Gem::Version.new(@variables["recipe"]) >= Gem::Version.new('11.0.3') -%>
compare "$(/usr/sbin/dmidecode -s bios-version)" "1.6.2" "BIOS version incorrect"
<% elsif @variables.include?("recipe") and
       Gem::Version.new(@variables["recipe"]) >= Gem::Version.new('11.0.1') -%>
compare "$(/usr/sbin/dmidecode -s bios-version)" "1.6.1" "BIOS version incorrect"
<% elsif @variables.include?("recipe")
       and Gem::Version.new(@variables["recipe"]) >= Gem::Version.new('11.0.0') -%>
compare "$(/usr/sbin/dmidecode -s bios-version)" "1.4.5" "BIOS version incorrect"
<% else -%>
compare2 "$(/usr/sbin/dmidecode -s bios-version)" "1.4.3" "1.4.5" "BIOS version incorrect"
<% end -%>



<% if @variables["hostgroup"] == "frontier" -%>
compare_ge $(ip neigh|grep -c PERMANENT) 150000 "Permanent ARP entries”
<% end -%>
```

GPU Checks

All GPUs present

ECC Errors this Boot

SMN Failures

xGMI Link Errors

xGMI Bandwidth

DGEMM Performance

EEPROM Issues

vBIOS IFWI Version

Ability to read metrics

Number of retired pages

RM Firmware version

Queue preemption timeout

Total GPU Memory

Available GPU Memory

SlingShot Checks

All Interfaces Present

Serial Number is not Blank

PCIe Link Width

PCIe Link Speed

PCIe Error Rates

uC Firmware Version

Link Status

Link Speed

Pause Mode

LLR Status

Correct AMA

Uncorrectable Errors

Packet Buffer Errors

Credit Underflows

Retry Handler Running

Link Flap Rate

Stuck cxi Services

File System Checks

- NVMe Devices Present
- NVMe Firmware Version
- NVMe PCIe Gen
- NVMe PCIe Width
- NVMe Smart Log Criticals
- NVMe Spare Space
- Persistent Volume Group Present
- All DVS Mounts Present
- Lustre Mounts Present
- df Returns within Timeout
- LNET NIs match AMA

Case Study: Slow MPI All-to-All

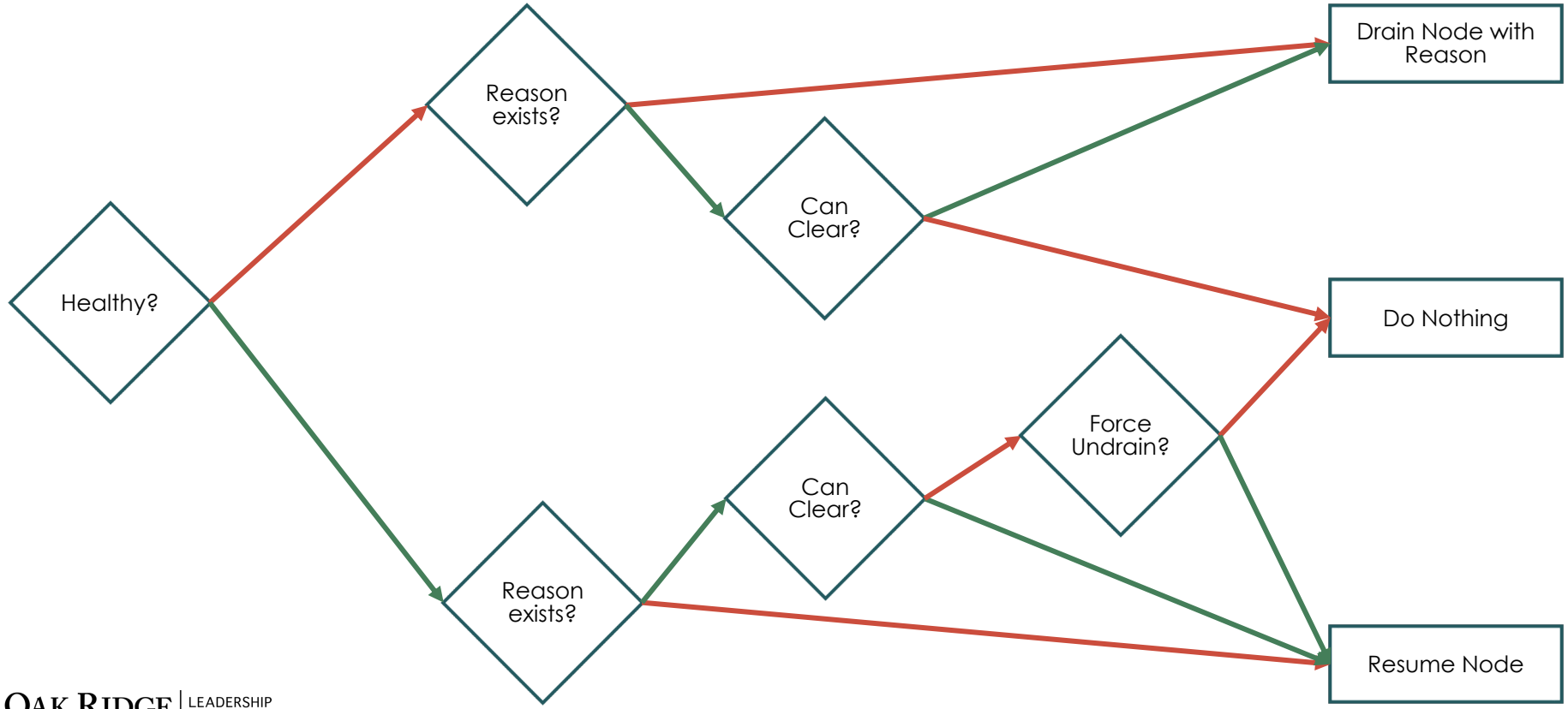
- MPI all-to-all performance was found to be degraded
- Binary search indicated that excluding certain nodes would cause performance to return
- Rebooting the problematic nodes would clear the problem
- No obvious cause from looking at the logs
- AMD tracked it down to a bug in the power management firmware that prevented the CPU from going into burst mode
- Developed a quick, simple test to identify this issue
- Firmware fix came a couple weeks later

Gaps

- BMC (nC) version not available in-band
  - We do not allow the nodes to talk to the hostctrl (BMC) network
  - Request to HPE to make this available has gone unfulfilled

Slurm Integration

- `checknode` reads the current Slurm `State` and `Reason`
- It updates the state if appropriate



Can clear if reason is:
checknode:*
Not responding
Kill task failed

downnodes tool

```
[root@admin1.frontier tmp]# downnodes -h
usage: downnodes [-h] [--nodes NODES] [--state STATES] [--reason REASON]
                 [--comment COMMENT] [--extra EXTRA] [--partition PARTITION]
                 [--exclude EXCLUDE] [--list] [--fanout FANOUT]
                 ...
positional arguments:
  command to run
optional arguments:
  -h, --help            show this help message and exit
  --nodes NODES, -n NODES
                        Only query these nodes
  --state STATES, -s STATES
                        Node states and state flags to include
  --reason REASON, -r REASON
                        Reason search string to match
  --comment COMMENT, -c COMMENT
                        Comment search string to match
  --extra EXTRA, -e EXTRA
                        Extra search string to match
  --partition PARTITION, -p PARTITION
                        Slurm parition to query
  --exclude EXCLUDE, -x EXCLUDE
                        Nodes to exclude from checking
  --list, -l            Output as a comma-separated list
  --fanout FANOUT, -f FANOUT
                        Fanout for parallel commands
```

downnodes

```
[root@admin1.frontier tmp]# downnodes -r rebooted
frontier02235 x2105c3s5b0n0 3d  Node unexpectedly rebooted
frontier02298 x2105c7s4b1n0 3d  Node unexpectedly rebooted
frontier02431 x2106c7s7b0n0 1d  Node unexpectedly rebooted
frontier02794 x2109c6s4b1n0 1d  Node unexpectedly rebooted
[root@admin1.frontier tmp]# downnodes -r rebooted -l
frontier[02235,02298,02431,02794]
[root@admin1.frontier tmp]# downnodes -r rebooted hostname
downnodes: frontier02235: exited with exit code 255
frontier02431: frontier02431
frontier02794: frontier02794
frontier02298: frontier02298
frontier02235: ssh: connect to host frontier02235 port 22: No route to host
```

Regular Checks

- File system or network issues can drain nodes in bulk
- When the system recovers, we would like the nodes to be returned for new jobs
- A 15-minute cron job runs `downnodes` to run `checknode` against any nodes with a drain reason that starts with "checknode"

SEC Integration

- Simple Event Correlator has been used to monitor Jaguar, Titan, Summit, and Frontier
- SEC runs on the admin node, all the leader nodes, and the Slurm controller node
- Watches controller, console, syslog, and slurm controller logs
- Certain failures will drain a node and set the reason
  - Most common is Slurm-detected node failure

SEC Node Failure

```
Node name -> xname: frontier04123 x2208c1s5b0n0
[2023-05-06T09:08:55.312] Killing JobId=1315086 on failed node frontier04123
Issuing command: scontrol update node=frontier04123 comment="app.the_user.test.nodefail.j1315086.n8192"
reason="app.the_user.test.nodefail.j1315086.n8192" state=drain


SEC has not caught any errors on frontier04123 in the past 10 minutes. Trying svtest...


frontier04123 didn't respond to ping. Node controller: x2208c1s5b0 is responsive, running svtest...
TLNC detected. Please wait...
node 0 is off. Current MMRs are empty. Checking captured MMRs...
This node had a power failure. Printing capture MMRs...


R_NFPGA_TLNC_PWR_CSR_CAP
************************
  early_pg_error                  = 0
  emergency_shutdown              = 0
  epd_halt                        = 1
  es_flt_ctlr_soc_s0_c0           = 0
  es_gpu0_thermtrip               = 0
  es_gpu1_thermtrip               = 0
  es_gpu2_thermtrip               = 0
  es_gpu3_thermtrip               = 0
...
  pg_vdr_abcd_s0_c0               = 1
  pg_vdr_efgh_s0_c0               = 1
  pg_vpp_abcd_s0_c0               = 1
  pg_vpp_efgh_s0_c0               = 1
  pg_vtt_abcd_s0_c0               = 1
  pg_vtt_efgh_s0_c0               = 1
  pwrgd_out_c0                    = 1
  pwrok                           = 1
  r_nfpga_tlnc_pwr_sts_cap        = ffffeff
  unused_31_28                    = 0


GPU 1 power fault
```

Hardware Engineer Workflow

- Triage engineers are assigned to rows and evaluate failures
- Nodes and their partners are drained and/or reserved with MAINT reservations
- When the nodes are idle, a hardware ticket is entered
- The physical work is completed
- The triager runs an in-depth test harness to determine if the issues are fixed

Future Work

- Continue to add checks as problems are identified

- Use the slurm extra field for the hardware techs to schedule draining
  - Automatically partner-aware, so partner nodes will drain when a hardware issue is stored to the extra field

OAK RIDGE National Laboratory

Questions?

ezellma@ornl.gov

https://github.com/olcf/frontier-checknode