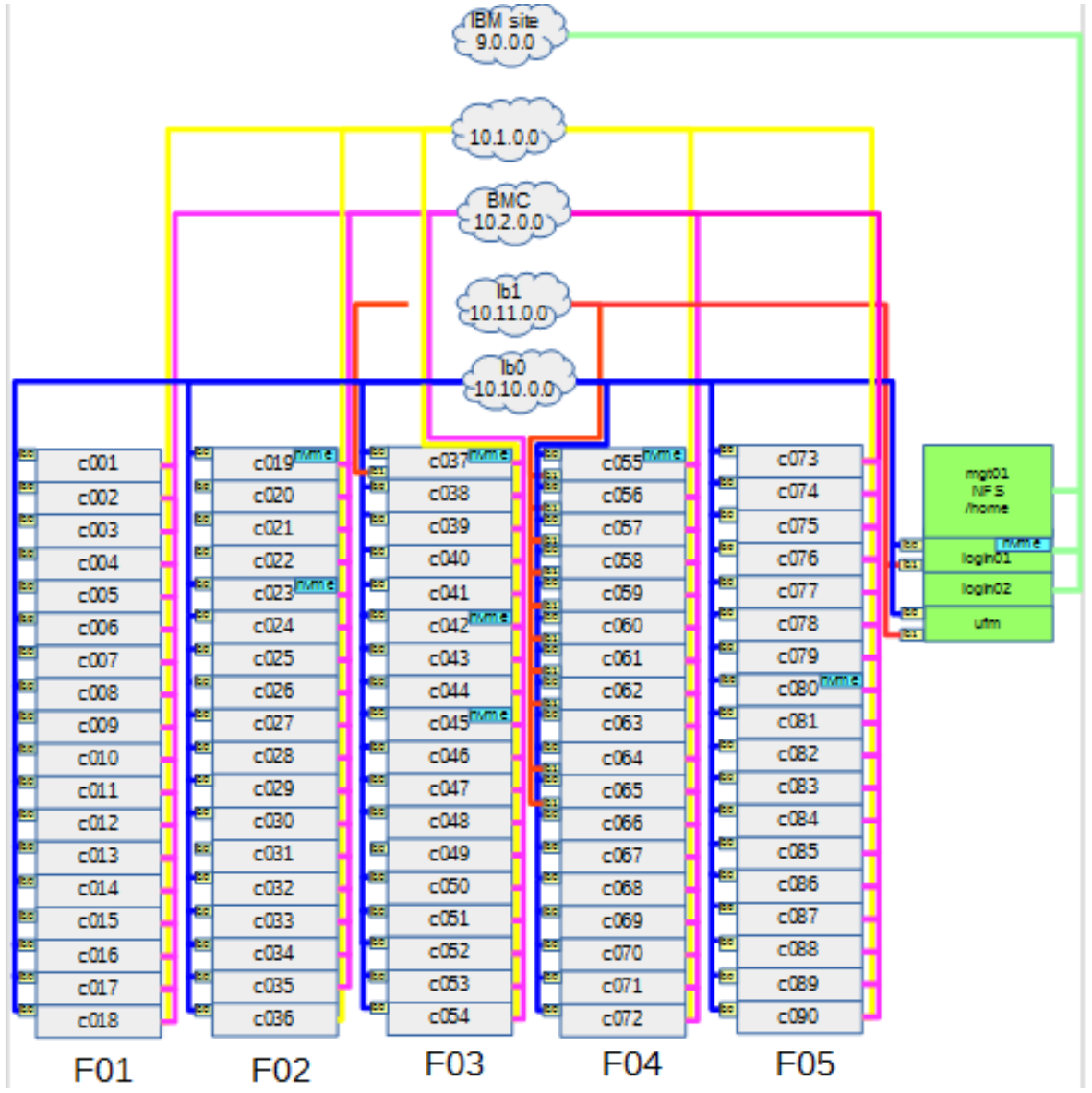
IBM Research
Cluster Management

(a journey)

R92 Cluster.

AKA Minsky



Current use of the R92 cluster

- Testing of CORAL workloads.
- Mellanox and Nvidia performance testing.
- Kernel work.
- Advanced testing of:
  - APEX workloads.
  - Machine Learning.
  - Spark

- Current user base (about 80 researchers).
- Current Administration.
  - Software and configuration (1) (with periodic additional help).
  - hardware (1).

Previous Research Clusters

- Tuletta 10 nodes
- Firestone 20+ nodes.

- Tuletta And firestone nodes also used to explore Ubuntu, Open Stack and Containers for HPC.
- Also, provided valuable experience with the current state of xCAT, and LSF.
  - We used this experience to influence how we deployed the R92 cluster.
- Our site was one of the first to get Tuletta boxes so, we were on the leading edge for xCAT working with them.

Goals of the R92 cluster.

- Provide our researchers a cluster to do HPC testing.
- Be able to isolate and reconfigure nodes for other work (i.e. kernel work and compiler work).
- Deploy software updates rapidly.
- Add new software (as requested) and bring on line rapidly (often within hours of the request).
- Provide a HPC model that is close to what the CORAL customers are using.
- Keep disk-full (login and friends) nodes and disk-less software nodes in sync.
- Be able to rapidly reconfigure hardware (change out IB cards, add nvme cards, etc).
- Keep track of xCAT and other configuration changes.
- preemptive notification of problems and failures (prior to CSM availability).
- Early deployment of ESS without the final ESS servers.

Learning from Experience

- Tuletta and Firestone Clusters.
  - “seat of the pants” administration.
  - no source control of changes (no one at IBM had done source control for xCAT yet).
  - Waiting for users to notify us of a problem sometimes resulted in chasing after a crash or lockup 2 days old.
  - Early software from Nvidia and Mellanox is buggy.
- Experienced IBM Administrators are used to:
  - 1 admin only touching a cluster.
  - applying changes directly to files as “root”.
  - relying on memory about what changed and when.
- Keeping disk-full and disk-less configurations in sync can be a lot of work.
  - lots of files doing the same job but differently duplicated between the configurations.

xCAT/LSF and Friends Source control.

- Source control is usually a push model
  - make a change to source control files.
  - test.
  - run a program to install those files.
- Current admin practice (when we are able to get help) is to modify the file in place.
- Our Spproach to this source control is:
  - to mirror the in their homes to a source control directory (rsync)..
  - run a program every day that checks if they match the working files,
  - send out “scold” mail to anyone that has privileges to modify the files if they don’t match.
  - Check and Mail is done with a “python” script and a “yaml” configuration file which informs the script where the files are and what type of comparisons are needed.

xCAT/LSF and Friends Source control (cont)

- Types of comparison.
  - Entire directories with exceptions.
    - used for xcat files referenced by xcat definitions.
  - explicit list of files (management fstab, network config files, etc).
  - find all files matching a specific search pattern (jenkins test configuration files).
  - xCat definition files.
    - xCat does not actually store definitions in files, so we had to fabricate them.
      - “lsdef –t <object-type> -z –l” produces a file a stanza file we can put under source control and then later use to recreate the definition.
      - python script is used to strip out “attributes” that are transient (like status time) and “attributes” that actually come from “group” definitions.
    - We actually found xcat errors with this changes in the definitions that no user actually made, but were produced by bugs in xCat.
- This allows us to also answer the question:
  - “hey” collective performance dropped after xyz date. What changed?

disk-less, disk-full synchronization.

- We are using the same model for program installs that is part of the CORAL “plan of record”
  - install “everything” on the diskboot image.
  - think LLNL is using this model.
- The Research install:
  - larger than the CORAL test team’s install
    - because of our mission is larger than the CORAL test groups
    - it includes additional packages (such as cuda-dnn, IBM advanced tool chain, etc)...
  - We are up to a 4.5GB download and a 16Gb foot print so far.
  - We are looking into using “modules” and ORNL model of doing at least some of the packages as network installs to mitigate this growing larger in the future.
- Login/compile nodes need to match the software of the compute nodes.
  - Solution, postscripts that can be executed both as part of the “genimage” process to do a diskless image AND a executed as a disk-full install as well as a live “updatenode” call.

disk-less, disk-full synchronization (methodology)

- Use a common set of postscripts for the disk-less install and the disk-full install.
- problem:
  - disk-less install builds on the management node
  - disk-full install builds on the target node.
- Solution:
  - xCAT disk-less post-install script is called with parameters and the disk-full postscript is not
  - we can use this to distinguish between their execution.
  - A common function to all installs is the key. It's basic rule is:
    - if we have a root directory as parameter 1,
      - execute the original script under chroot.

disk-less, disk-full synchronization (methodology)

- problems encountered.
  - some postscript installs required proc and dev to be mounted
    - these MUST be unmounted on exit, so we had to use script “traps” to make sure they were unmounted.
    - leaving these mounted and deleting the image directory resulted in needing to reboot the management node before ‘genimage’ would work again.
  - ctl-c could still leave some mount points, needed to get an xcat change that it checked for mount points prior to removing the directory with “rmimage”.
- Side-effect.
  - compute nodes could have new software added, without having to reboot. This gave us more options for responding to our users update requests.

post-install directory.

- A single post install script is referenced in the xCAT osimage object.
- That postscripts looks in the common postscripts.d subdirectory.
  - executes all executable files in that subdirectory.
- Then it looks in its local postscripts.d subdirectory.
  - executes all executable files in that subdirectory.
- This allows us to do differences between login and compute nodes, but have almost all of the installs common.

post-scripts directory

```
/install/postscripts/custom
|-- rh73_compute/
|   |-- compute.postboot*
|   |-- compute.postinstall*
|   |-- postboot.d/
|   `-- postinstall.d/
|-- rh73_login/
|   |-- compute.postboot*
|   |-- compute.postinstall*
|   |-- postboot.d/
|   `-- postinstall.d/
`-- rh7x_common/
    |-- functions
    |-- postboot.d/
    `-- postinstall.d/
```

```
/install/postscripts/custom/rh7x_common/postinstall.d
|-- 000-install-yum-repos*
|-- 010-install-ldap*
|-- 020-install-nfsmounts*
|-- 030-config-lsf*
|-- config-libnuma*
|-- config-lsf*
|-- install-advtools*
|-- install-cuda8*
|-- install-gpfs*
|-- install-ibm-psdk*
|-- install-ibm-sdk*
|-- install-ibm-spectrummpi*
|-- install-ibm-xlc*
|-- install-ibm-xlf*
|-- install-ldap*
|-- install-mellanox*
|-- install-nfsmounts*
|-- install-nvidia-dcdiags*
|-- install-otherpkgs*
|-- install-pam-lsf*
|-- install-pgi*
|-- install-ppedev*
|-- install-pperte*
|-- install-x-essl*
|-- install-x-ibm-pessl*
|-- mellanox/
|   |-- mlnxofed_ib_install*
|   `-- mlnxofed_ib_install.v2
`-- update-limits*
```

Pre-CSM RAS system alerts.

- Since we did not start out with CSM we needed a way to alert us to problems.
  - crashed nodes.
  - zombie tasks.
  - unexpected configuration changes
    - (someone installed or removed an ib card).
    - ib cable connected or disconnected.
    - nvme card moved.
  - LSF jobs over due and un-killable.
- Solution (jenkins CI),
  - small, self contained jenkins tasks that check things at intervals (15 min, 60min, and 24 hours).
  - Jenkins tasks notify the Admin's via e-mail that something went south.
  - and provide a dashboard about the overall health.
- When CSM diagnostics are available, we can trigger them with the same Jenkins CI framework,
- This also provides a way to check that the CSM ras system is catching things, if we see one of these tests go off without a corresponding CSM ras event, then we have an "escape...

PRE CSM node status.

- To run the Jenkins tests reliably we need to know what to expect.
  - some nodes are down for repair.
  - some nodes have 1 ib card, some have 2.
  - some nodes have nvme cards, others don’t.
  - Things change constantly.
- As a temporary measure (until we can get a CSM that will keep its underlying DB around between updates).
  - use the xCAT “usercomment” node attribute to encode this information.
    - `lsdef  c460c0[02,03] -c -i usercomment`
    - `c460c002: usercomment=[online]`
    - `c460c003: usercomment=[offline]`
    - `c460c055: usercomment=[online,nvme,ib1_down]`
- The “[]” allow easy parsing by the test scripts and they also allow the user of the field for general comments.
- we use the “offline” status also to remove nodes from lsf with a “dynamic” group script.

backup

jenkins dashboard.

S	W	Name ↓	Last Success	Last Failure	Last Duration
		r92-admin-tests » r92-chk_noping	9 min 38 sec - #2398	N/A	1.4 sec
		r92-admin-tests » r92-chk_nvidia_smi	8 min 58 sec - #5866	N/A	2.6 sec
		r92-admin-tests » r92-chk_rpower	21 min - #1269	N/A	2.3 sec
		r92-admin-tests » r92-chk_slot_consistency	14 hr - #221	N/A	2.7 sec
		r92-admin-tests » r92-chk_ufm	8 min 13 sec - #1818	N/A	1.7 sec
		r92-admin-tests » r92-chk_xcat_stat_failed	9 min 28 sec - #3740	N/A	1.4 sec
		r92-admin-tests » r92-chk_zombies	8 min 48 sec - #3367	N/A	2.6 sec
		r92-admin-tests » r92-config-changes	14 hr - #158	N/A	19 sec
		r92-admin-tests » r92-lsf-overdue-jobs	9 min 8 sec - #4663	1 hr 39 min - #4657	0.11 sec
		r92-admin-tests » r92-lsf_unreachable	9 min 18 sec - #3253	N/A	1.4 sec
		r92-admin-tests » r92-switchconf	14 hr - #50	N/A	5.6 sec
		r92-admin-tests » r92_chk_hca_attributes	21 days - #1976	14 hr - #2239	2 min 12 sec
		r92-admin-tests » r92_chk_ip_connectivity	8 min 38 sec - #4022	N/A	1.9 sec
		r92-admin-tests » r92_chk_ipstate	8 min 28 sec - #4031	N/A	6.3 sec

Icon: S M L

Legend RSS for all RSS for failures RSS for just latest builds

example config.xml for source control.

```
name: diff xcat/install/custom
  type: diff
  sysdir: /install/custom
  gitdir: ./xcat/install/custom
  exclude: [sw,".gitignore",".git"]
...
- name: diff lsf/conf
  type: diff
  sysdir: /shared/lsf/conf
  gitdir: ./lsf/conf
  # exclude files we don't want to expose
  exclude:
[".gitignore",".git","*.pem","pamauth.conf","server.pem","users.xml
- name: diff lsf/conf
  type: diff
  sysdir: /shared/lsf/10.1/misc/exec_scripts
  gitdir: ./lsf/shared/lsf/10.1/misc/exec_scripts
- name: diffsel /etc
  type: diffsel
  gitdir: ./system
  files:
    - /etc/sysconfig/jenkins
    - /etc/sysconfig/network-scripts/ifcfg-en*
    - /etc/fstab
    - /etc/yum.repos.d/*
```

```
- name: xcatdef
  type: xcatdef
  gitdir: ./xcat/defs
  tables:
    - site
    - osimage
    - node
    - group
    - network
    - route
    - policy

- name: findsel /jenkins
  type: findsel
  sysdir: /var/lib/jenkins
  gitdir: ./jenkins
```