OSCAR 2006
May 2006

Heterogeneous clusters with OSCAR: Infrastructure

Erich Focht

NEC HPC Europe

Overview

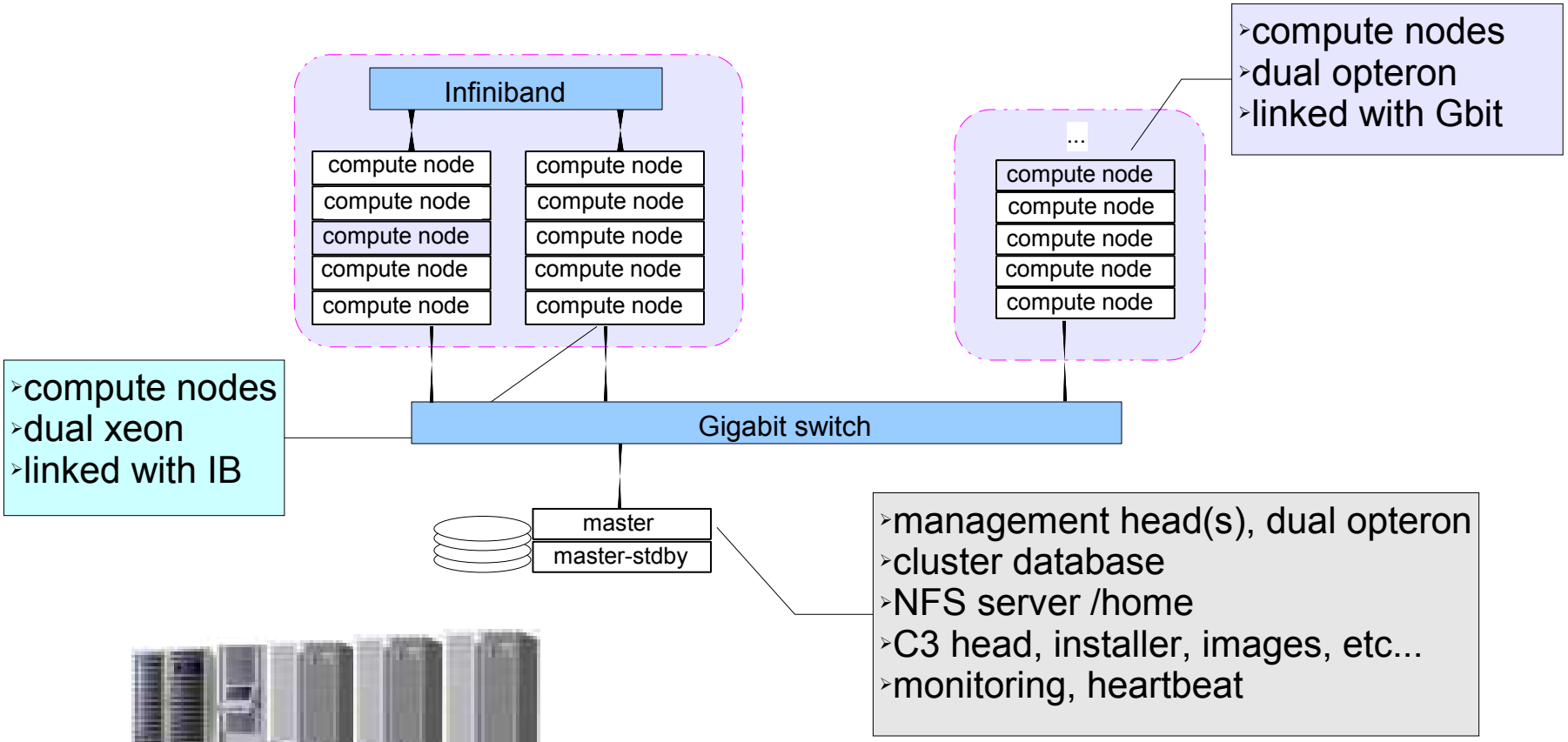
- Why heterogeneous clusters?
- Infrastructure changes
  - package repositories
  - OCA::OS_Detect
  - Prerequisites and generic-setup
  - Yume
  - PackMan
  - SystemInstaller
- Installation
- Administration

Heterogeneous cluster

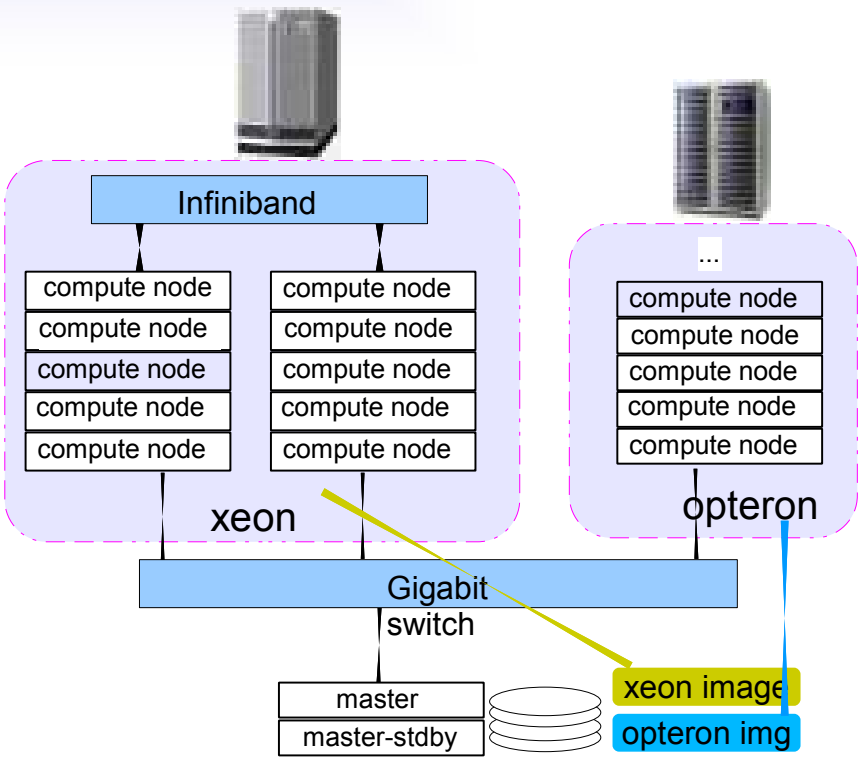
- Why?
  - ... multiple architectures within the same cluster
    - Xeon, IA64, Opteron, Nocona
    - Extension of existing cluster (no additional mgmt node)
    - Safe migration environment
  - ... different distributions inside the same cluster
    - RHEL3/4, SUSE, FC3/4/5, Debian ...
    - e.g. because applications are validated by ISVs for different distros
  - ... multiple interconnect types in same cluster
    - Myrinet, IB, Gbit, ...

Example

- Subclusters have different architecture (and distro)



Cluster of clusters



- Solution:
  - Use OSCAR / SIS with multiple images
  - Simplest: one image per (sub)cluster
  - Don't share images across (sub)clusters
  - Define clients carefully
  - Tweak installer to deal with pxelinux and elilo, create correct dhcpd.conf (for ia64)

Package repositories

- Where should RPMs go when dealing with multiple distros?

Before:

- /tftpboot/rpm

Now:

- /tftpboot/distro/$distro-$ver-$arch : original distro packages
- /tftpboot/oscar/$distro-$ver-$arch : OSCAR packages

- Separated distro from OSCAR packages
- Easier to update/manage distro packages
- Easier to clean up OSCAR packages

Package repositories

- Where should RPMs go when dealing with multiple distros?

$distro is <u>real</u> distro name:
- redhat-el-ws
- centos
- scientificlinux
- fedora
- mandriva

Before:
- /tftpboot/rpm

Now:
- /tftpboot/distro/$distro-$ver-$arch : original distro packages
- /tftpboot/oscar/$distro-$ver-$arch : OSCAR packages

- Separated distro from OSCAR packages
- Easier to update/manage distro packages
- Easier to clean up OSCAR packages

Package repositories

- Where should RPMs go when dealing with multiple distros?

$distro is compatible distro name:
- rhel
- fc
- mdv

because OSCAR deals properly with compatible distros

Before:

- /tftpboot/rpm

Now:

- /tftpboot/distro/$distro-$ver-$arch : original distro packages
- /tftpboot/oscar/$distro-$ver-$arch : OSCAR packages

- Separated distro from OSCAR packages
- Easier to update/manage distro packages
- Easier to clean up OSCAR packages

OCA::OS_Detect

- Detects OS and architecture for:
  - local node (old behavior)

    ```
    $os= OCA::OS_Detect::open();
    ```

  - node image (new)

    ```
    $os= OCA::OS_Detect::open(chroot => $imgpath);
    ```

  - package repository (new)

    ```
    $os= OCA::OS_Detect::open(pool => $pool);
    ```

  - fake detection for finding package manager and compatible distro names when distro is known

OCA::OS_Detect

- Returns hash reference:
  - `$os->{distro}`
    - example: „centos“
  - `$os->{distro_version}`
    - example: „4“
  - `$os->{distro_update}`
    - where available
  - `$os->{compat_distro}`
  - `$os->{compat_distrover}`
  - `$os->{pkg}`
    - „rpm“ or „deb“
  - `$os->{arch}`
    - as returned by ***`uname -i`***
    - detection on image uses something like ***`file /bin/bash`***

Prerequisites and Packages

- Now follow generic-setup format
  - use of RPMS/ subdirectory is deprecated
  - distro/
    - common-rpms/
    - common-debs/
    - $distro$ver-$arch/
  - Example:
    - distro/rhel4-i386/
  - No need to care about .rpm/.deb copying to repository
  - Compatible distro names are used!

Prerequisites Installation

- scripts/install_prereq
  - configuration files:
    - share/prereqs/prereqs.order
    - share/prereqs/*/prereq.cfg
  - deletes/installs packages before config.xml is read
  - Example: (from yume/prereq.cfg)

```
[fedora:3:*]
perl-IO-Tty
!yum
yum-oscar
createrepo-0.4.3-5.1e.noarch.rpm
yume
```

Yume

- Wrapper around ***yum*** (Yellowdog Updater Modified)
  - „Smart rpm“, resolves dependencies, uses cache
- Uses universal package metadata: ***createrepo***

  - Prepare repository cache

```
yume --prepare --repo /tftpboot/fc3-i386
```

  - Export repository through apache

```
yume --export --repo /tftpboot/fc3-i386
```

    > use: --repo http://oscar_server/repo/tftpboot/fc3-i386

  - Install/deinstall packages

```
yume --repo ... -- install|remove packagename
```

  - Install into an image:

```
... -- --installroot $IMGDIR install packagename
```

Yume

- ***--repo ...*** command line args can be omitted in most cases
  - if on master, detecting default repository names
  - on remote nodes: calling scripts/distro-query on the master (via ssh)
- easy way to update nodes
- ***scripts/repo-update***
  - download updates from yum repositories on the internet
  - keep local repository clean (remove old RPMs)

PackMan

- OSCAR package management framework
  - hides specific package manager
  - creating instance:

```
use PackMan;
$pm = PackMan->new();
```

  - for a chroot path:

```
$pm = PackMan->new("/chroot_path/");
```

or

```
$pm->chroot("/chroot_path");
```

PackMan Old API

- Methods:
  - change chroot directory
    ```
    $pm->chroot($newpath);
    ```
  - install packages
    ```
    $err = $pm->install(pkg_file1, ...);
    ```
  - remove packages
    ```
    $err = $pm->remove(pkg1, ...);
    ```
  - query package(s) version
    ```
    @versions = $pm->query_versions(pkg1, ...);
    ```
  - check if package(s) are installed
    ```
    ($inst, $ninst) = $pm->query_installed(pkg1, ...);
    ```

PackMan Extended API

- Relies on ***yume*** and ***rapt*** (has RPM and DEB support!)
- "Smart" package management methods:
  - check "smart" capability
    ```
    $pm->is_smart();
    ```
  - install packages smartly (resolve dependencies)
    ```
    ($err,$out) = $pm->smart_install(pkg_file1, ...);
    ```
  - remove packages (and remove dependencies as well)
    ```
    ($err,$out) = $pm->smart_remove(pkg1, ...);
    ```
  - specify repositories
    ```
    $pm->repo(repo_url1, repo_url2, ...);
    ```
  - generate repository metadata
    ```
    $pm->gencache();
    ```

PackMan Extended API (2)

- “Smart” package management methods:
  - export repositories through httpd

    ```
    $pm->repo_export();
    ```

    - http://hostname/repo/path...
  - unexport repositories (remove them from httpd config)

    ```
    $pm->repo_unexport();
    ```

  - register a callback function which will be invoked for each output line generated by the smart package manager:

    ```
    $pm->output_callback(\&func,args,...);
    ```

  - clean caches of smart package manager

    ```
    $pm->clean();
    ```

  - enable/disable progress output info in output stream

    ```
    $pm->progress(...);
    ```

SystemInstaller

- Many functions in OSCAR
  - build image to be deployed to client nodes
  - setup disk partitioning info
  - prepare autoinstall scripts for systemimager
  - setup DHCP server for installation
  - keeps track of cluster data, NICs, MACs, images in SIS database
- Was hardwired to use DepMan for building images
  - pass list of packages
  - call depman to build list of dependencies
  - detect appropriate dumb package manager method (rpm/deb/...)
  - install each package

SystemInstaller

- Many functions in OSCAR
  - build image to be deployed to client nodes
  - setup disk partitioning info
  - prepare autoinstall scripts for systemimager
  - setup DHCP server for ins
  - keeps track of cluster data database
- Was hardwired to use Dep
  - pass list of packages
  - call DepMan to build list of dependencies
  - detect appropriate dumb package manager method (rpm/deb/...)
  - install each package

SystemInstaller contained yet another package manager abstraction. Not a particularly smart one...

SystemInstaller-OSCAR

- new SystemInstaller: integrated with OSCAR
- removed own distro-setup package lists
  - OSCAR provides them in oscarsamples/
- removed DepMan calls
- removed own package manager abstraction
- only one installation method used: smart PackMan
  - resolves dependencies by itself
  - allows multiple distro/arch support
- added distro/arch selection to Image build panel
  - selection menu built from available distro repositories (or URL files)
- Multiple distro/arch support was never easier!

Installation

- Image integration:
  - `integrate_image` script
    - For importing foreign images into SIS
    - Fixup of things OSCAR otherwise cares about
    - Bad: OSCAR packages need to be handled manually
      - Yume-opkg helps!
  - Some manual tuning of images before install
  - Additional API step: `post_rpm_install_nochroot`
    - Allows per image configuration of packages

Yume-opkg

- OSCAR database package query tool
  - Which OSCAR packages (opkgs) need to be installed?
  - Which RPMs need to be installed for an opkg?

```
yume-opkg --distro DISTRO            \
          --distrover DISTRO_VERSION \
         [--arch ARCH]               \
          --listopkgs|--listrpms [opkg1 [opkg2...]]
```

Administration

- User Administration
  - /etc/passwd : system uids are different for rhel/mdv/suse/debian
  - new ***sync_files*** package
    - deals with distro templates
- restrict commands to subcluster
  - SC3 package
- System Monitoring
  - new ganglia configurator
- Resource Management
  - existing resource managers are flexible enough

Extension: SC3

- Sub-Cluster Command and Control
  - Subcluster selection:
    - `--image <imagename>`
    - `--domain <domainname>`
    - `--nodelist "node1 node2 ..."`
  - Determine active nodes in subcluster
    - gmember
    - cexec hostname
  - Subcluster.pm perl class
    - interacts with SIS database
    - translates subclusters into C3 cluster_name:node_range
    - use dynamically generated scalable C3 config file

SC3: parallel commands

- `scexec subcluster -- command args`
  - `--image <name> [--execimg] [--onlyimg]`
  - execute command on active nodes of subcluster defined by image <name>
  - uses cexec, could use gexec
  - `--execimg :` Execute command in the image, too
    - chroot if same architecture as master node
    - ssh + chroot on first active member node if architectures different. Image directory must be NFS-exported by master and mounted on client nodes
  - Advantage: only one command for managing nodes and image, easier to keep in sync and avoid mistakes.

SC3: parallel commands (2)

- `scpush subcluster source target`
  - `--image <name> [--writeimg] [--onlyimg]`
  - Source : file or directory (requires slight modification of cpush)
  - Target : directory
- `scrpm subcluster -- rpm_options`
  - `--image <name> [--execimg] [--onlyimg]`
  - Query/list/install/remove RPMs on nodes
  - Copies RPMs to shared filespace, first
  - Same RPM actions to image
  - Uses remotely mounted image if architectures differ

Conclusion

- Infrastructure changes
  - ... quite a few ...
  - need to be understood
  - well integrated into OSCAR
- Complex heterogeneous setups are now easy to handle with OSCAR!