KestrelCluster

Jon Ander Hernández

itsas

EHUko Software Librea sustapenerako taldea

16 de Diciembre de 2011

Introducción

¿Qué es KestrelCluster?

KestrelCluster es un conjunto de herramientas que facilitan *la configuración* y *la gestión* de un cluster.

- La instalación y configuración se realiza mediante un sistema de *plantillas* y *scripts*[1].
- Los nodos se registran simplemente iniciándolos, asociando su mac con un nombre y un grupo.

[1] Normalmente contienen comandos sed para editar los ficheros.

- Es un paquete para Debian Squeeze y Ubuntu 10.10, 11.04, 11.10[2].
- Está diseñado con la finalidad de ser extendido mediante módulos. De modo que podamos añadir soporte para distintas APIs, programas, ...
- Sigue la filosofia KISS. "keep it simple and straightforward".
  Si hay algo que se puede hacer usando un mecanismo del sistema lo usamos

[2] Aunque recientemente hemos descubierto que la versión estable dejó de funcionar a partir de la versión final de Ubuntu 11.04

¿Qué es un cluster?

"Is a *group* of *linked* computers *working* together closely thus in many respects forming a single computer."

¿Por y para qué querríamos usar un Cluster?

"to *improve performance and availability* over that of a single computer"

"while being much more *cost-effective* than single computers of comparable speed or availability."

Clasificación de clusters

La clasificación se realiza según la función que desempeña :

High Availability (HA)

"They operate by having redundant computers or nodes which are then used to provide service when system components fail."

Load balancing

"is a computer networking methodology to distribute workload across multiple computers or a computer cluster, network links, central processing units, disk drives, or other resources, to achieve optimal resource utilization, maximize throughput, minimize response time, and avoid overload."

High-performance computing (HPC)

"uses supercomputers and computer clusters to solve advanced computation problems"

De todos ellos, este es uso caso habitual de Kestrel.

HPC y Message Passing Interface

- is an *API specification* that allows processes to communicate with one another by **sending and receiving messages**.
- is de **facto standard** for communication among processes that model a parallel program running on a **distributed memory system**.
- MPI's goals are high performance, scalability, and portability. MPI remains the dominant model used in high-performance computing today.
- *Provide* :
  - essential virtual topology
  - synchronization
  - communication functionality between a set of processes
  - in a language-independent way

Implementaciones MPI

Existen multitud de implementaciones. De las libres las más importantes son *MPICH* y *OpenMPI*.

Las preguntas que nos hicimos fueron:

- ¿Soportamos las 2?
- ¿Damos a elegir una de las 2?
- ¿Se pueden instalar de manera paralela?

Charm++

Denis, uno de los desarrolladores, quería usar un API muy interesante con Kestrel.

- It provides high-level mechanisms and strategies to facilitate the task of developing even highly complex parallel applications.
- Charm++ programs are written in C++ with a few library calls and an interface description language for publishing Charm++ objects. Charm++ supports multiple inheritance, late bindings, and polymorphism.
- ...
- `http://charm.cs.uiuc.edu/`

Single System Image

Mucha gente encontrará que prefiere no tener que reescribir sus programas para usar MPI o Charm++. ¿Tenemos alguna opción?

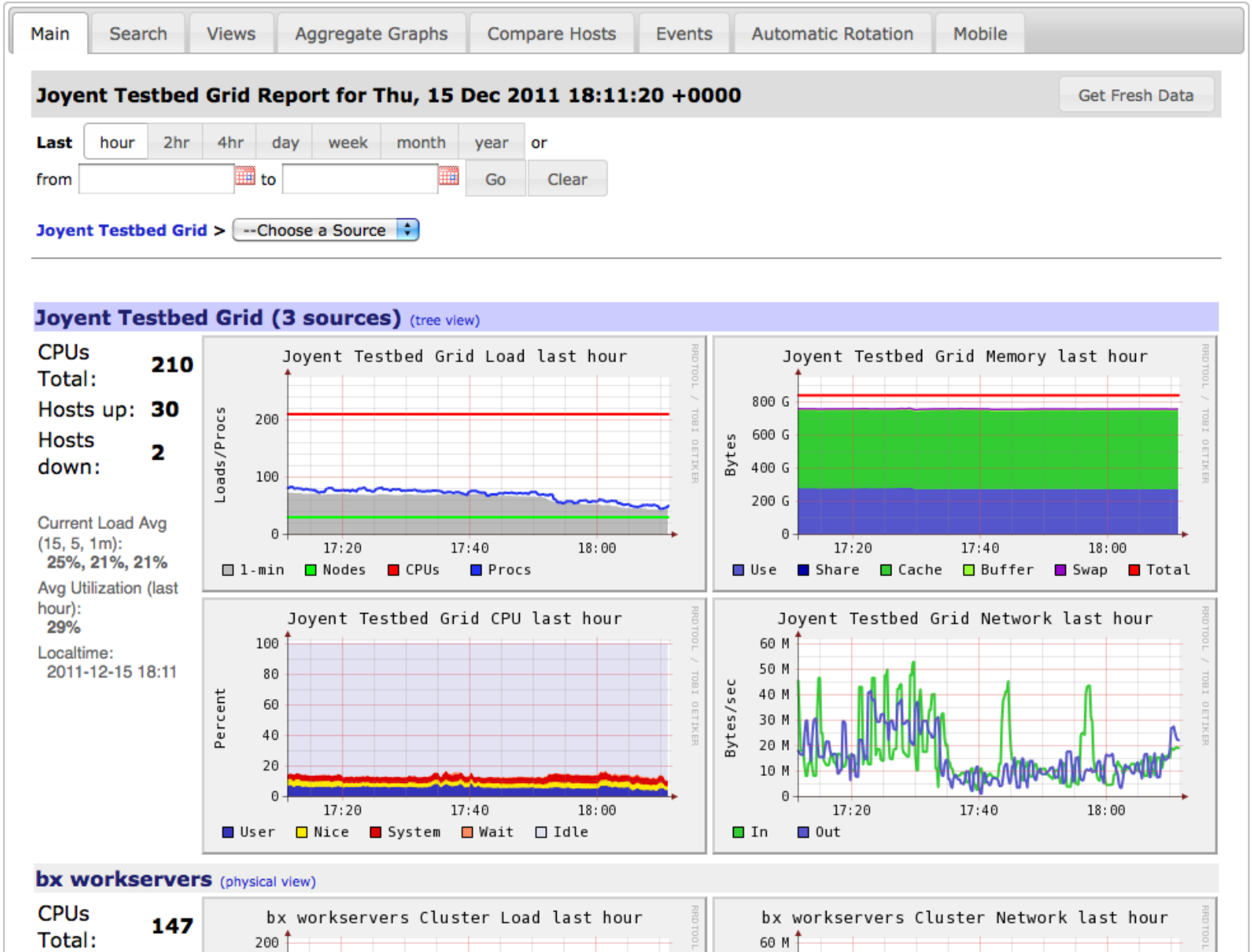
Single System Image (SSI) cluster is a cluster of machines that appears to be one single system.

- OpenMOSIX
- LinuxPMI
- Kerrighed
- ...

Aparte de frameworks también hay muchas herramientas casi imprescindibles

Monitorización

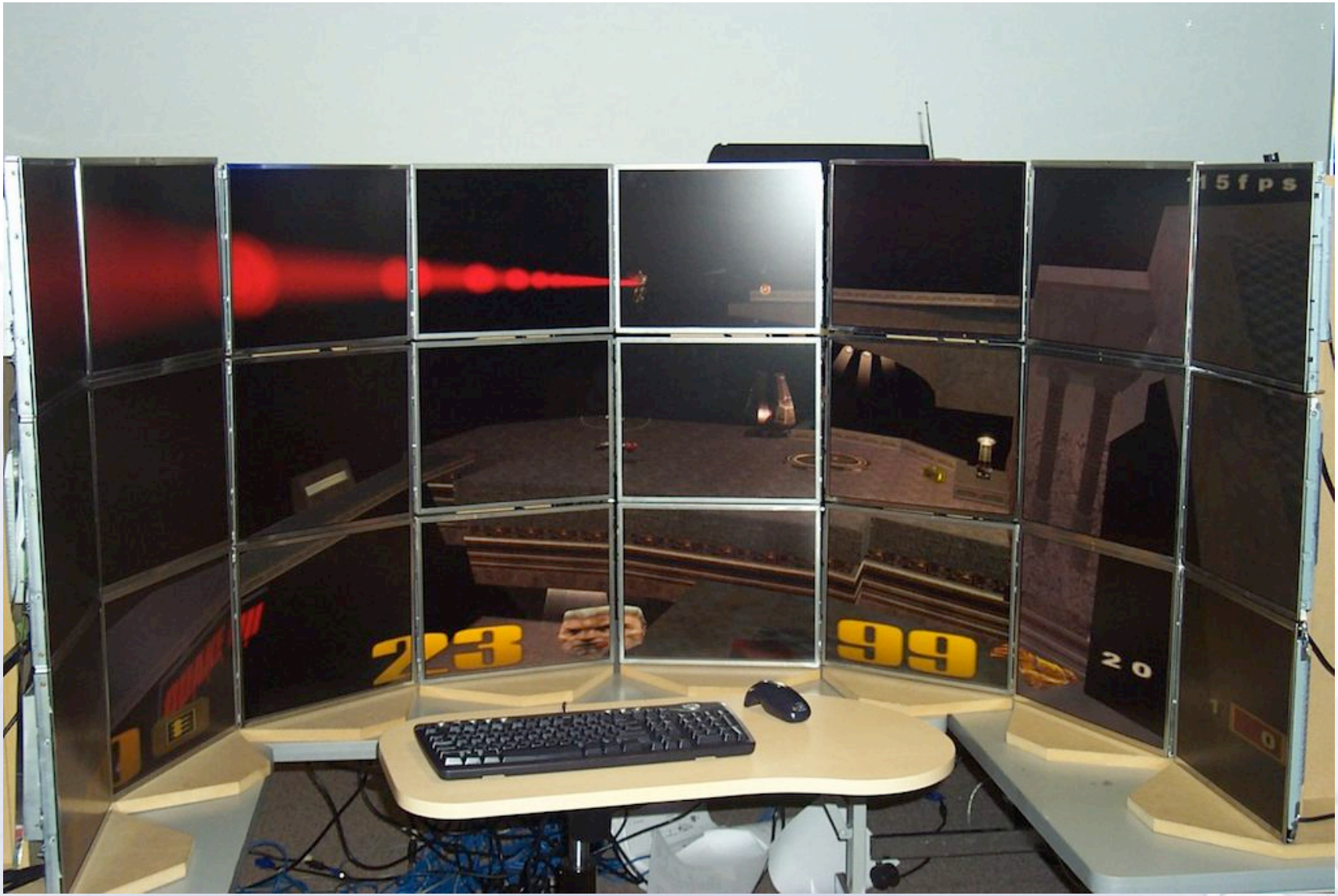
Ganglia



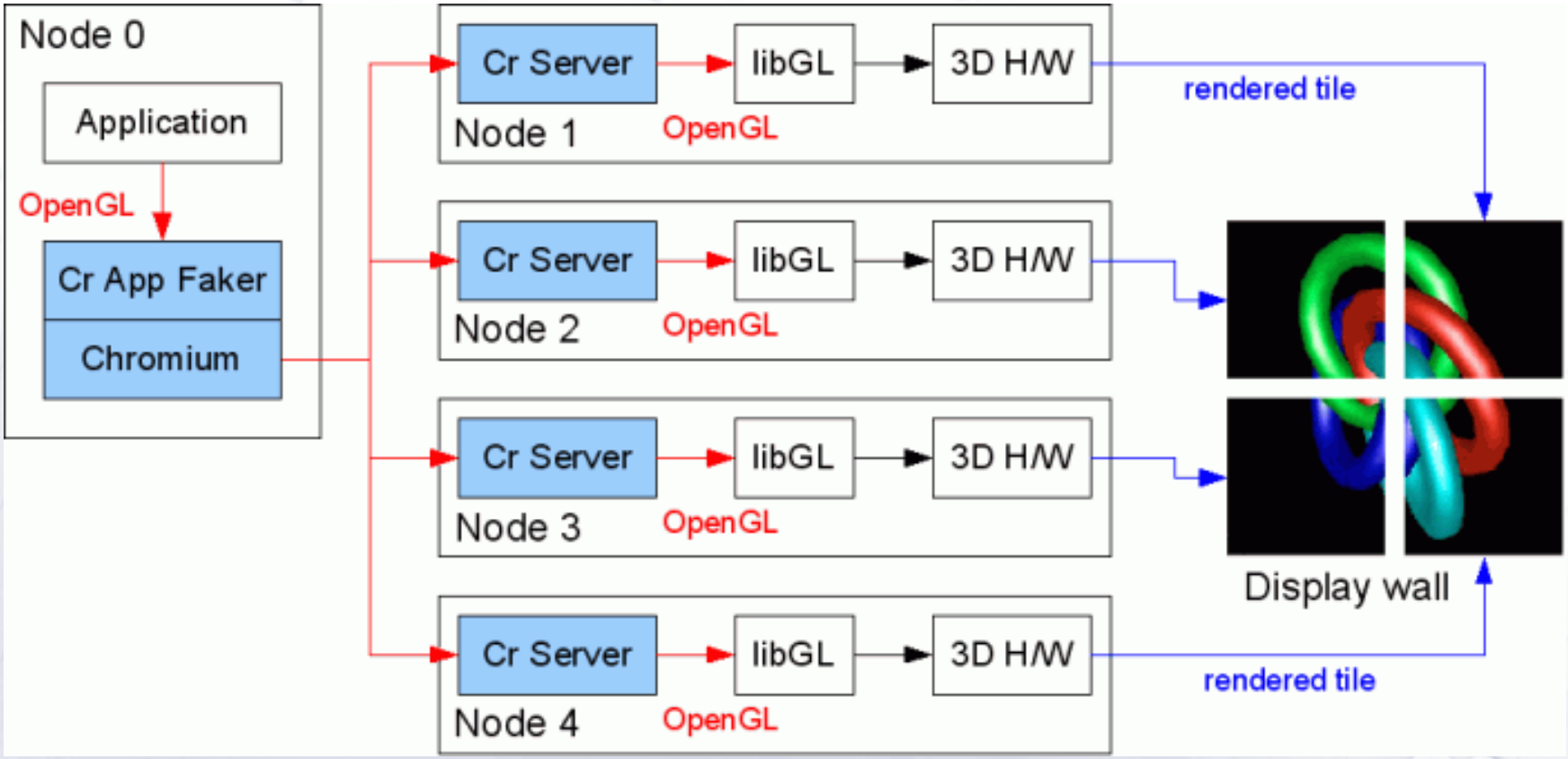
Planificadores de tareas

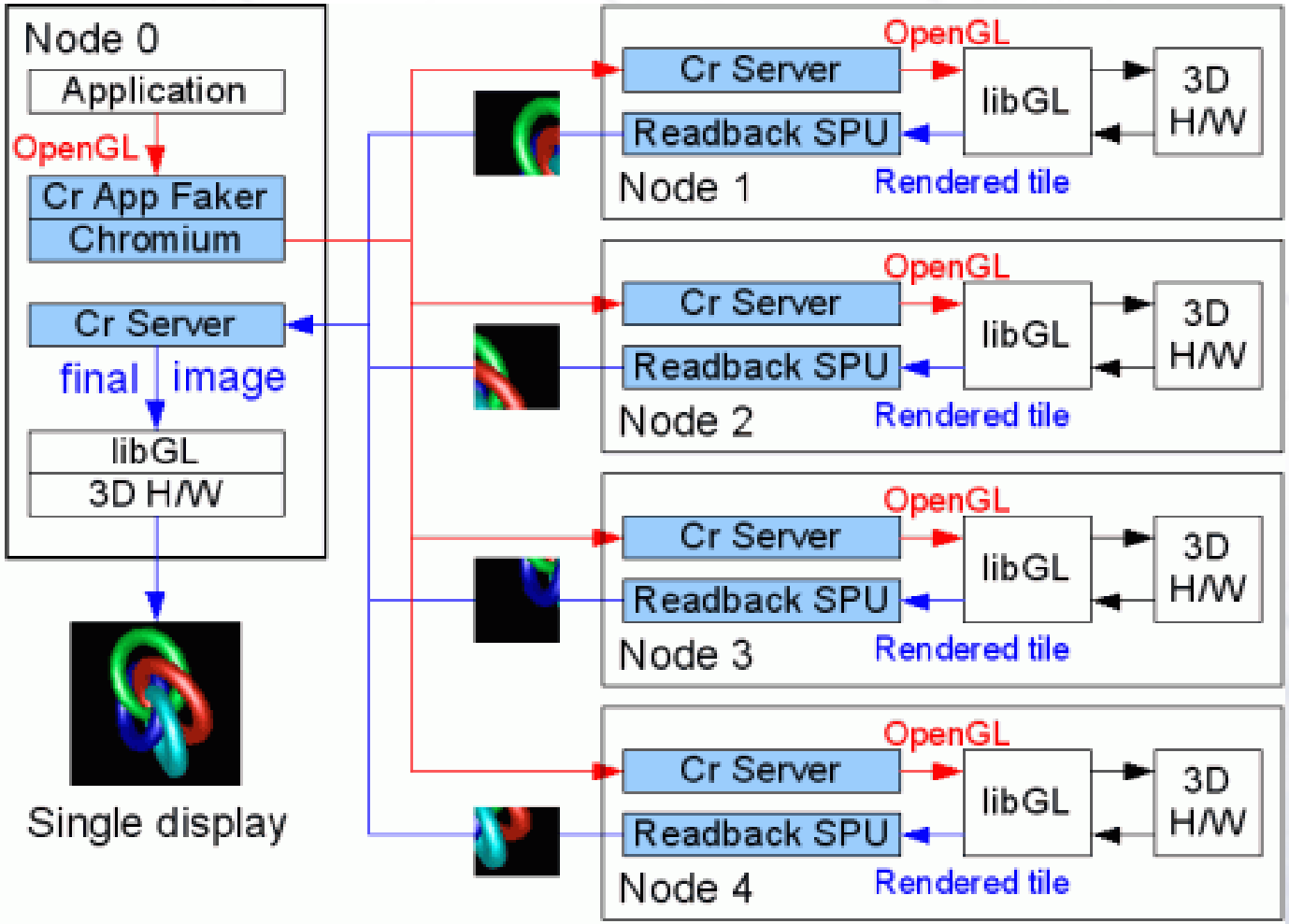
- SLURM
- Torque
- ...

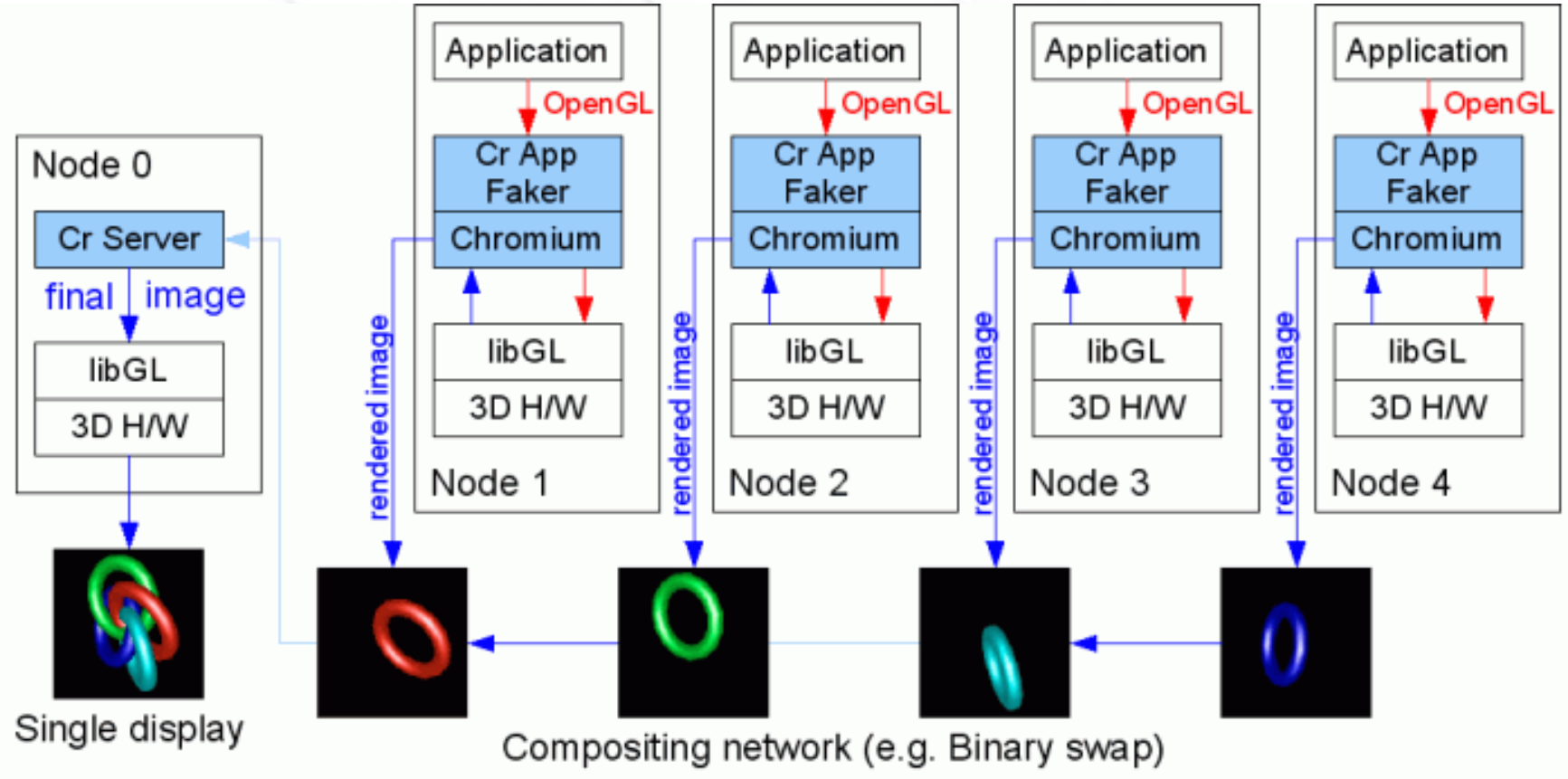
¿Todos los clusters son para *computar datos* o para *redundar servicios*?



Xdmx + Chromium, `http://www.plastk.net/`


Node 0
Application
OpenGL
Cr App Faker
Chromium
Cr Server
libGL
3D H/W
Node 1
OpenGL
Cr Server
libGL
3D H/W
Node 2
OpenGL
Cr Server
libGL
3D H/W
Node 3
OpenGL
Cr Server
libGL
3D H/W
Node 4
OpenGL
rendered tile
rendered tile
Display wall


Node 0
Application
OpenGL
Cr App Faker
Chromium
Cr Server
final image
libGL
3D H/W
Single display
OpenGL
Cr Server
Readback SPU
libGL
3D H/W
Node 1
Rendered tile
Node 2
Node 3
Node 4


Node 0
Cr Server
final
image
libGL
3D H/W
Single display
rendered image
Application
OpenGL
Cr App Faker
Chromium
libGL
3D H/W
Node 1
Node 2
Node 3
Node 4
Compositing network (e.g. Binary swap)

Nuestra conclusión:
Hay muchas necesidades diferentes que podrían compartir una base común y usarse un sistema con módulos

¿Cómo monto un cluster Linux?

- Usar una distribución especifica.

  Problema: Como todas las distribuciones, normalmente acaban desactualizadas o es difícil instalar nuevos programas.

- Usar una buena distribución[3] y configurarla.

  Problema: Se requiere muchos conocimientos ya que hay que saber configurar servicios muy diversos.

- Usar una buena distribución y una buena guía.

  Problema: Las guías suelen ser bastante complicadas de seguir, suelen quedarse desactualizadas, y ¡ai de ti como te salga algo distinto de la guía!

[3] En Debian por ejemplo todo el software citado ya está empaquetado

Guias interesantes

- Debian Clusters for Education and Research: The Missing Manual
  `http://debianclusters.org/index.php/Main_Page`

- How-to: John the Ripper on a Ubuntu 10.04 MPI Cluster
  `http://www.petur.eu/blog/?p=59`

Nuestra conclusión:

Tenemos distribuciones que nos permiten hacer todo lo que quisieramos, pero necesitamos algo que nos ayude a configurarlas de una manera más sencilla

¿Cómo funciona un cluster Linux?

¿Cómo funciona un cluster Linux live? / Ingredientes

- Compartir el sistema de archivos raíz mediante NFS 3.

  El 4 es más complejo y mucho más eficiente aunque nosotros sólo lo soportamos en Debian
  Requiere configurar el fichero */etc/exports* y el fichero */etc/fstab*[4].

- Compartir el directorio home :
  - Necesitamos compartir las propias aplicaciones entre los nodos, datos, resultados, etc...
  - Normalmente instalamos las bibliotecas que usan nuestras aplicaciones también en el home.
  - Cada usuario tiene sus cosas separadas.

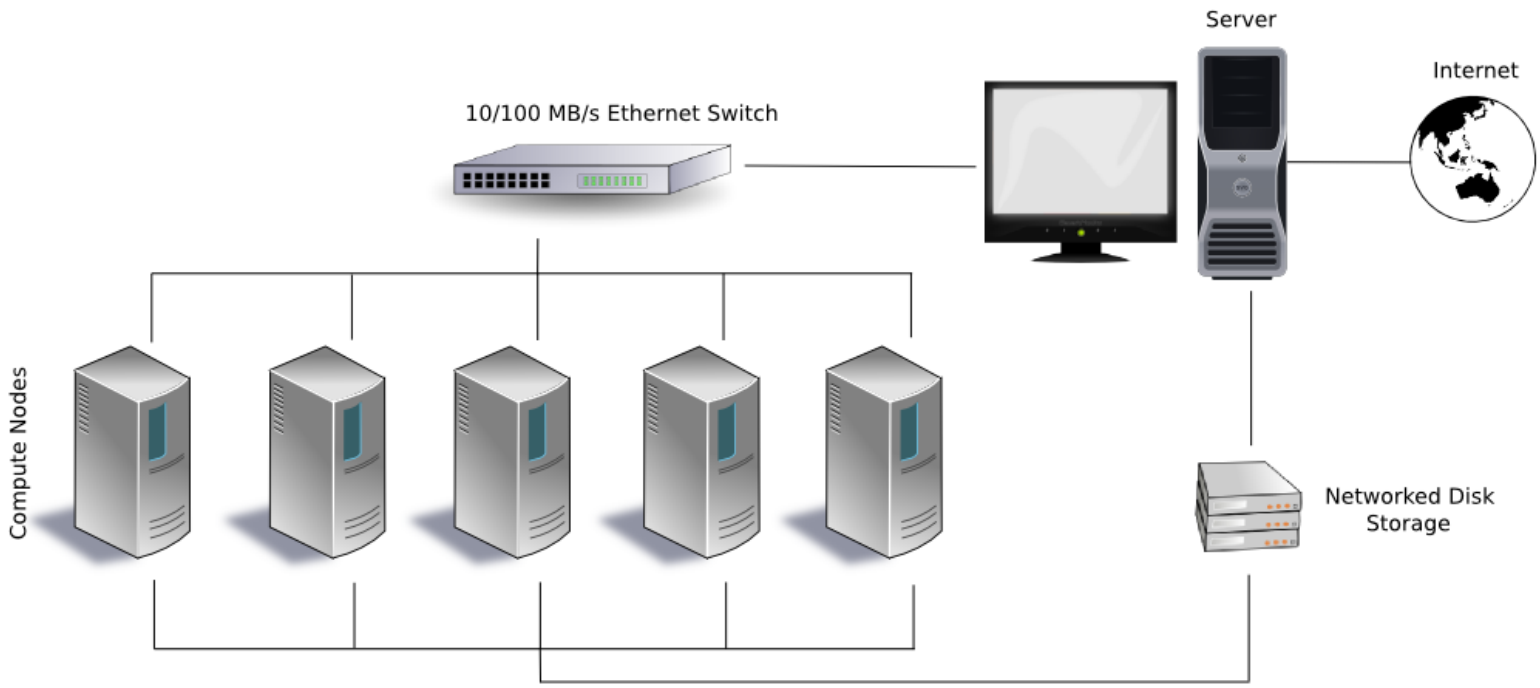
  Es interesante usar un sistema de archivos distribuido frente a uno centralizado :
  - Red Hat Global File System (GFS)
  - Oracle Cluster File System (OCFS)
  - ...

[4] NFS4 requiere hacer un mount bind al directorio */etc/exports*

- SSH. Una aplicación MPI se invoca usando SSH.

1. Necesitamos crear certificados digitales para realizar las conexiones (el acceso por contraseña no es una opción con varias maquinas... ;-).

2. Creamos claves para cada usuario y las añadimos como claves autorizadas para el propio usuario.

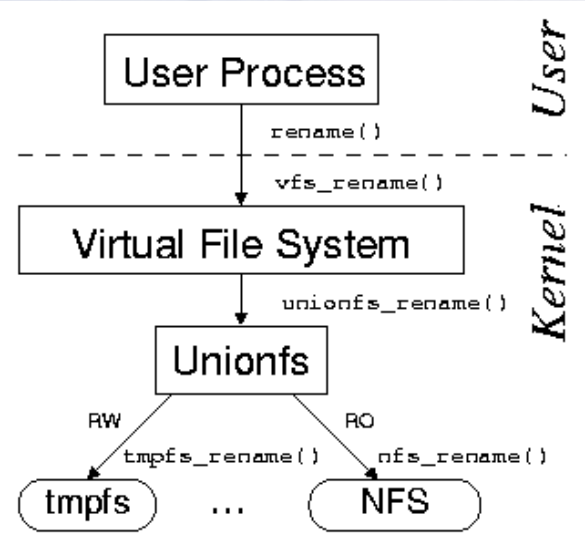


- Arranque por red : Preboot Execution Environment
  - Elementos necesarios :
    1. Servidor de DHCP → dhcpd vs dnsmasq
    2. Servidor de TFTP → tftpd-hpa vs dnsmasq
    3. Bootloader PXE → grub, grub2 e pxelinux (forma parte de syslinux)
  - Funcionamiento :
    1. La BIOS durante tras el POST delega el arranque en la tarjeta de red usando una PXE ROM.
    2. PXE firmware broadcasts a DHCPDISCOVER packet extended with PXE-specific options.
    3. El servidor de dhcp responde indicando un path y una ip de donde descargar el NBP usando TFTP[5].
    4. El bootloader emplea las APIs del firmware PXE para localizar el kernel y el initramfs a través de la red (Pre-boot, UDP, TFTP, Universal Network Device Interface (UNDI)).
  - Posibilidades :
    - PXE permite configurar el arranque según la mac de la máquina (el kernel, el initramfs y los parámetros de arranque).

[5] Una especie de FTP que usa udp

¿Cómo funciona el modo live?

- La raíz del sistema en los nodos es de sólo lectura. Si fuera un sólo nodo se podría plantear de escritura, pero no si está compartido. Un sistema funcional necesita poder escribir /var/log, /var/run, /var/lib, ...
- Solución : UnionFS. Permite unificar capas de sistemas de archivos.
- Unificamos un tmpfs, un sistema de archivos *RAM*, en modo RW encima del sistema de archivos NFS en modo *lectura*.



¿Cómo se consigue arrancar una maquina cuyo sistema está en otra?

- Linux como kernel UNIX, por diseño sólo implementa *mecanimos*, las políticas se dejan para las aplicaciones.

  Ejemplos: La carga de servicios, la propia carga de módulos, las acciones esperadas tras enchufar un dispositivo, ...

- Si tuvieramos un sistema mínimo funcionando, montar un NFS, configurar un unionfs, etc, ...sería trivial.

- Solución: Usar un initial ram disk, es decir un sistema cargado en memoria para montar el root del sistema definitivo.

- El bootloader carga este inital ram disk junto con el kernel.

  Aunque la mayor parte de las veces este inital ram disk es inecesario y sólo ralentiza el arranque.

Dracut

- Es el futuro de los initial ram disk.
- Diseño modular. Cada módulo define los programas, scripts y/o módulos del kernel que se deben añadir al inital ram disk, y este obtiene las bibliotecas que requiere y las instala reconstruyendo enlaces, directorios, etc...
- Su pilar fundamental es udev, lo cual de una manera muy elegante aporta paralelismo.
  - Pedimos a udev que inicialize los dispositivos.
  - Se generan una cascada de eventos que a su vez pueden inicializar otros dispositivos.
    Podemos cargar el subsistema usb, encontrar un disco duro, activar el soporte usb mass storage, etc...
  - Se da un timeout y udev se queda procesando los eventos hasta que se encuentra un dispositivo capaz de cargar el sistema

¿Por qué es interesante Dracut?

- Es el único inital ram disk que permite montar el sistema con NFS4.
- El modo live apenas
- Por que hemos desarrollando nosotros un módulo para arrancar por WiFi.

Diseccionado KestrelCluster

KestrelHPC 1.0

- Desarrollado por Denis Sanchez de Argoitia
- Alumno de la FISS[6] como parte de su Proyecto de Final Carrera en el CEIT[7].

[6] Facultad de Informática de San Sebastian
[7] Centro de estudios e investigaciones técnicas de Gipuzkoa

KestrelHPC 1.0

Objetivos :

- Nodos live. Imprescindible no instalar nada en los nodos.
- Facilitar la ejecución de aplicaciones MPI manteniendo una lista de los nodos.
- Que sea *instalable* en una maquina.

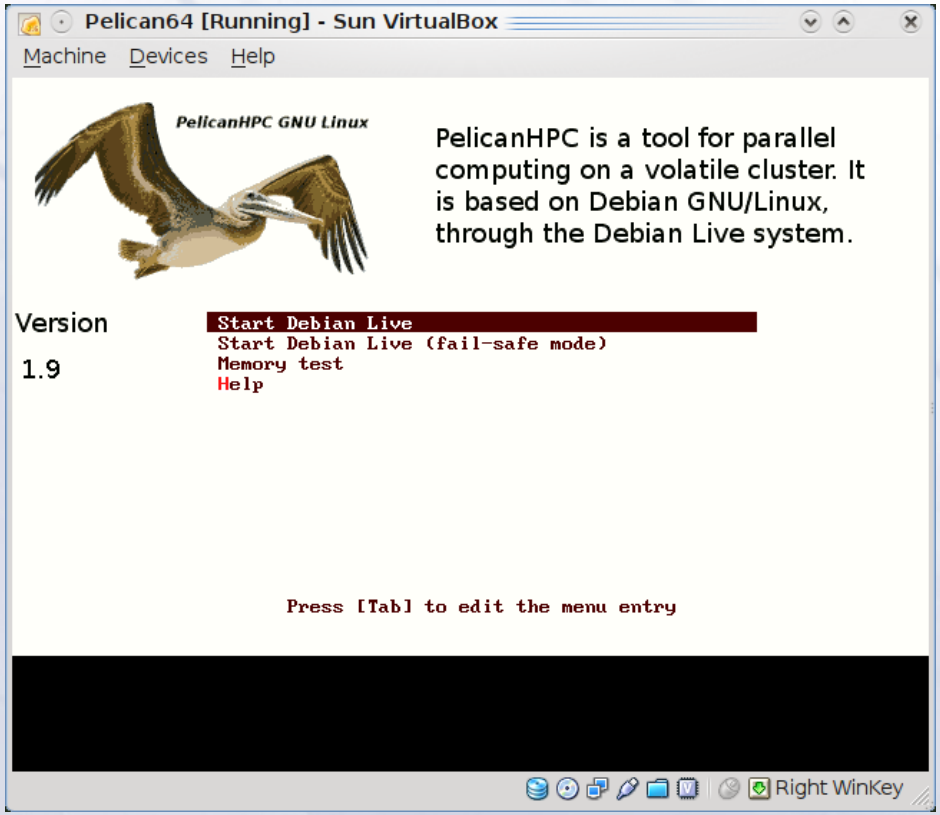
Solución :

- Basado en una distribución Debian.
- Adaptar[8] PelicanHPC para instalar sus scripts en un sistema Debian.

[8] Un fork

PelicanHPC

- Desarrollado por Michael Creel
- Profesor de econometrics en la Universitat Autònoma de Barcelona.



http://pareto.uab.es/mcreel/PelicanHPC/Tutorial/PelicanTutorial.html

Features

- /pelican_config file to allow for persistence, customization and headless boot.
- autodetection of persistent frontend home
- autodetection of frontend and node local scratch space
- ability to run local scripts post boot and setup
- node beep after boot
- firewall
- automated node booting using wake-on-lan
- configuration of slots and optional frontend inclusion for mpi
- ganglia
- static IP assignment configurable using MAC addresses.
- node startup/shutdown script
- possibility to serve DHCP to machines that are not compute nodes

KestrelHPC 2.0

Reescrito desde cero por Jon Ander Hernández

- Prácticas no remuneradas en empresa en el CEIT
- Comunidad (Octubre 2010 - Diciembre 2011)

GitStats - KestrelHPC

Total Files	184
Total Lines of Code	9622 (21591 added, 11969 removed)
Total Commits	285 (average 5.4 commits per active day, 0.9 per all days)
Authors	1 (average 285.0 commits per author[9])
Repositorio	`https://github.com/jonanh/KestrelHPC`

[9] El sistema de plugins de la RPC fue contribuido por Ander Martinez y comiteado por mi

Objetivos :

- Un cluster es algo sencillo.

  ¿Por que no describir todos los pases como una colección de recetas y así facilitar su configuración?

- Modular.

  Un cluster no tiene sólo porque ser HPC con MPI

- Seguro
  - Trabajamos con imágenes y el home compartidos por nfs, lo cual implica que nada de lo exportado debe comprometer nuestro ordenador
  - Hay que comprobar que el nodo que ha arrancado es realmente un nodo, y no otro equipo sospechoso o el mismo equipo arrancado con otro sistema operativo.

- Múltiples imágenes para los nodos.

  Cada departamento puede crear su propio sistema instalando el software que necesitan directamente desde Debian

- Facilitar el registro de nodos. Arrancamos los ordenadores y se registran automáticamente.

Objetivos técnicos:

- KISS. No añadir mecanismos inecesarios. No necesitamos usar bases de datos para gestionar ni los nodos que tenemos, ni para gestionar los usuarios. Usamos el registro del servicio dhcp para saber que nodos están registrados, el fichero /etc/hosts para saber que nodos están conectados, y usamos usuarios del sistema con grupos para gestionar los permisos.
- Cualquier módulo debe poderse reemplazar facilmente. Añadimos un componente con el mismo nombre en /etc/kestrel/<fichero>
- Separamos los scripts en scripts de configuración y scripts de instalación.
- Usamos una rpc para recibir eventos de los nodos que convertimos en llamadas a scripts de eventos.
  - Registro de un nodo. Añadir al fichero del servidor de dhcp
  - Conexión de un nodo. Añadir el nombre de la maquina a /etc/hosts

KestrelHPC 3.0

Objetivos :

- Fácilitar la comprensión de lo que se modifica en el sistema o en los nodos. Podemos expresar "flags" en los propios nombres de los ficheros
- Añadir un control de versiones de los ficheros modificados.
  Cuando desactivamos Kestrel, queremos que nuestro sistema vuelva estar exactamente como lo estaba antes.
- Poder activar y desactivar Kestrel con facilidad.
- Soporte Linux Containers (LXC) para modificar las imágenes.
  Así cualquiera puede acceder a una imagen, modificarla, instalar software, etc... sin miedo a que esa persona pueda evadir el confinamiento y acceder a la maquina.
- Muchas, muchas mejoras menores:
  - Mayor robustez en la carga del demonio y el RPC
  - bash completion Por ejemplo, nos autocompleta las diferentes fechas del fichero de log, para poder consultar un log concreto
  - sistema de plugins en el RPC
  - Posibilidad de desactivar conjuntos de templates/Scripts usando un label[10].

[10] También podemos desactivar una funcionalidad para una imagen en concreto

Comienzo del taller :-)

¿Preguntas? ¿dudas antes de seguir?

¡Gracias a todos por venir! ;-)

Referencias

- I ❤ http://en.wikipedia.org 0:-)

- “KestrelHPC”

  http://kestrelhpc.sourceforge.net/