Kerrighed / XtreemOS cluster flavour

Jean Parpaillon

Reisensburg Castle – Günzburg, Germany
July 5-9, 2010

Summary

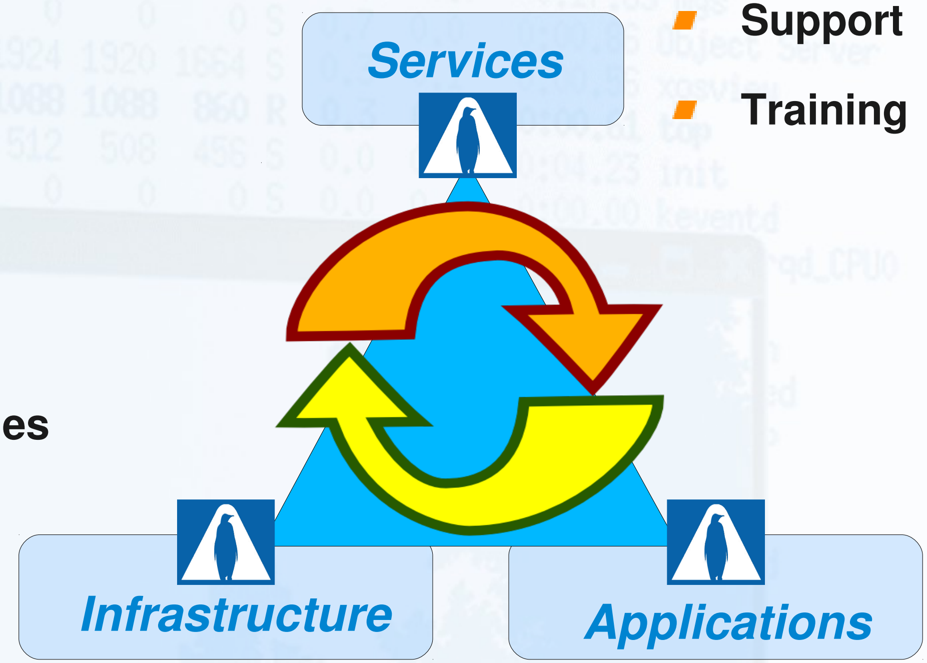
- **Kerlabs**
- **Context**
- **Kerrighed**
  - Project
  - Features
  - Deployment

Kerlabs – Who we are

- **KERLABS, a spin-off from INRIA**
  - Kerrighed technology industrialization
  - Kerrighed: a Single System Image at kernel level
  - 3 PhD thesis
- **Founded in 2006**
  - Based in Rennes, France
  - 8 people team
- **Our skills**
  - Distributed Operating System
  - Parallel architecture and programming
  - High-end interconnection technologies

Kerlabs – Our services

- Expertise
- Hard, Soft, Network Integration
- Deployment
- Support
- Training

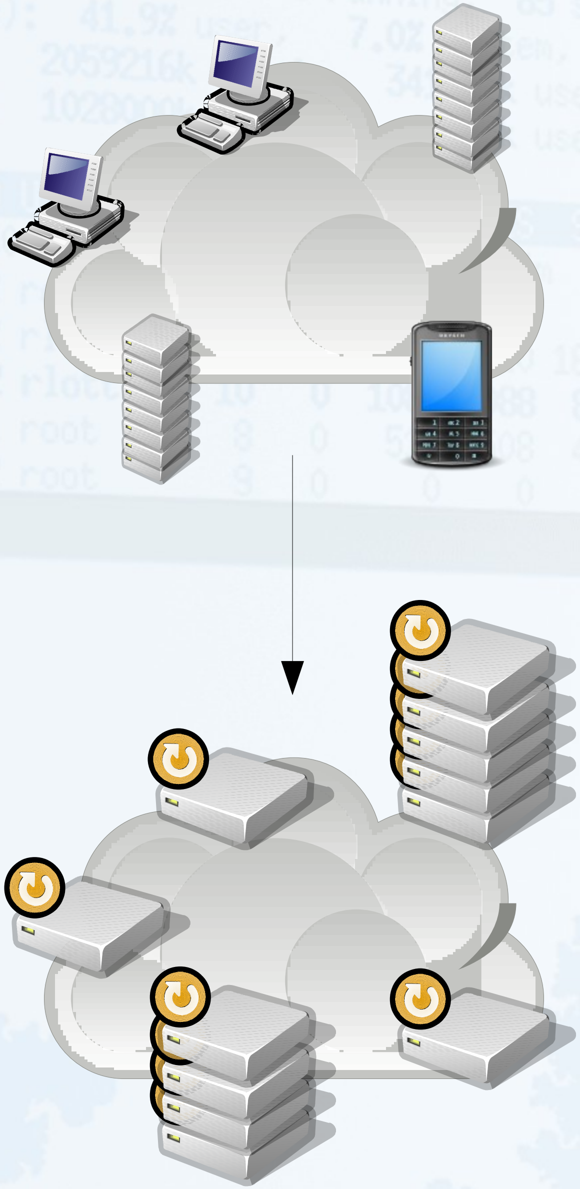


- +200 computing nodes
- Storage servers
- Heterogeneous interconnection
  - GbE, 10GbE, Infiniband, …
- Virtual SMP : resources aggregation

- Tests and validation
- Parallel development assistance

Context

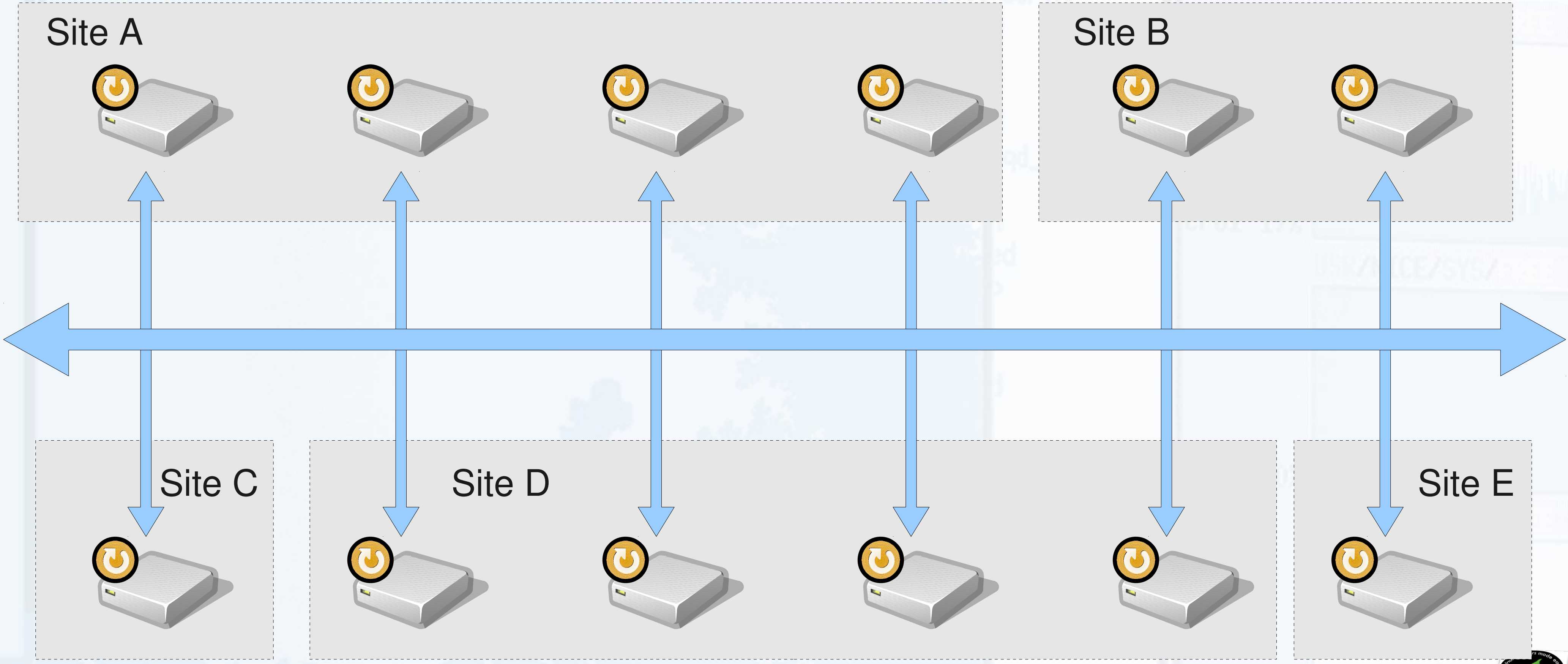
Grid outlines



- **Provides services on top of distant sites**
- **Hiding heterogeneity of**
  - Architecture
  - Computing power
  - Memory amount
  - Topology

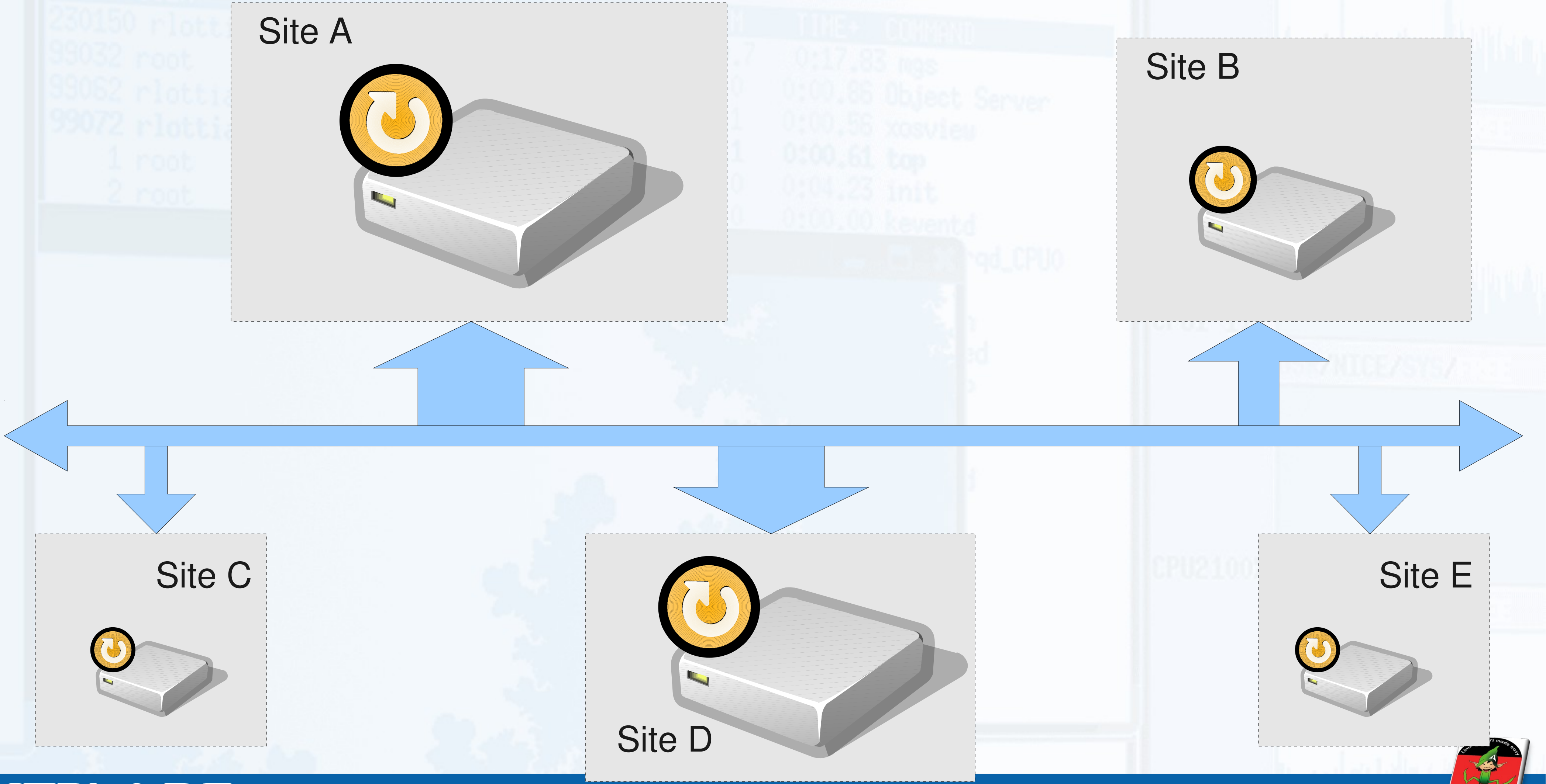
Let's zoom in !

- Are all resources equally distant ?
- Can we leverage resources locality ?



Let's zoom in !

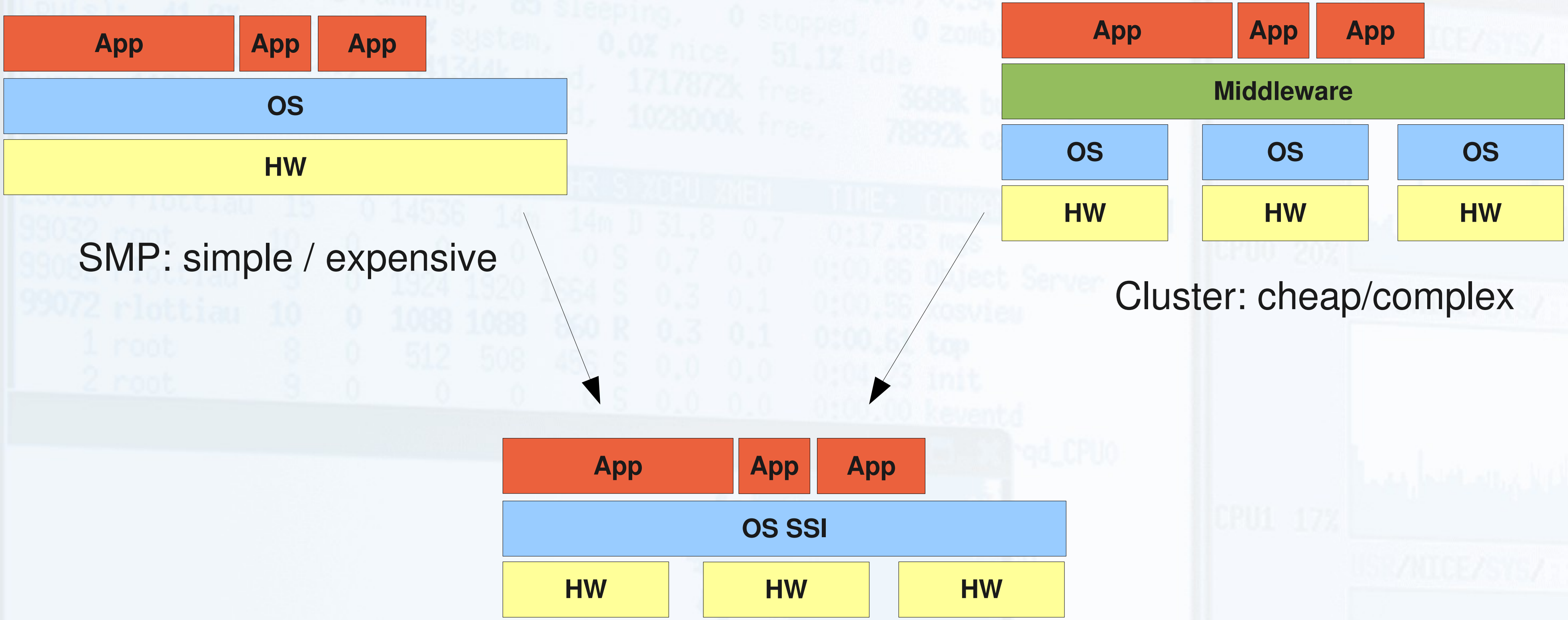
- **Let's optimize the use of cluster on a given site**



Cluster properties

- **A lot of (as far as possible) homogeneous nodes**
- **Complex for user:**
  - Balancing CPU load
  - Managing memory split between nodes
  - Managing network
- **Complex for admin:**
  - Optimizing node use through job scheduler policies
  - Managing users rights, datas, etc. across nodes

Single System Image



App App App
OS
HW

SMP: simple / expensive

App App App
Middleware
OS OS OS
HW HW HW

Cluster: cheap/complex

App App App
OS SSI
HW HW HW

SSI (Virtual SMP): simple/cheap

- **Resources abstraction**
  - CPU, Memory, Filesystem, Network, etc.
- **An Operating System for clusters**

Kerrighed
- The project -

Kerrighed : the project (1)

- **Initiated in PARIS team, IRISA, France in 1999**
  - Directed by Christine Morin
  - Collaboration between INRIA, EDF and Université Rennes 1
  - 3 PhD. thesis
  - Several engineer contracts
  - 30 year * men research
- **2006 : an open source project**
  - Kerlabs foundation in 2005, INRIA spin-off
  - External contributions
  - Website, mailing list, bug tracker, *etc.*
  - Partnership with XtreemOS European Project

Kerrighed : the project (2)

- **From research to industry**
- **Public source repository**
- **SVN (until 2.4.x) :** `svn://scm.gforge.inria.fr/svn/kerrighed/trunk`
  - Git (from 3.0) : `git://git-externe.kerlabs.com`
    - Mirror : `http://mirrors.git.kernel.org`
- **Deployment integration**
  - Standard compilation tools (`autotools`)
  - Debian packages, Mandriva
  - OSCAR (not maintained)
  - LiveCD
- **About 700 regression tests**
  - LTP + Kerrighed specific tests

Kerrighed
- Features -

Kerrighed in a nutshell

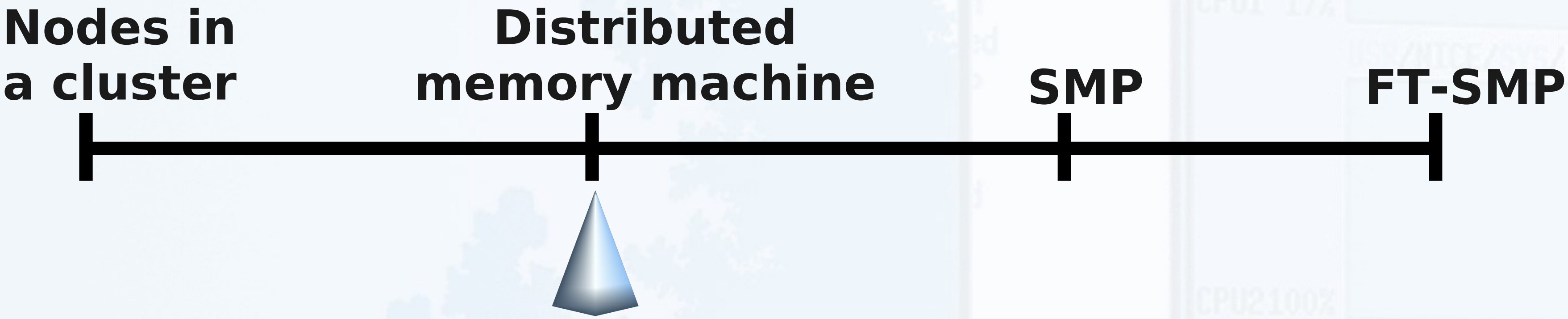
- **Single System Image Operating System**
- **Standard**
  - Extends Linux kernel
- **Dynamic**
  - Transparent load balancing
- **Elastic**
  - Node addition/removal
- **Adaptable**
  - Fully configurable (with default policies)
- **Reliable**
  - Checkpoint/restart

Adaptable

- **All features are not required for all applications**
  - Checkpointing...
  - Migration...
- **Some features are even not wanted**
  - Checkpointing short application
  - Distribute highly communicating processes

Capabilities : on-demand SSI

- To control what we want to do
- To control what is allowed to do (basic)



Nodes in a cluster

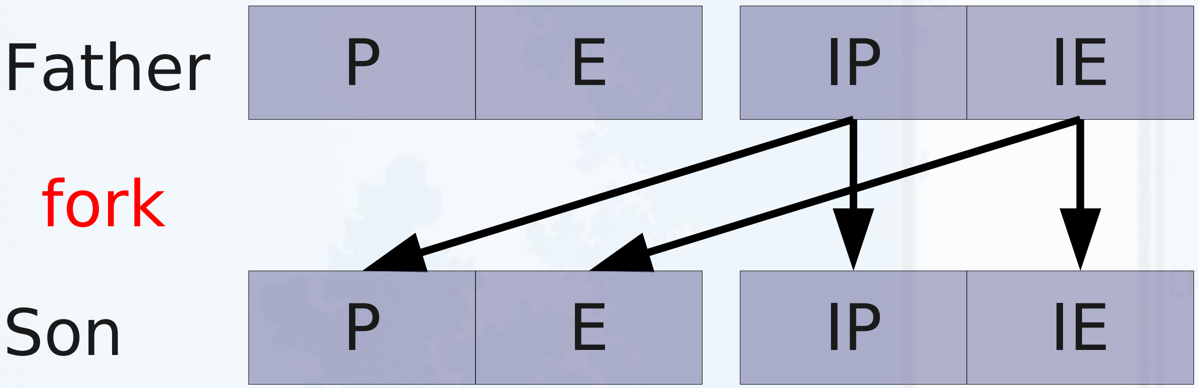
Distributed memory machine

SMP

FT-SMP

Capabilities : inheritance mechanism

- **Each process has 4 sets of capabilities**
  - Own *permitted* (P): capabilities it is allowed to use
  - Own *effective* (E): capabilities it is using
  - Default *permitted* for its sons (IP)
  - Default *effective* for its sons (IE)

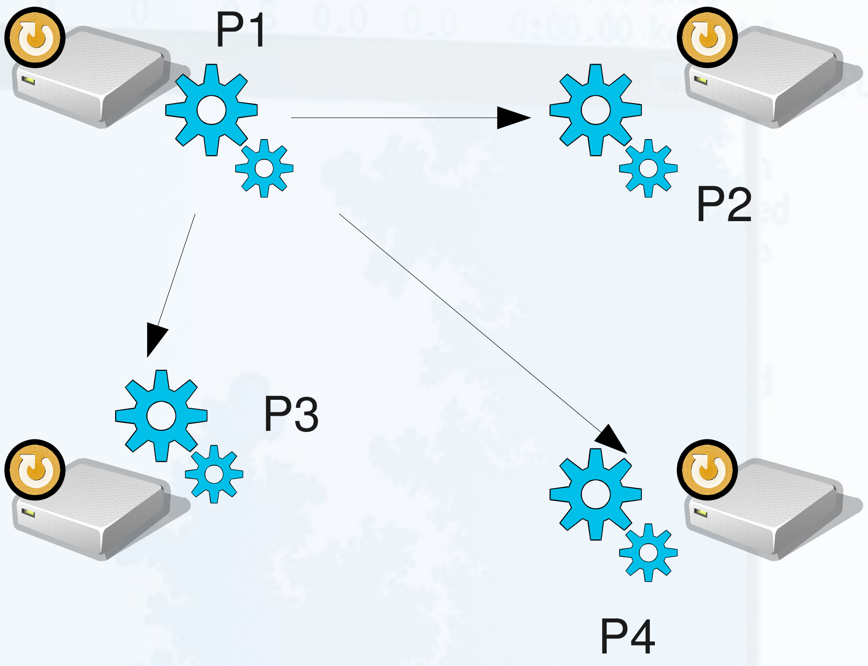


Interface

- **Command line interface to modify capabilities :**
  - krgcapset [pid] <options>
- **Options**
  - `--show` : list process capabilities
  - `-e | --effective` : set *effective* capabilities
  - `-p | --permitted` : set *permitted* capabilities
  - `-d | --inheritable-effective` : set sons' *effective* capabilities
  - `-i | --inheritable-permitted` : set sons' *permitted* capabilities

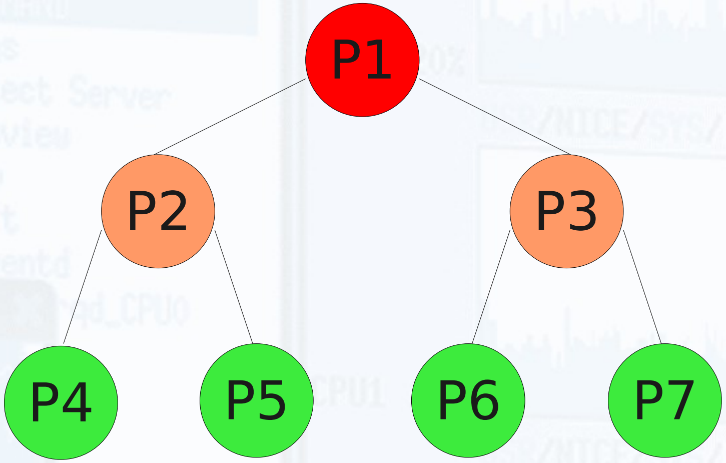
Feature: static balancing

- **Allow application to fork on distant node**
- **Capability: DISTANT_FORK**
- **Well suited for short applications**



Feature: static balancing

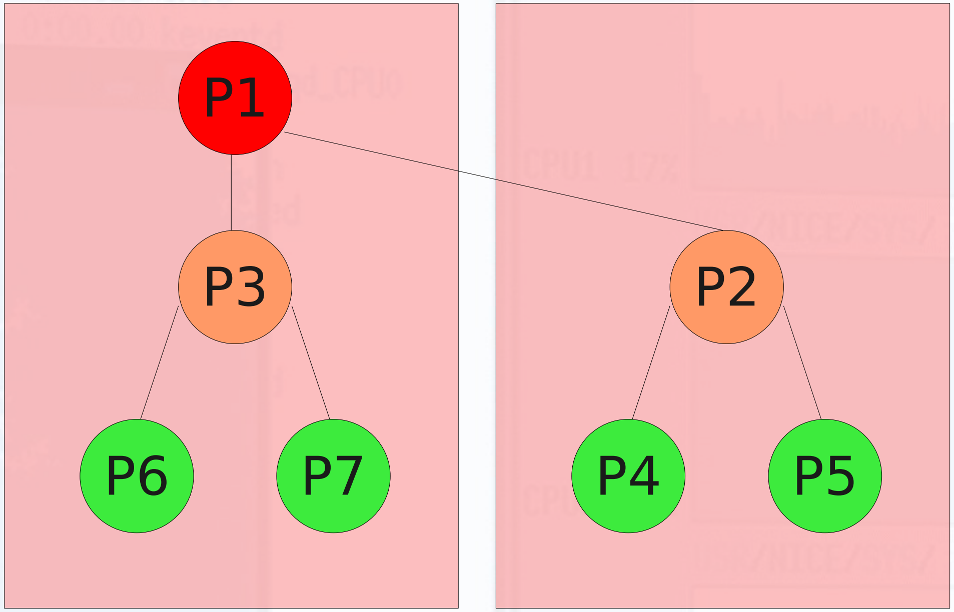
```
void main(void)
{
    int nb_sons = 0;
    int depth = 0;

    while (nb_sons != 2 && depth < 2) {
        if (fork())
            nb_sons++;
        else {
            depth++;
            nb_sons = 0;
        }
    }
}
```



Feature: static balancing

```
void main(void)
{
    int nb_sons = 0;
    int depth = 0;

    while (nb_sons != 2 && depth < 2) {
        if (fork())
            nb_sons++;
        else {
            depth++;
            nb_sons = 0;
        }
     }
}
```

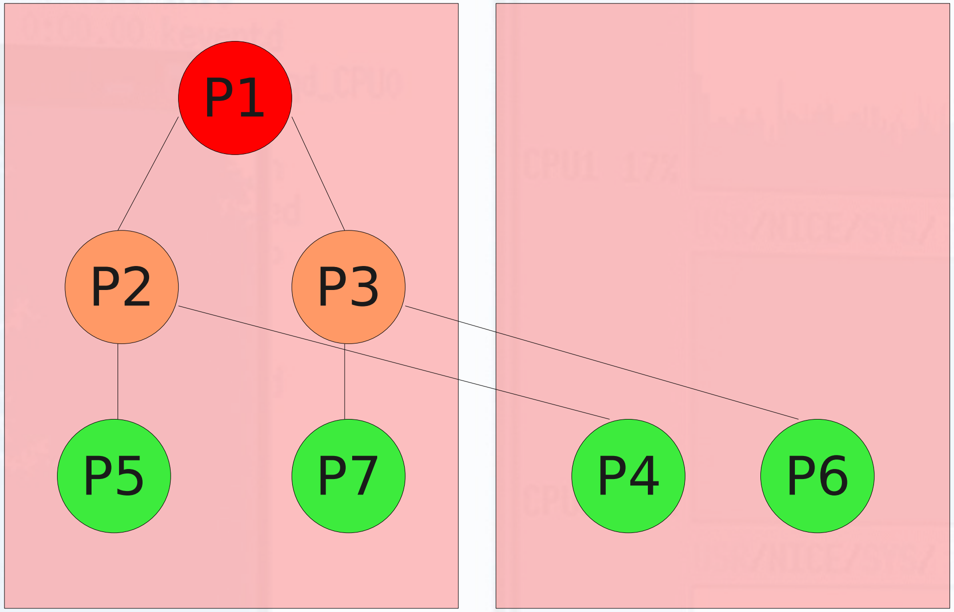
```
paraski33% krg_capset -e +DISTANT_FORK
paraski33% ./fork-test
```



Feature: static balancing

```
void main(void)
{
    int nb_sons = 0;
    int depth = 0;

    while (nb_sons != 2 && depth < 2) {
        if (fork())
            nb_sons++;
        else {
            depth++;
            nb_sons = 0;
        }
     }
}
```

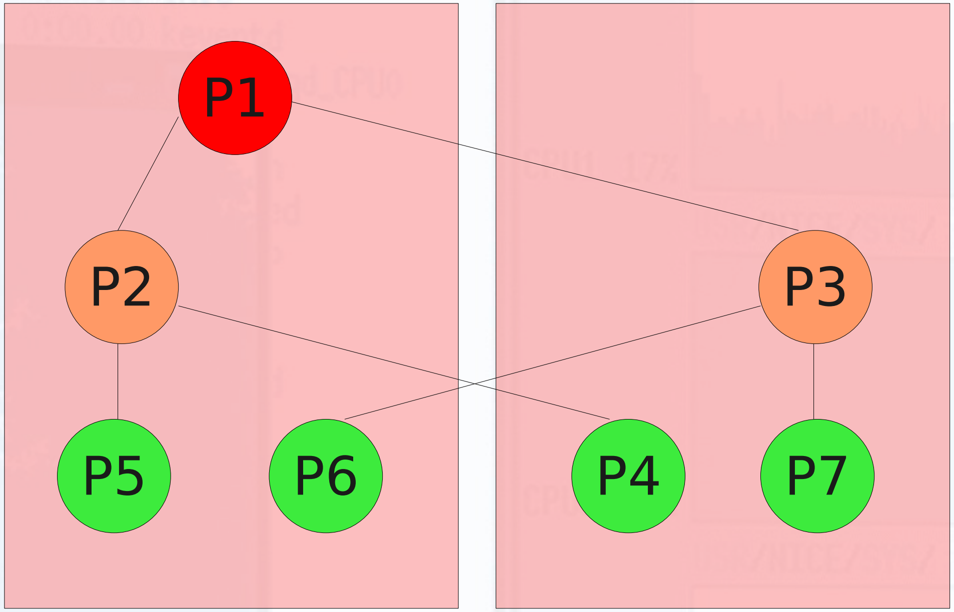
```
paraski33% krg_capset -d +DISTANT_FORK
paraski33% ./fork-test
```



Feature: static balancing

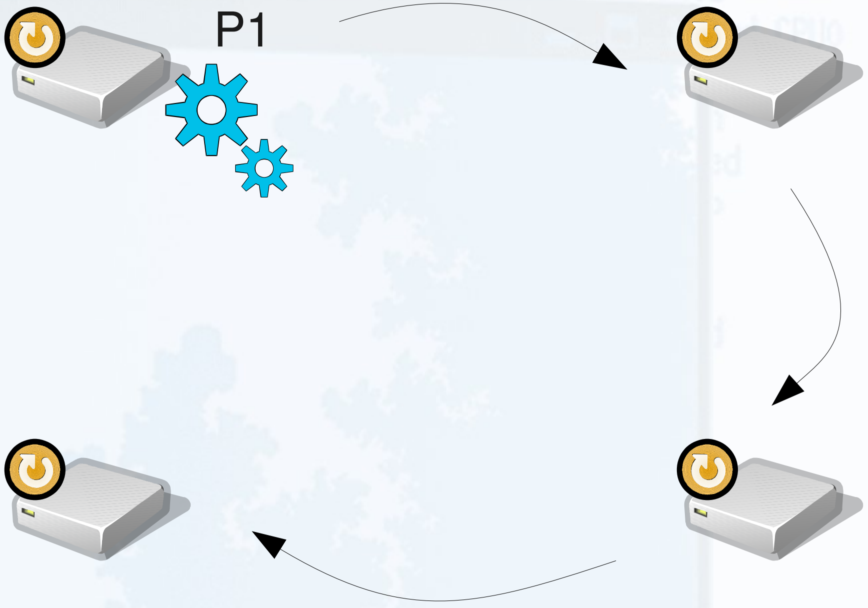
```
void main(void)
{
    int nb_sons = 0;
    int depth = 0;

    while (nb_sons != 2 && depth < 2) {
        if (fork())
            nb_sons++;
        else {
            depth++;
            nb_sons = 0;
        }
     }
}
```

```
paraski33% krg_capset -e +DISTANT_FORK
paraski33% krg_capset -d +DISTANT_FORK
paraski33% ./fork-test
```



Feature: process migration (1)

- Dynamic load balancing
- Capability: CAN_MIGRATE
- Adapted to long applications

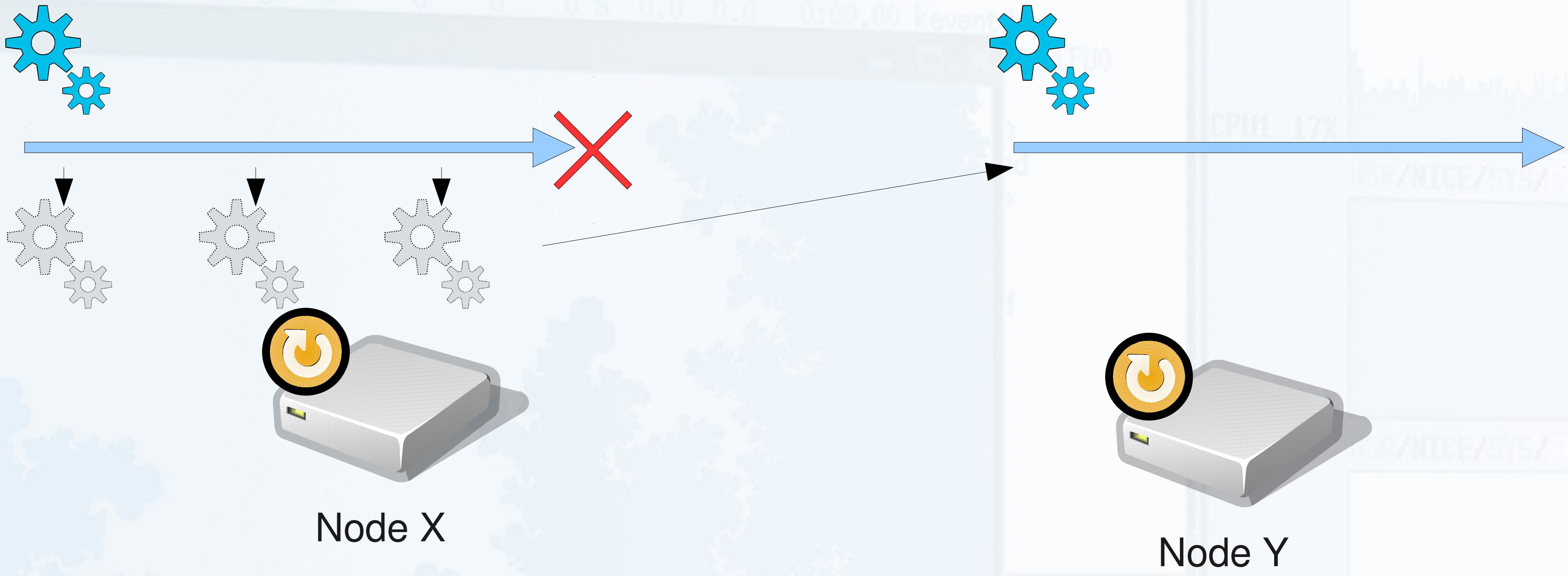


Feature: process migration (2)

- **Automatic : global (customizable) scheduler**
- **Manual : `migrate <pid> <nodeid>`**
- **Cost:**
  - Memory: lazy migration
  - Regular files: distributed FS
  - Special files (devices, etc.): File Access Forwarder server
- **Adapted to long applications**

Feature: checkpoint/restart (1)

- **Checkpoint: save status of an application**
- **Restart: restart an application from checkpoint**
  - On the same nodes
  - On different nodes



Feature: checkpoint/restart (2)

- **Goals**
  - Fault tolerance
  - Hardware maintenance
  - Debugging: restart a dead application with a debugger
  - Scheduling: stop an application to free CPU AND memory (unlike SIGSTOP)

Feature: checkpoint/restart (3)

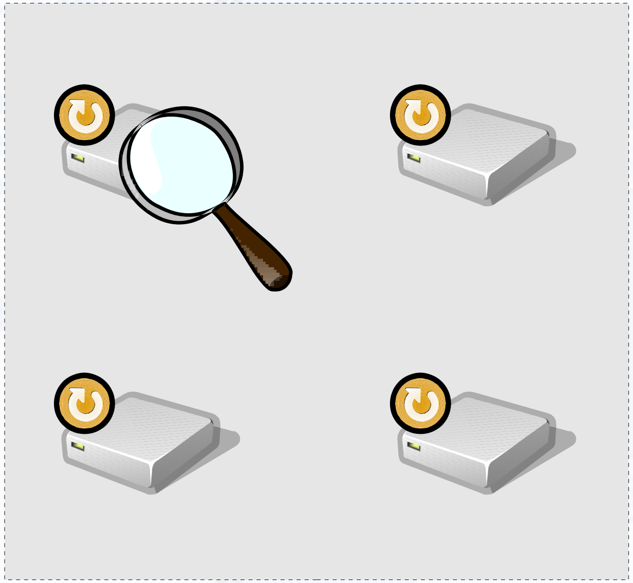
- **Application status**
  - Memory, registers
  - Files: relies on 3rd party tools (cp, filesystems, etc.)
- **Application limits**
  - Set of process
  - With a set of files
  - Communicating
    - pipes, sockets, IPC objects
  - Limits evolve through
    - fork(), exit(), open(), close(), mmap(), etc.

Feature: checkpoint/restart (4)

- **Features**
  - Support SysV IPC
  - C/R of distributed applications: Kerrighed, OpenMPI
- **Ongoing features**
  - Incremental checkpointing
  - Callbacks: inform application of checkpoint/restart
- **Usage**
  - **Start** application with `krgcr-run(1)`
  - **Freeze**, **checkpoint** with `checkpoint(1)`, `ipccheckpoint(1)`
  - **Restart, unfreeze** with `restart(1)`, `ipcrestart(1)`

Feature: local resources view

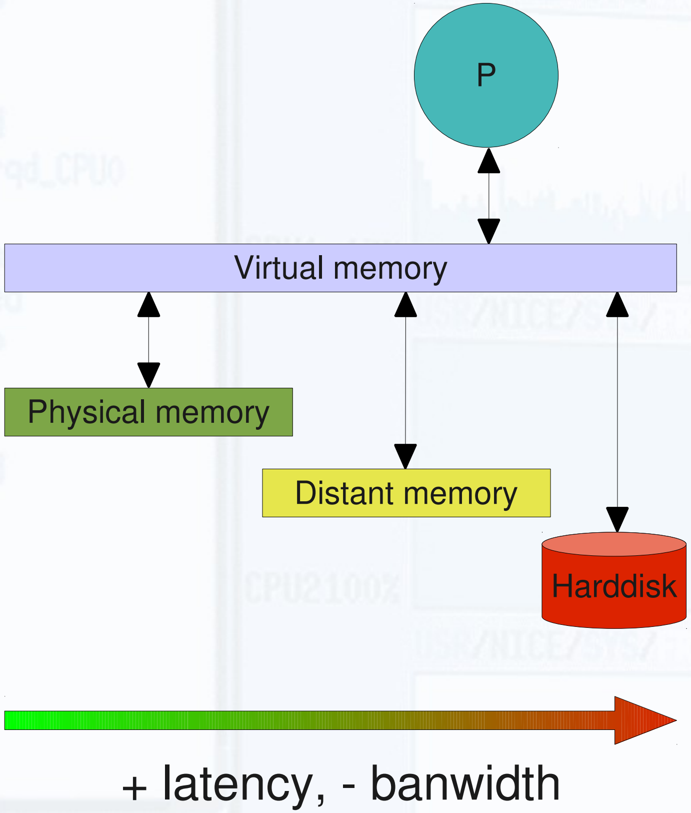
- **Capability: SEE_LOCAL_PROC_STAT**
- **Default: cluster-wide /proc**
  - CPU usage, memory, PIDs, *etc.*
- **Allows to see local resources**
  - Monitoring
  - Isolation
  - ...



Feature: memory aggregation

- **Capability: USE_REMOTE_MEMORY**
- **Use case: a process needs more memory than available on 1 node**
  - Hard-disk swap : sloooooowdown !
- **Idea: add another level of memory**
  - Use distant nodes memory as a swap
- **Experimental from 2.4.x**

	Capacity	Bandwidth	Latency
RAM	2 GB	~5 GB/s	~ 50 ns
Distant mem (GbE)	16 GB	~120 MB/s	~30 µs
Distant mem (IB 4x DDR)	16 GB	~2.5 GB/s	~2 µs
Harddisk drive	80 GB	~5 – 50 MB/s	~ 5 ms

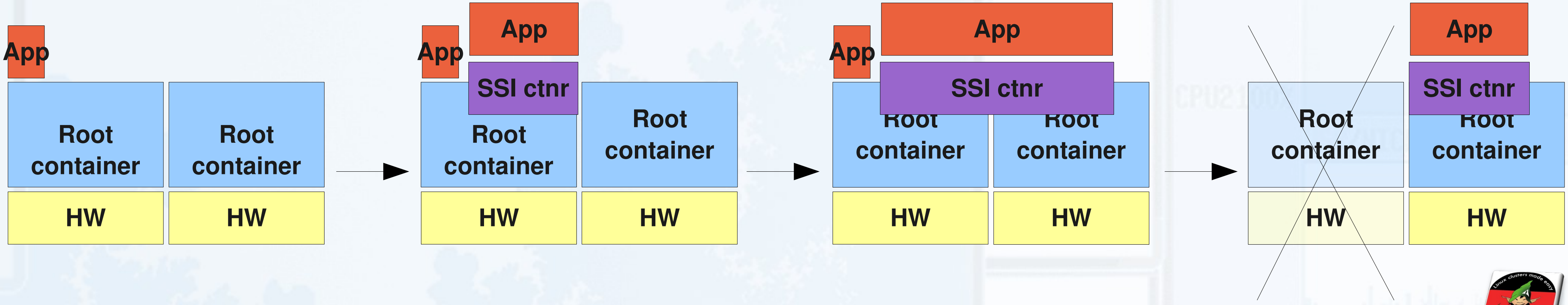


Kerrighed containerization (1)

- **Kerrighed issues**
  - Integration: processes exists before cluster start and after cluster shutdown
  - Checkpoint/restart: what if pid exists when restarting ?
  - QoS: want to control resources attribution
- **Idea: isolation/virtualization**
- **Linux containers**
  - Lightweight
  - Hierarchical
  - Highly configurable: PID, hostname, mount points, IPC, network, user (todo)

Kerrighed containerization (2)

- **Kerrighed implementation**
  - Cluster resources in a container
  - Allow node addition in a live cluster
- **Ongoing features**
  - Nodes removal
  - QoS
  - Containers in a Kerrighed container



Customizable scheduler (1)

- **Goal: balance load on the cluster**
  - Process migration
- **Many parameters**
  - Classical: CPU load, memory load
  - Others: network, temperature, tokens, *etc.*, *etc.*
- **Many policies**
  - Preemption, etc.
- **Customizable scheduler**
  - Through configfs
  - Design schedulers with probes, sink, filters, policies, *etc.*
  - Apply to group of processes

Customizable scheduler (2)

- **Examples**
  - Scheduling tokens
    - On each node a USB key contains tokens
    - Tokens use is heterogeneous
    - Goal:
      - Balance token use
      - Warn admin when tokens stock is low
  - e-mailing campaign scheduling
    - Each e-mail campaign has its own priority
    - Goal:
      - Balance e-mail campaign sending

Deployment (1)

- **Requirements**
  - Nodes must share
    - Kerrighed container filesystem
    - Kernel (obviously)
    - Arch
  - Nodes can have different
    - CPUs, memory, network, devices, etc.

Deployment (2)

- **Installation guide:** `http://kerrighed.org/docs/releases/3.0/INSTALL`
- **Typical installation outlines**
  - **Install a distribution in a chroot:** debootstrap, etc.
  - **Install Kerrighed in this chroot:** follow installation guide
  - **Share the chroot with NFS:** exportfs
  - **Serve the kernel (and initrd, eventually) through TFTP**
  - **Setup bootloader:** *e.g.* pxelinux
    - append ro root=/dev/ram ip=dhcp nfsroot=192.168.122.1:/srv/chroot/kerrighed session_id=1 node_id=1
- **Kerrighed container is accessible through ssh on port 2222**

Documentation

- **Installation guide**
  - http://kerrighed.org/docs/releases/3.0/INSTALL
  - INSTALL file in tarball
- **Online man pages: kerrighed(7)**
  - krgcapset(1), krgcapset(2), kerrighed_capabilities(7)
  - checkpoint(1), restart(1), ipccheckpoint(1), ipcrestart(1)
  - migrate(1), migrate(2)

Other Resources

- **Kerrighed**
  - Website : http://kerrighed.org
  - Mailing lists : kerrighed.dev@listes.irisa.fr, kerrighed.users@listes.irisa.fr
  - IRC : #kerrighed@irc.freenode.net
- **XtreemOS : http://xtreemos.eu**
- **OSCAR : http://oscar.openclustergroup.org**

Nous contacter...

KERLABS

www.kerlabs.com

contact@kerlabs.com

KERLABS

80, avenue des buttes de coësmes
35000 RENNES - FRANCE

Tél : +33 6 81 97 23 97