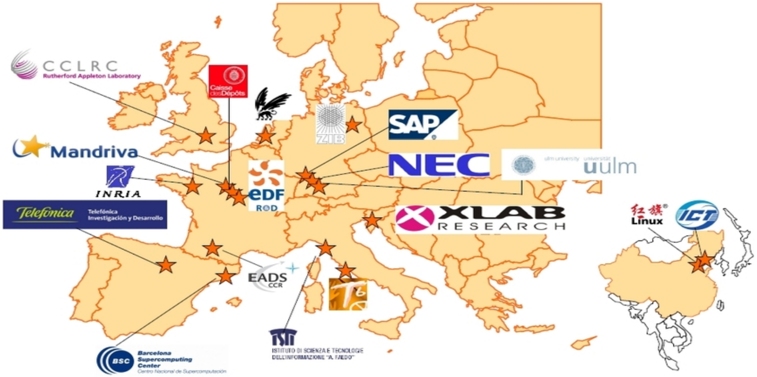
XtreemOS

Surbhi Chitre

IRISA, Rennes, France

July 7, 2009


CCLRC
Rutherford Appleton Laboratory
Caisse des Dépôts
Mandriva
INRIA
Telefónica
Telefónica Investigación y Desarrollo
ZIB
SAP
NEC
uulm
eDF R&D
XLAB RESEARCH
红旗 Linux
ICT
EADS CCR
ISTI
ISTITUTO DI SCIENZA E TECNOLOGIE DELL'INFORMAZIONE "A. FAEDO"
BSC
Barcelona Supercomputing Center
Centro Nacional de Supercomputación

XtreemOS
Enabling Linux for the Grid

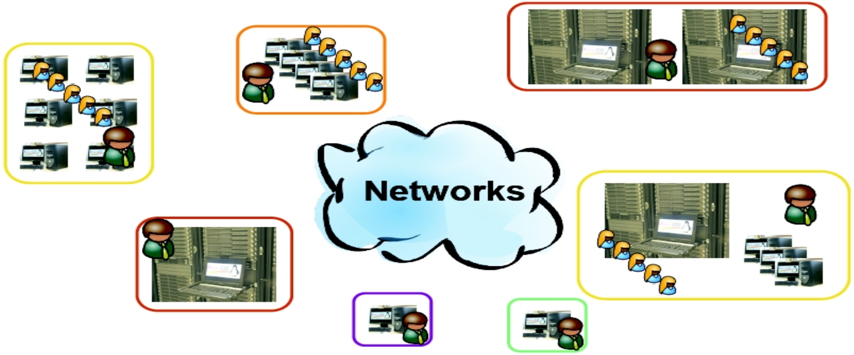
Outline

- What is XtreemOS
- What features does it provide
- Job Management in XtreemOS
- How is it new and different
- Conclusion

Discussion Path

Discussion Path

Grids

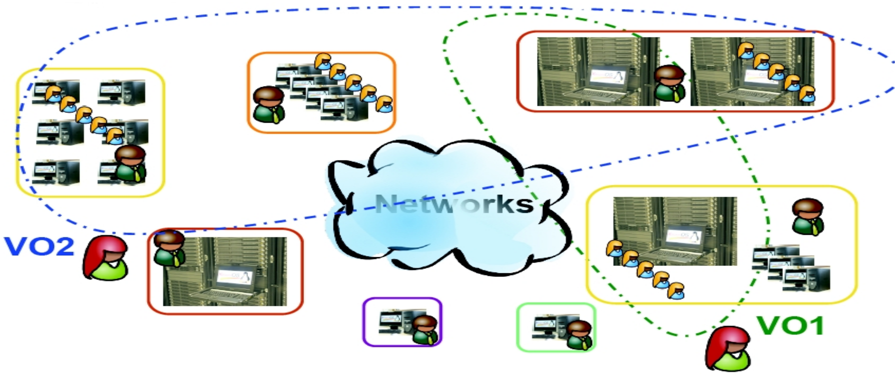


- Linux based Operating System for the next generation grids
- Installation CD - mobile, PC, cluster flavor
- Distributed Resource Abstraction
- Secure Resource Sharing
- Scalable - encorporates millions of nodes, users
- High Availability - replicated services
- Legacy applications executed
- Execute applications like ./application
- Job Monitoring, isolation, fault tolerance provided

Types of Actors

1. User
   - Offload huge computations to grid
   - Security
   - Monitoring
2. Administrator or Owner of Resource
   - Non trusted users should not be allowed
   - Node should not be attacked
3. Application Developer
   - Easy to develop applications with no modification

VO Management



VO Management

Requirements

- VOs have lifespan
- Resource sharing on demand
- Users, Resources - freely join / leave VO, members of multiple VO
- VO user account different from local account
- VO-level Policy - can a user access a VO resource
- Node Policy - can a VO user access this resource

XtreemOS
Enabling Linux for the Grid

VO Management

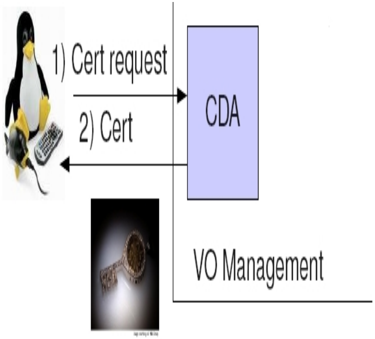
XtreemOS features

- Natively supported
- Confidentiality, Integrity, Authenticity provided
- Manages VO lifecycle
- Manages users, resources VO credentials, distribution
- Enforces VO and node policies upon resource usage
- Authenticate and provide access control to local nodes

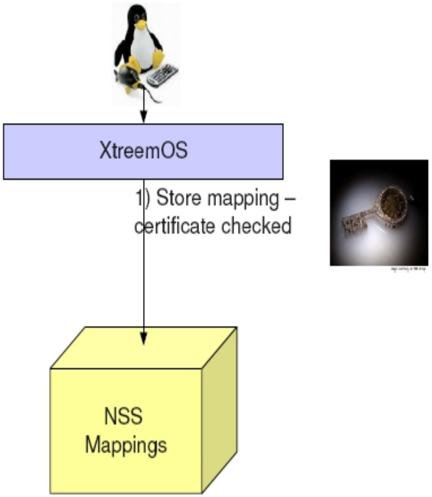
Security

- Single Sign On
- Pluggable Authentication Module
- Name service switch and key retention - Session Management
- Grid/VO users are never local users

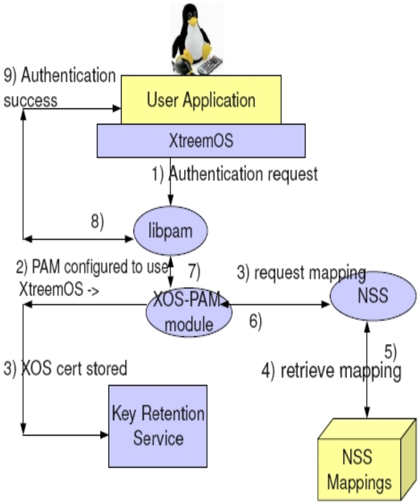
Single Sign On



Single Sign On



Single Sign On



Entities: Users, Resources and Services

- Resources are discovered - advanced p2p techniques
- Services are decentralised and replicated
- User information is stored in a DHT.

Data Management - XtreemFS

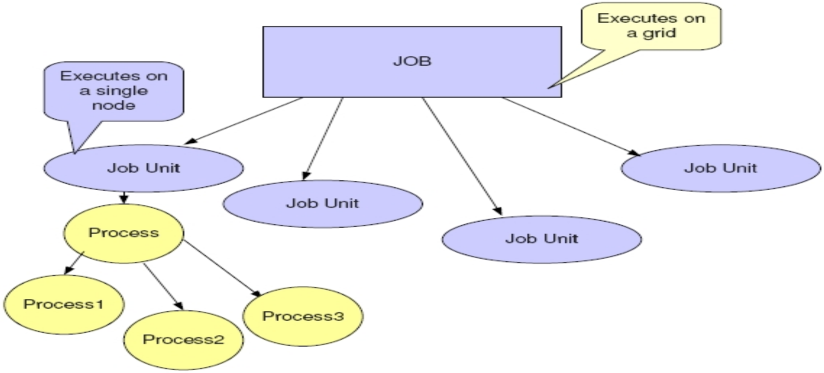
- Distributed, spanning the grid
- Posix like
- Self replicating
- Transactional consistency
- Grid User has a corresponding fs space
- Automatically mounted when a user executes a job

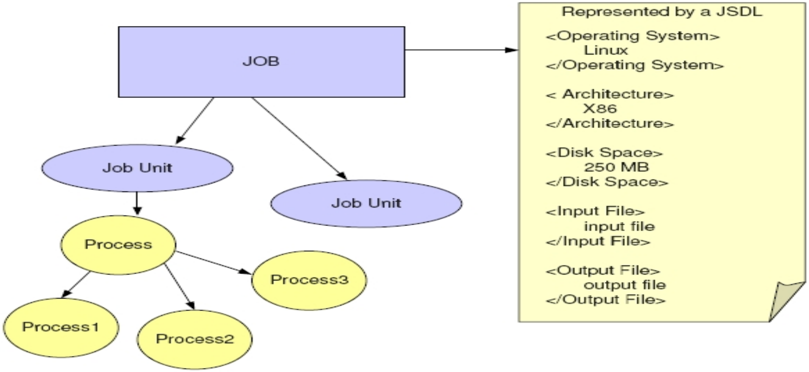
Job Management

- Offload execution of heavy jobs in the outside grids!
- Execute jobs securely
- Have control over your jobs
- When it fails because of some problem outside the job, restart it from a last know point
- Debug a job if it fails from a last know point
- Store the output securely.

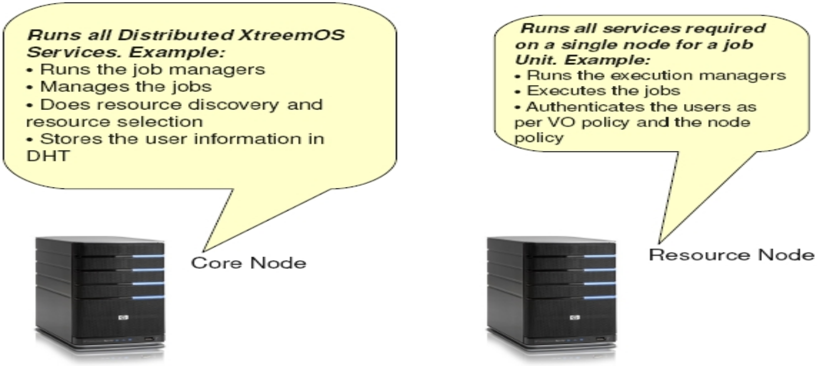
Discussion Path

Job Submission

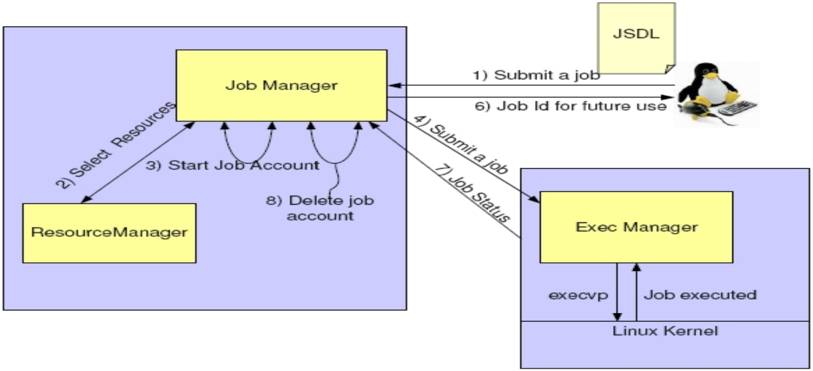



JOB
Job Unit
Job Unit
Process
Process1
Process2
Process3
Represented by a JSDL
<Operating System>
Linux
</Operating System>
< Architecture>
X86
</Architecture>
<Disk Space>
250 MB
</Disk Space>
<Input File>
input file
</Input File>
<Output File>
output file
</Output File>

Job Submission


Runs all Distributed XtreemOS Services. Example:
• Runs the job managers
• Manages the jobs
• Does resource discovery and resource selection
• Stores the user information in DHT
Core Node
Runs all services required on a single node for a job Unit. Example:
• Runs the execution managers
• Executes the jobs
• Authenticates the users as per VO policy and the node policy
Resource Node

Job Submission

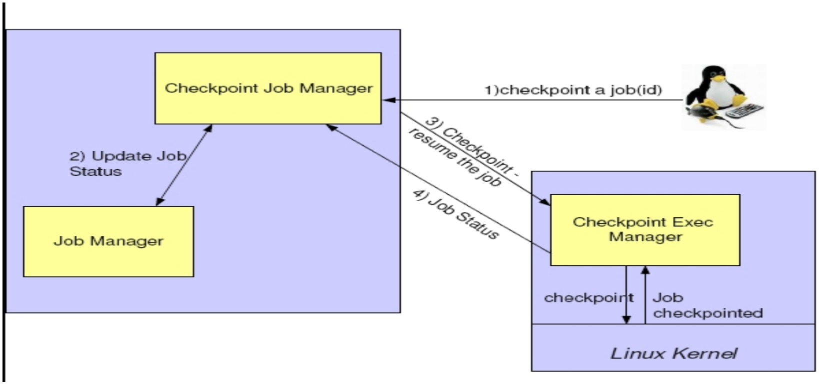


XtreemOS
Enabling Linux for the Grid

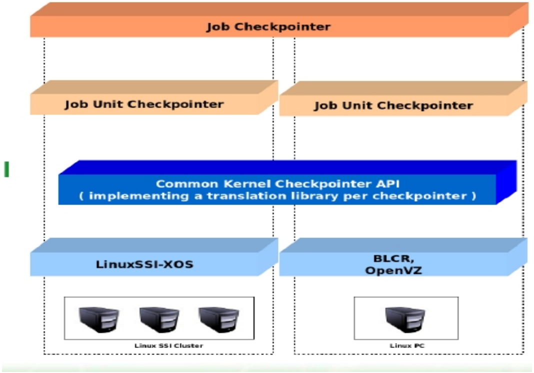
Why checkpoint

- Grid Node stops participation, restart job on some other node
- Debugging
- Recover job from some last known state
- Checkpointing - restart used

Job Submission



Multiple checkpointers



Discussion Path

- Operating system evolution.
- Virtual Private Server
- Containers - root access, separate filesystem, process tree, network stacks, IPC objects and other resources
- Resources - limits and gurantees.
- Containers - checkpointed, restarted, migrated.
- On migration - network connections can be resumed.

How do you get OpenVZ?

- OpenVZ - patch to Linux kernel
- Compile and install the kernel
- Configure grub
- Reboot in this kernel

Separately Download Userspace utilities

- vzctl - managing containers.
- vzpkg - templates
- vzquota - quotas

On the surface - OpenVZ kernel?

- Root container - id always 0
- *All* the processes now start in a root container
- A new container is a child of this container.

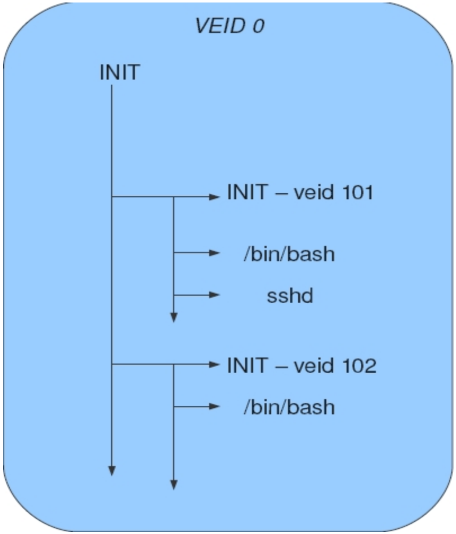
Create a container - Example

- Template - root filesystem and default programs to run in the container
- vzctl - userpace utility for creating, starting, executing, stopping and deleting containers.
- vzctl create 101 –ostemplate <template name>
- vzctl set 101 –ipadd 192.168.10.10 –nameserver 192.168.10.2 –save
- vzctl start
- vzctl set 101 –userpasswd root:test
- vzctl exec 101 /etc/init.d/ssh start
- ssh 192.168.10.1

What happens when you start a container?

- A new VPS created.
- Virtual process id and real process id
- User space applications - see only virtual process id.
- Processes in the container can be seen from outside (ps/pstree)

process hierarchy



Checkpoint and Restart

- vzctl chkpnt 101 –dumpfile dump
- vzctl restore 101 –dumpfile dump
- A process runnning inside a container can be multiprocess, have IPC, have threads, access files etc, it will still be checkpointed.
- As long as the container is restarted immediately, the open network connections can be saved.
- When you restart after a long time, the connection can be lost

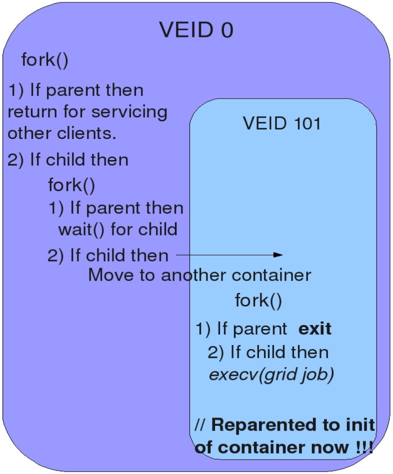
Requirement

- Track a job execution
- Submit a job to a container through the Application execution manager of XtreemOS.
- Checkpoint, restart and migrate a job through the Checkpoint Restart Manager of XtreemOS

Job Submission

- Job should be submitted to a container than to a native kernel.
- Identify that the job needs OpenVZ.
- Done through jsdl tag addition
- A job can contain job units.
- Job units of a same job should be submitted to a same container.
- A new container should be created.

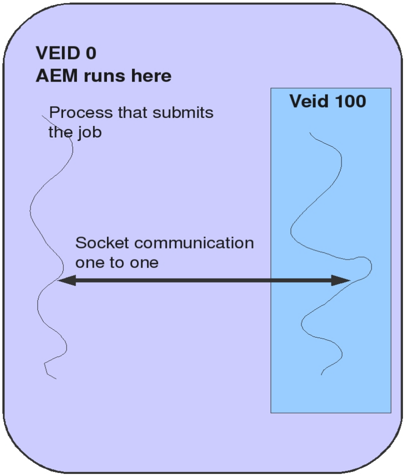
Job Submission - no foreign dependencies


VEID 0
fork()
1) If parent then return for servicing other clients.
2) If child then
fork()
1) If parent then wait() for child
2) If child then
Move to another container
VEID 101
fork()
1) If parent **exit**
2) If child then
execv(grid job)
// **Reparented to init of container now !!!**

Job Submission

- Containers can be accessed using root access only!
- XtreemFS dir should be accessed by the corresponding user.
- Job submission with no foreign process dependency - does not allow checkpointing.
- loader application shall launch a grid job.
- loader shall setgid and setuid appropriately.
- loader application shall help in job tracking
- Stage the job executables and the dependencies to the container

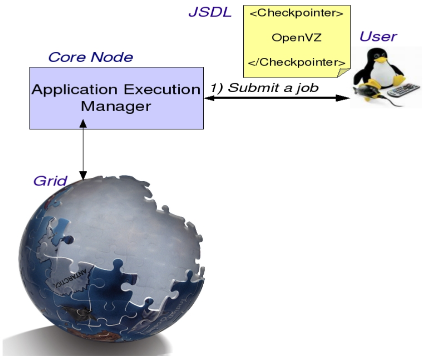
Simple solution - socket communication



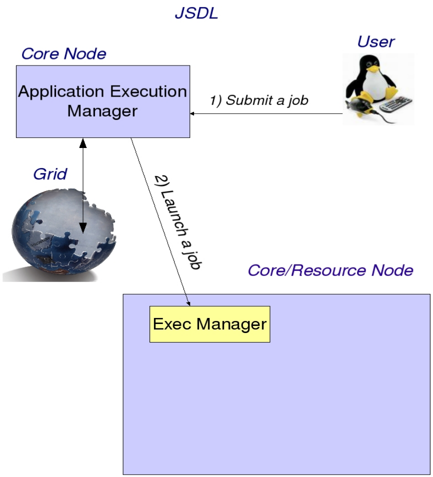
Job tracking

- Cleanup needed after the job has finished.
- Eg: XtreemFS unmount, container cleanup etc.
- No wait() or waitpid() from a process in the root container.
- Container cannot suspend/die automatically when the job has terminated.
- Cannot use kernel connectors.
- server process should exit.
- container should be suspended for future restart
- Job is now alive as long as the container is alive and running - checkpointing based on this.

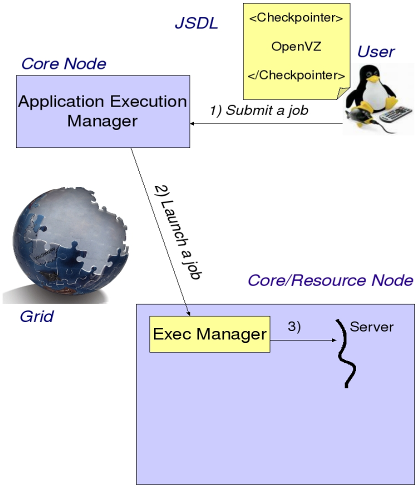
Job submission



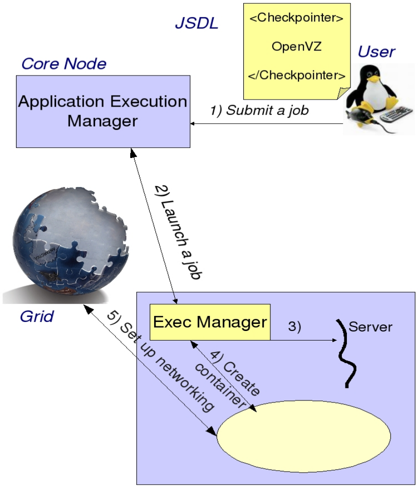
Job submission



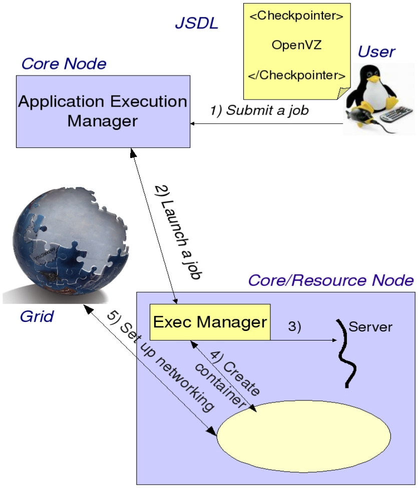
Job submission



Job submission

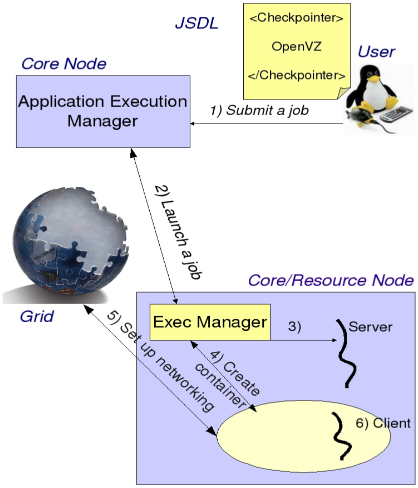


Job submission

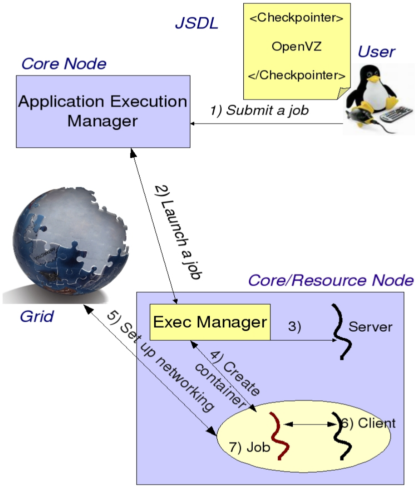


XtreemOS
Enabling Linux for the Grid

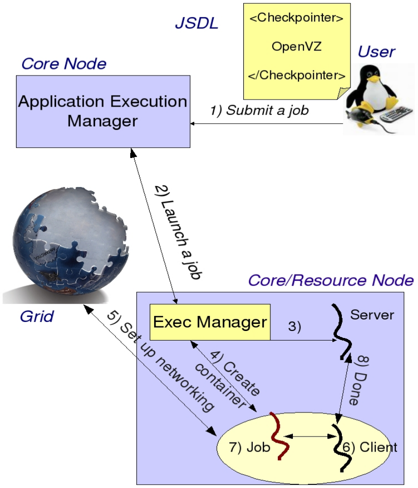
Job submission



Job submission

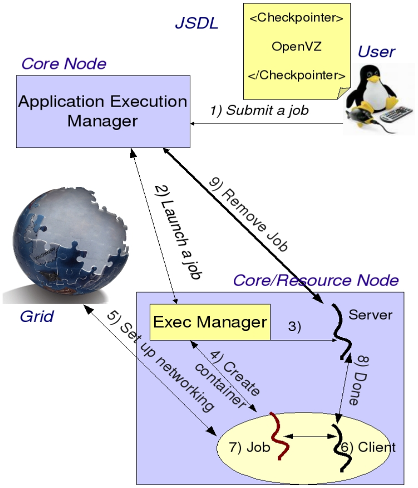


Job submission



XtreemOS
Enabling Linux for the Grid

Job submission



Checkpoint, Restart

Checkpoint

- Given a jobid a container id should be identified
- You checkpoint this container which executes the job
- If a container is not running, do not checkpoint

Restart

- Given a jobid the dump file should be located
- Given a jobid the container id should be identified
- A server should be restarted on the same port
- If a container is already running, do not restart
- After job finishes execution, suspend the container - for future restart

To sum it

Integration

- Foreign process dependency should be avoided at job submission for enabling checkpointing.
- Job tracking is important for job monitoring and cleanup of job
- OpenVZ integration lets an XtreemOS be submitted to a container.
- The job can be checkpointed and restarted to any node, once networking is set up.

Discussion Path

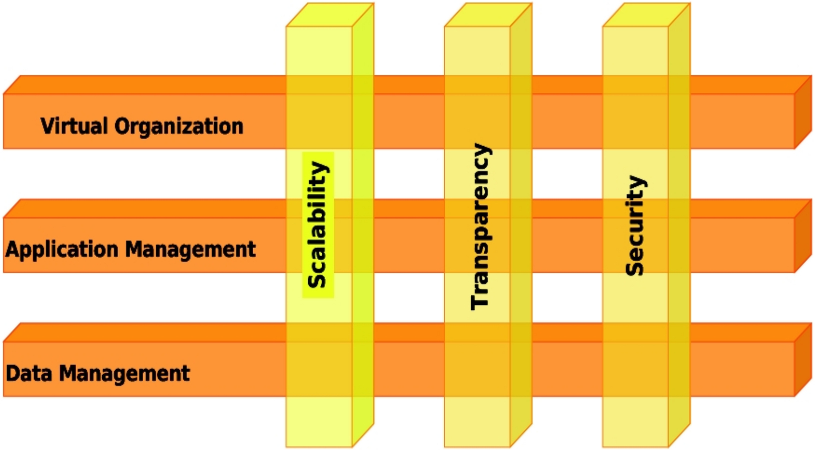
New Features

- Not a middleware but a O.S
- Support for interactive applications shall come soon
- All required grid related components in one CD
- execute legacy application the legacy way - ./elf-executable

Conclusions

- XtreemOS is *not a Middleware* but a *O.S*
- Provides true abstraction of grid resources
- Provides VO management and data management
- Provides scalability, security, fault tolerance, high availability
- Provides possibility of adding different security mechanisms
- Provides job monitoring, isolation, fault tolerance, debugging.
- Provides possibility of adding different checkpointers
- Easy to use, adminster, legacy application support
- Good solution for grids

Thank You !


Virtual Organization
Application Management
Data Management
Scalability
Transparency
Security