A Survey of Computing Migration

Xing Lin, Robert Ricci, Eric Eide
Flux Research Group, School of Computing

Introduction

Motivations:
- **Load balance**: Make more effective use of Clouds and Emulab resources
- **Save power and facility**: Consolidate computing to reduce power and facility consumption
- **Fault tolerance**: Migrate computing from partial failure machines; periodically checkpoint computing

Current State:
- Created taxonomy of mechanisms used in the literature
- Investigated how container, checkpoint and restart are designed and implemented in OpenVZ
- Generated initial ideas about how a new kernel could be designed

Future Plan:
- Design a better mechanism to support file system and network migration
- Integrate live Container migration of OpenVZ in Emulab
- Design a new kernel architecture that supports computing migration as a built-in capability

OS Level

Strategy:
Migrate the whole operating system environment

Features:
- Enabled by virtualization
- Migrate both operating system and process states

Representative systems:
- Xen, VMotion

Process Level

Strategy:
Migrate a single process or a group of processes

Features:
- Kernel-enabled or container based migration
- Lightweight but more complex

Representative systems:
- Zap, LXC, BLSR, OpenVZ

Language Level

Strategy:
Migrate objects, agents or threads across runtime environment

Features:
- Can tailor support for particular application classes
- Lightweight and OS independent

Representative languages:
- Java/JESSICA, Agent Tcl

System Comparison

Level: ■ OS ■ process ■ language

good fit for: ● clouds ● Emulab ● HPC

	Kernel objects	Memory	File system	Network	Good fit summary
■ MOSIX	Full support - redirection	Eager(dirty) ● ●	Global file system - extend UNIX fs	Full support - Redirection	● 0 ● 1 ● 1
■ BLCR	Partial support - checkpoint ● ● ●	Eager(all)	Linux file system	Not support sockets - supports MPI ●	● 1 ● 1 ● 2
■ Zap	Full support - checkpoint ● ● ●	Eager(dirty) ● ●	Pod Virtual fs - network fs ● ● ●	Full support - VNAT, DNS	● 2 ● 3 ● 3
■ JESSICA	Full support - redirection	Eager(dirty) - Delta sets ● ●	UNIX file system	Full Support - redirection	● 0 ● 1 ● 1
■ Xen	Full support - checkpoint ● ● ●	Precopy ● ●	Network-attached storage (NAS) ● ● ●	Full support - ARP ● ●	● 4 ● 4 ● 2
■ OpenVZ	Full support - checkpoint ● ● ●	Precopy or Eager(all) ● ●	Linux file system - chroot	Full support - ARP ● ●	● 3 ● 3 ● 1

Legend for Comparison

Kernel objects:
- **Full support**: Processes can use any kind of kernel objects
- **Partial support**: Certain kinds of objects are not supported
- **Redirection**: System calls will be redirected to the home node
- **Checkpoint**: Objects are dumped and recreated at another machine

Memory:
- **Eager(all)**: Processes are suspended and all memory space is copied
- **Eager(dirty)**: Processes are suspended and dirty pages are copied
- **Precopy**: Major memory is copied before processes are suspended. Then processes are stopped and modified pages are copied

Network:
- **Full support**: Network connections can be maintained during migration
- **Redirection**: A shadow process is left at home machine, to communicate with outside world. Migrated processes redirect packages to home machines
- **ARP**: Send out an ARP to advertise the new IP-to-MAC address mapping

This material is based upon work supported by the National Science Foundation under Grant No. 0709427. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.