HELSINGIN YLIOPISTO
HELSINGFORS UNIVERSITET
UNIVERSITY OF HELSINKI

# Virtualization and High Availability

Mika Karlstedt

AMICT'08

May 2008

**Faculty of Science**

# Content

- Virtualization
- High Availability
- High Availability combined with Virtualization
- My Research question

# Virtualization

- Software is used to create virtual machines
  - Similar concept than virtual memory
- Virtual machines run operating systems and appear to be real computers
- Control is in the hypervisor layer
  - Similar concept than operating system

# Virtualization techniques

- Full virtualization (e.g. VMware, QEMU)
  - Virtualizes the whole system with no support from the OS
  - 2 ways to implement (VMware vs QEMU)
- Paravirtualization (XEN)
  - Modified OS + hypervisor
- Light weight virtualization (Lguest, Openvz)
  - Modified kernel
- Hardware support helps

# Use cases

- Migration of Virtual machines
  - Helps management
- Isolation between virtual machines
  - Provides better security and reliability
    - Enhanced complexity reduces both
- Possibility to run different OS in the same physical computer
  - For example RTOS and Linux in mobile phone

# High Availability (HA)

- HA stands for High Availability
  - The service should be available always
  - Expressed in different ways (five 9s = 99.999 %)
- Requires hardware redundancy
  - In other words cluster of nodes (computers)
- Requires also complex middleware or HA framework

# Different kinds of HA

- There are different flavors of HA (for stateful servers)
  - Cold standby
  - Warm standby
  - Hot standby or Primary-backup or 2N redundancy
- Stateless servers are a special case
  - No state to preserve or restore

# Use cases

- Web farms
  - Cluster of stateless servers + stateful controller
    - Controller uses 2N model
- Telecom networks
  - Real-time capable stateful servers with low-level hardware redundancy (FRUs)

# Comparison

- Virtualization
  - One machine appears to be many machines
- HA
  - Many machines appear to be one machine
- They both provide prosess (server) management
  - HA mostly automated, virtualization mostly manual

# Why combine them

- Virtualization provides no protection against hardware failure
- Virtual machines needs to be managed manually
- HA requires heavy support from the application
  - In other words the application must built the HA in mind (and is tied to the HA framework)
- Upgrading applications is difficult in both
  - A bit easier in HA and basically impossible with virtualization

# Use cases for combined approach

- HA application with clustered application
  - We create the HA application but also use another application that has its own cluster
    - For example clustered DB to provide extra reliability
  - There can be just one entity controlling the cluster
  - HA framework creates a cluster of virtual machines for the clustered DB

# Use cases cont.

- Legacy applications
  - DX200 real-time OS and Linux
    - DX200 is old but too expensive to replace
    - Among other things it requires uniprocessor
  - New software is built on top of Linux
- Virtualization provides virtual hardware
  - Legacy systems may not support modern gigabit network cards

# Problem

- Writing real-time HA application is difficult
  - Checkpoints while the system is active
  - Replicas need to be consistent with each other
  - Recovery time must be short and automated
- In many cases HA is built into the application i.e. no HA framework
- There are standards for HA frameworks
  - But none is very good

# My use case

- Virtualization is used to provide HA
  - Migration is modified to cloning
    - Instead of migrating the system, we create a clone
  - Non-HA-aware applications can become HA
  - Framework clones the external messages
    - Primary deals with them and replies
    - Backup serves them normally but framework discards the replies

# Issues

- If the internal state is compromised, virtualization clones it
  - Both nodes crash at the same time
- Solution: create checkpoints
  - Advantage: cloned VM is not active
  - Disadvantage: we need to store the messages delivered after the VM was cloned and frozen
  - Framework should raise the priority of cloning processes

# Issues cont.

- How does framework know when the server has failed?
  - HA framework uses heartbeat
- HA applications use similar technique between the primary and backup
  - The application could help, but ...

# Issues cont.

- When the server uses external services
  - Primary sends the request and gets the reply
  - Reply is cloned to the backup
- How to ensure that backup sees the reply only after it has sent the request itself
  - Some kind of connection tracking?
- EverRunHA implements similar features

# Upgrading software

- Upgrading software is tricky
  - HA applications can run years => definate need for upgrades
  - Upgrading software while it is active is difficult
- Approach
  - Clone a new copy
  - Stop it gracefully => it writes the state to disk
  - Start upgraded version which reads the state from the disk
  - The framework stores events received during the upgrade
    - Actual primary is still serving the clients
- Requires little help from the application

# Conclusions

- Virtualization can make building HA applications easier
- But the virtualization framework must become a kind of HA framework itself
- Combining them can ease dynamic software upgrade