CSC548, Fall 2006
Homework 5 – Project Progress Report

Name: Arun Babu Nagarajan
Project: Enhanced Proactive Falut tolerance system for HPC with Xen virtualization
Website: http://www4.ncsu.edu/~abnagara/csc548_project

A) Preliminary analysis

Currently working on the pre-deployment option. A preliminary study was conducted to identify the memory usage associated with the NPB codes as shown in the table. Here RSS is the Resident Set Size which gives the memory usage of the process + shared pages. Vsize represents the total address space of the process. Also it is observed that, with the current configuration, the memory occupied before any process is running on the guest VM is around 250-300 MB. So there probably is scope for pre-deployment. Returning the free pages to Xen before migration is to investigated and implemented.

	Class B		Class C	
	RSS (MB)	Vsize (MB)	RSS (MB)	Vsize(MB)
BT	107.90	119.41	406.01	423.33
CG	110.45	125.85	282.09	306.23
EP	2.26	10.61	2.28	10.61
LU	59.14	50.30	183.06	193.60
SP	95.55	111.88	351.39	379.35

B) Migration framework analysis:

The migration framework, which is available as a part of the user-land tool suite has been analyzed and the files involved identified. The sequence of execution has been identified and the structure of image file is analyzed. (The flow is available in the website).

Started to instrument the migration framework for adding a function to compare the image/checkpoint files. Currently modified the python framework to suite our requirements and working on the C code now.

Where was the effort spent? : The migration framework is implemented as part python part C. Also there is interaction with xend. Understanding this and figuring out the way to include functions consumed time.

C) Gathered details on reading kernel data structures

The plan is to perform a black box comparison of image files. In case they don't match a lot, we need to identify the reason. Inspecting the kernel data structures in the image file might give us some idea. So some resources have been gathered which aim at reading the kernel image/dump and interpreting details like which are user pages, kernel pages, etc,. At this point, we are not sure yet whether this might be useful or a correct approach.

Timeline

Week ending Nov 10:

- Black box comparison of image files
    - Get detailed information on number of page matches
    - If very little common pages are identified the reason needs to be identified. (Possible reasons could be that the pages of a process are differently distributed in a different way)
- Experiment with stop/copy migration (rather than live migration) and gather data

Week ending Nov 17

- If the black box comparison did not give good results, reading the kernel data structures should be considered to identify reason
- Figure out how to use ballooning driver to release the free memory pages to Xen when not migrating

Week ending Nov 24

- Integrate ballooning into current migration framework
- Continue work on pre-deployment

Week ending Dec 01

- Sensitivity study on time migration
- Continue work on pre-deployment/ integration into migration framework if pre-deployment is found feasible