Liming (Michael) Yu
CMPE 297 – Fall 2012
Assignment #1

2. In the hypervisor architecture you chose, what is the mechanism by which a guest environment initiates a hypercall? What are the performance implications for this mechanism?

In XEN (the hypervisor architecture I chose), the mechanism by which a guest environment initiates a hypercall is through paravirtualization. The goal of paravirtualization, or semi or partial virtualization, is to avoid dealing with problematic instructions as much as possible. Because in a paravirtualized environment, one knows the incoming instruction stream (and could have prior access to the code), problematic instructions could be replaced with something equivalent that emulates it. The reason for this is to provide the ability to trap these instructions that a guest system could execute inside a paravirtualized environment.

The way this is implemented in a paravirtualized environment is that the hypervisor (which runs directly on the hardware of the host system to handle resource and memory allocation to the vm's in this case) resides in ring 0, the kernel resides in ring 1, and the user space (and all user applications) reside in a higher ring such as ring 4. The guest domain (vm) runs in ring 1 alongside the kernel. The guest domain (vm) uses hypercalls (to request privileged operations and instructions) to the hypervisor in ring 0. If the guest domain (vm) wants to request privileged operations and instructions, it sends a hypercall to the hypervisor in ring 0. This is possible because the hypervisor exposes a set of hypercalls that correspond to these instructions. Essentially, the hypercall acts as a software trap from the guest domain (vm) to the hypervisor.

In terms of performance, one goal in paravirtualization is to reduce the portion of the guest domain's (vm) execution time spent performing operations and instructions which are more difficult and take more time to run in a virtual environment versus a non-virtual environment. Through the use of hypercalls and the hypervisor, paravirtualization provides a way to allow the guest domains (vm) to perform these instructions. As a result, execution time is decreased and performance is improved in comparison to executing these instructions in a virtual domain (where the execution performance is worse). In comparison to full virtualization, paravirtualization was slightly faster based on benchmark testing here => http://shortrecipes.blogspot.com/2009/03/xen-performance-of-full-virtualization.html