Stýrikerfi
Project 3

Matthías Páll Gissurarson
mpg3@hi.is
Haukur Óskar Þorgeirsson
hth152@hi.is

23. janúar 2013

1

The first question we would raise here is wether or not this could indeed happen, as we would think the CPU timer of OS_2 would stop while OS_1 is running. The fact that it does not raises concerns because when the timer runs out OS_2 think it has given it's process enough time, but that proves to be wrong if the timer keeps running while OS_1 is running.

One idea would be to have the VM tell OS_2 to give said process more time, if this is possible. Otherwise, the VM could store whatever message the timer gives until OS_1's time is up. Another possibility is to give control back to OS_2 and then giving OS_1 more time the next time it gets a time slice. this, however could lead to one OS hogging the CPU, say OS_2's CPU timer goes off every 5 ms. Then OS_1 would always get far smaller timeslices.

2

One possibility is to have two small "transfer disks", buffers which have about as much space as one of the VMs can write in one timeslice, in reality files in the host. Both OSs can read both transfer disks (TDs), but only one OS can write in each TD. Then the hypervisor could act as a middle man in such a way that it would emulate some sort of bus (for example a LAN cable) "connected"to the VMs, or anonymous pipelines. Say OS_1 has a timeslice it uses to send data to OS_2. Then in reality the hypervisor writes the data OS_1 is trying to relay to TD_1, and when OS_1's time slice ends OS_2 recieves the message as if it were recieving it real-time.

Another implementation would be to emulate a shared virtual hard drive (a third drive) with locks preventing OS_2 from writing the same file OS_1 is writing and vice-versa.