CS167 Homework Assignment 2

Due 11:59pm, March 9, 2018

1. As discussed in class, virtual machines can be nested, i.e., a virtual machine monitor can run on a virtual machine. In what follows we say that the level-0 VMM run directly on the real hardware. A level-1 VMM runs in a virtual machine (VM) of the level-0 VMM. In general, a level i VMM runs in a VM of the level i-1 VMM. In this problem we consider how virtual machine monitors designed for Intel's VT-x enhancement to the x86 architecture can run on virtual machines. A rough description of how VT-x works is that the machine runs either in root mode (in ring -1) or in non-root mode (in rings 0-3). The VMM runs in root mode, its virtual machines (VMs) run in non-root mode. When in non-root mode, certain actions cause VMexits, which result in traps to root mode. The intent is that these VMexit-causing actions are those, such as modifying certain important registers or modifying certain memory locations, that would affect other VMs. The VMM specifies, for each of its virtual machines, which events cause VMexits.

    a. [5%] An "off-the-shelf" VMM for the VT-x architecture runs in hardware ring -1 and provides a virtual environment that appears to be an x86 with support for rings 0 through 3 only. Explain why such a VMM does not support nested virtualization. (The expected answer is short.)

    b. [10%] As part of switching to non-root mode to run a virtual machine, the VMM specifies which actions by the VM trigger *VMexits* (traps to the VMM). Thus, in principle, each VM may have a different set of actions that trigger *VMexits*. When virtualization is nested, *VMexit* traps cause the bottom-level (level-0) VMM to be invoked. Since only the bottom-level VMM is running in real ring -1, only it can switch to a VM and thus establish which actions by the VM cause *VMexits*. Suppose a VMM at level 1 (i.e., running in a VM of the level-0 VMM) attempts to switch to one of its VMs, and thus specifies which actions cause *VMexits*. The action of switching to a VM will have been set up by the level-0 VMM to cause a *VMexit*, and thus the action causes a trap to ring -1, which is handled by the level-0 VMM. Thus it's the level-0 VMM that actually tells the hardware which actions cause *VMexits*. What set of actions does it specify as causing *VMexits*? (The choices would be those actions specified by the level-1 VMM for its VM, those specified by the level-0 VMM for its VM, the union of these two sets of actions, or the intersection of these two sets of actions.)

    c. [10%] In general, when a level-i VMM, for i>1, switches to one of its VMs, must the level-0 VMM invoke the level-1 VMM, which in turn invokes the level-2 VMM, etc., or can the level-0 VMM start the level-i VM directly?

2. A standard feature on many x86-based systems has been a programmable interval timer (PIT). As used by operating systems, it is given an initial positive value. It then counts down at some frequency (say 1 MHz) until it reaches zero. When this happens, it sends a clock interrupt to the processor, resets itself to the initial value, and counts down again, ad infinitum. For the following questions, you may assume a uniprocessor system. The expected answers are relatively short.

    a. [12%] Assume that the PIT's initial value is set just once, when the system boots. Explain how the PIT might be used by an operating system to drive a time-slice-based scheduler, where each thread is given a time slice of one centisecond (hundredth of a second), and clock interrupts happen every millisecond (thousandth of a second).

    b. [13%] Our operating system is now running on a virtual machine. We would like its threads to get one-centisecond time slices of virtual time, i.e., even though several

centiseconds of real time may have elapsed, the time slice is not over until the virtual machine has been running for a centisecond. Explain what is done on the virtual machine monitor (VMM) to ensure that this happens. Note that it's not necessary for these time slices to be measured exactly, as long as they are on average the correct length (with low variance).

3. Assume we're using the Rhinopias disk drive.

    a. [8%] Suppose we are using S5FS and we've doubled the block size from 512 bytes to 1024 bytes. Is the maximum expected transfer rate roughly doubled? Explain.

    b. [8%] We've done the same with FFS. Is the maximum expected transfer rate roughly doubled? Explain. Assume that we're using two-way block interleaving: when we allocate successive blocks on a track, one block is skipped. Thus, in the best case, every other block of a track is allocated to a file.

    c. [9%] Why should we care about the block size in FFS?

4. [25%] Explain how renaming a file can be done using the consistency-preserving approach so that a crash will not create situation in which the file is lost. Be sure to take into account the link count stored in the file's inode. Note that there may have to be a temporary inconsistency; if so, explain why it will not result in lost data.