CMPE283 – Virtualization
Final Exam

Name: **Sunil Tiwari (011825476)**

Select the best answer for each question. Each question is worth 5 points unless otherwise noted.

1. (15 points) When optimizing hypervisor performance, it has been stated that optimizing to reduce the total number of exits provides the largest amount of performance improvement, as compared with other optimization techniques. Why is this the case?

https://software.intel.com/en-us/blogs/2009/06/25/virtualization-and-performance-understanding-vm-exits

VM exits in response to certain instructions and events (e.g., page fault) are a key source of performance degradation in a virtualized systems. But have you ever wondered why? What exactly happens during a VM exit anyway?

A VM exit marks the point at which a transition is made between the VM currently running and the VMM (hypervisor) which must exercise system control for a particular reason. In general, the processor must save a snapshot of the VM's state as it was running at the time of the exit. For Intel architectures, here is an approximation of the steps:

1. Record information about the cause of the VM exit in the VM-exit information fields (exit reason, exit qualification, guest address), and update VM-entry control fields.

2. Save processor state in the guest state area. This includes control registers, debug registers, MSRs (see next item), segment registers, descriptor-table registers, RIP, RSP, and RFLAGS, as well as non-register state like pending debug exceptions.

3. Save MSRs in the VM-exit MSR-store area. MSR stands for Machine Specific Registers, in case you are not familiar. They are used to control and report on processor performance.

4. Load processor state based on the host-state area and some VM-exit controls. This includes host control registers, debug registers, MSRs, host table and descriptor-table registers, RIP, RSP, and RFLAGS, page-directory pointer table entries, as well as non-register state.

5. Load MSRs from the VM-exit MSR-load area.

Oh, and don't forget that after the VMM has performed its system management function, then a corresponding VM entry will be performed that transitions processor control from the VMM to the VM!

Now you can see why VM exits generate considerable overhead, to the tune of hundreds or thousands of cycles for a single transition. To mitigate this problem, BTW, considerable effort has gone in to both

reducing the number of cycles required by a single trasition, and creating features that obviate the need for system management (and hence VM exits) in the first place.

2. (15 points) Explain how ASIDs/VPIDs improve performance when used with nested paging.

http://developer.amd.com/wordpress/media/2012/10/NPT-WP-1%201-final-TM.pdf

1. *Starting 64-bit AMD Opteron Rev-F processors support Address Space IDs (ASIDs) to dynamically partition the TLB.*
2. *The hypervisor assigns a unique ASID value to each guest scheduled to run on the processor.*
3. *During a TLB lookup, the ASID value of the currently active guest is matched against the ASID tag in the TLB entry and the linear page frame numbers are matched for a potential TLB hit.*
4. *Thus, TLB entries belonging to different guests and to the hypervisor can coexist without causing incorrect address translations, and these TLB entries can persist during context switches.*
5. *Without ASIDs, all TLB entries must be flushed before a context switch and refilled later.*
6. *Use of ASIDs allows the hypervisor to make efficient use of processor's TLB capacity to improve guest performance under nested paging.*

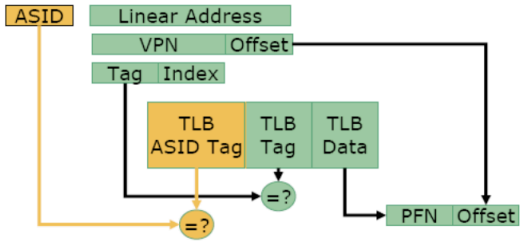

Figure 6: Address Space ID (ASID)

3. (15 points) What problems would a hypervisor author need to overcome when creating a VM live-migration algorithm that worked across different host CPU types?

- During the live migration, the disk should be shared between the two hosts(VM's)
- The migration should be done on as strong network to make any forward progress white transferring the memory pages.
- Both the host VM's(Source and Destination) should have the same architecture

(Intel → Intel)
(AMD → AMD)

- Both The CPU's should not have significant difference in the of features;

- _____C_____ What would happen first if “external interrupt exiting” was enabled on a VCPU and an external interrupt was received by the PCPU?

    A. Not enough information is provided to answer the question accurately
    B. Control would flow to the currently running VM’s interrupt handler for that interrupt
    C. The interrupt would always be dispatched to the VM’s non maskable interrupt handler OR Nothing, if the interrupt was masked
    D. The guest VM would exit to the VMM
    E. The interrupt would be held, and delivered during the next guest VM entry

- _____B_____ Which of the following statements regarding IOMMUs is true?

    A. An IOMMU provides safety when dealing with passthrough devices performing DMA
    B. An IOMMU provides safety when dealing with emulated devices performing DMA
    C. IOMMUs are used for port-based I/O using IN/OUT instructions
    D. None of the above

- __**True**__ True/False? Type 1 hypervisors run natively on the host hardware.

http://searchservervirtualization.techtarget.com/answer/Virtual-security-tactics-for-Type-1-and-Type-2-hypervisors

There are two types of hypervisors: Type 1 and Type 2.

Type 1 hypervisors run directly on the system hardware. They are often referred to as a "native" or "bare metal" or "embedded" hypervisors in vendor literature.

Type 2 hypervisors run on a host operating system. When the virtualization movement first began to take off, Type 2 hypervisors were most popular. Administrators could buy the software and install it on a server they already had.

Type 1 hypervisors are gaining popularity because building the hypervisor into the firmware is proving to be more efficient. According to IBM, Type 1 hypervisors provide higher performance, availability, and security than Type 2 hypervisors. (IBM recommends that Type 2 hypervisors be used mainly on client systems where efficiency is less critical or on systems where support for a broad range of I/O devices is important and can be provided by the host operating system.)

- _____**False**_____ True/False? If VT-d is supported on a host, it must be enabled in order to use
  VT-x/VMX.

VT-d/VT-i is for device virtualization features.
VMX/SVM is for hardware assisted virtualization (for trapping the non-trappable instructions)

8. (20 points) Describe how the TLB is used by the processor, VMM, and guest VMs in a multi-cpu host environment.

http://www.informit.com/articles/article.aspx?p=29961&seqNum=4
http://www.freepatentsonline.com/6490671.html

A method for maintaining virtual memory consistency in a multi-processor environment comprises allocating a subset of virtual memory to a process, and mapping the subset of virtual memory to a first subset of physical memory. A translator lookaside buffer (TLB) is maintained in each processor, each TLB comprising a plurality of TLB entries. Each TLB entry represents a mapping between a virtual address in the subset of virtual memory and a physical address in the first subset of physical memory. When the subset of virtual memory is to be unmapped, a reference to the first subset of physical memory is placed into a free list, and marked as dirty. When the number of dirty references exceeds a predetermined threshold, the corresponding entries in each processor's TLB are invalidated. Alternatively, all TLB entries can be invalidated. The free list comprises a plurality of free list entries, where each entry comprises a reference to virtual memory which is either unmapped or whose mapping is dirty.

Note that a context switch normally requires that the entire TLB be flushed. However, because this is such a common operation and because TLB fault handling is relatively slow, CPU architects over the years have come up with various schemes to avoid this problem. These schemes go by various names, such as *address-space numbers, context numbers, or region IDs*, but they all share the basic idea: The tag used for matching a TLB entry is expanded to contain not just the virtual page number but also an address-space number that uniquely identifies the process (address space) to which the translation belongs. The CPU is also extended to contain a new register, asn, that identifies the address-space number of the currently executing process. Now, when the TLB is searched, the CPU ignores entries whose unique number does not match the value in the asn register. With this setup, a context switch simply requires updating the asn register—no flushing is needed anymore. Effectively, this scheme makes it possible to share the TLB across multiple processes

9. **(15 points) Describe how LXC implements namespace virtualization, and what namespaces it provides. (Not In Syllabus)**