CS302 OS Mid-term Homework

> Name: 刘仁杰
>
> SID: 11911808

1. Terminologies of virtualization

(1) Define virtualization in 100 words?

Virtualization is a technique that relies on software to simulate hardware functionality and enable the possiblity to run serveral different virtual systems on a single hardware which solve the problems such as legacy upgrading and resources underutilization.

(2) Please explain the three usage models of virtualization: workload isolation, workload consolidation, workload migration.

- workload isolation: Workload isolation by utilizing virtualization has benefits to the overall system in two perspectives. One is system security, which will be improved because instructions executed are confined to the inner of the VM, so vicious instructions won't do harm to the hardware. The other is system reliability, which can be enhanced because software programs are seperated in different VMs, so software crash on one VM will not lead to the misbehavoir in other VMs.
- workload consolidation: There are mainly three benefits that virtualization has brought about to workload consolidation. The first two are to solve the problem of proliferation of heterogeneous data and underutilized servers, virtualization combine distributed workloads on serveral VMs managed by one VMM regardless of different hardware, reducing the difficulties to manage the numerous data and better leveraging the hardware resources. The Third is that virtualization eliminate the necessity to do entire system software-hardware upgrades by allowing physical devices to run legacy and new OSes and softwares concurrently.
- workload migration: virtualization decouples the guest from the underlying hardware. So, it is easier to migrate the guest to a different platform. In addition, whenever there are workload balancing or failure-prediction agents, workload migration can be automatically triggered by VMM, which improves the service quality while undertaking a lower operational cost.

(3) List 3 well-known VMMs.

- Xen
- Virtual PC
- VMotion

(4) Explain the following terms: paravirtualization, full virtualization, binary translation, hardware-assisted virtualization, hybrid virtualization.

- paravirtualization: In paravirtualization, guest OS are modified at compile time inside VM in order to support hypercalls handle instructions and communicating to the host OS and hardware device.
- full virtualization: In virtualization, the virtual hardware is not modified and functionally identical to the underlying true hardware device.

- binary translation: In binary translation, guest-OS binaries are modified directly by VMM to support legacy OS. Thus, this technique can support more OS, even very old and outdated ones. However, this modification endure a higher performance overhead.
- hardware-assisted virtualization: In hardware-assited virtualization, VMM runs in the privilege mode and allow a cpu instruction set to run directly without modifying guest OSes.
- hybrid virtualization: hybrid-virtualization combines paravirtualization and hardware-assisted virtualization. For I/O devices, Timer, Idle handling, Interrupt controllers and MMU are identified to use para-virtualization while maintaining the same cpu behavoir as the hardware-assisted virtualization.

2. Privilege levels

(1) How many privilege levels does x86 (IA-32) provide? How are they used by OS and user processes (considering an ordinary OS without virtualization)? Please provide three examples of the importance of privilege separation.

1. x86 (IA-32) provides 4 privilege levels, from 0 (highest) to 1 (lowest).
2. In a nonvirtualized system, OS operates at level 0 and all software applications run at level 3.
3. Three examples of the importance:
    1. This can protect the hardware which will only be accessed by trustedworthy processes with higher previliege, preventing the hardware being messed up by user processes.
    2. When user processes crashed, this ensure that OS kernel processes with higher privilege won't be blocked by processes at a lower previliege, and can continue to clean up the mess and mantain the stability of the system.
    3. Processes with higher previliege can mitigate the potential damage of a computer security vulnerability if vicious user programs take place.

(2) What is ring compression?

Ring compression is the situation where guest OS and guest application should run in different ring privileges, but limited by IA-32 pageing in 64-bit mode, privilege levels 0-2 are not distinguished, so guest OS and guest software must in the same privilege. Thus, guest OS can not be protected from the protential harm brought about by guest software processes.

(3) Without Intel VT-x, how does Xen address ring compression for X86 (IA-32)?

Without Intel VT-x, Xen run guest OS in ring 1 and application run in ring 3.

(4) Without Intel VT-x, how does Xen address ring compression for x86-64?

Without Intel VT-x, hypervisor runs in ring 0 with both guest kernel and applications run in ring 3. However, guest kernel run in different address space (different page tables) from applications. Since processes in Xen have two different page tables seperately in application context and kernel context, failure or crash on application context won't lead to the crash of the kernel. Also, The context transition between application and kernel is handled by Xen. Thus, vicious application process won't have an effect on kernel context.

(5) What is ring aliasing?

Ring aliasing refers to situation where software is run at a privilege level other than the level for which it was written. First example is that push instruction will push the current privilege level on the stack. Another example is that br.call will store current privilege level into PFS register which can be read at any privilege level. In this two cases, the guest OS can easily know it is not running at privilege level 0.

(6) What are VMX root and VMX non-root in VT-x?

VMX root operation and VMX non-root operation are two forms of CPU operation which support all four privilege levels in VT-x architecture. VMM is ran in VMX root while its guest OSes and applications are ran in VMX non-root. Moreover, with different privilege levels privided, the guest OS can run at its intended privilege level, while the software running in VMX non-root is deprivileged.

(7) How does Intel VT-x address the challenges of ring aliasing and ring compression?

- ring aliasing: Intel VT-x allows a VMM to run guest software as its intended privilege level. So instructions like push and br.call won't reveal that the guest OS is actually running in a virtual machine.
- ring compression: In VMX non-root operation of Intel VT-x, guest OS can run at its intended privilege levels with all four privilege levels provided. Thus the guest OS won't be ran at the same privilege as software.

3. System calls, interrupts and exceptions

(1) In the context of ordinary OS without virtualization, what are the purpose of system calls? What is the main difference between system calls and function calls? On x86-64 Linux, how are system call parameters passed to the kernel?

- purpose of system calls: system calls provide an API between a process and an OS to allow user-level processes to request services of the operating system. It can also protect the OS and hardware from directly accessed by user processes since system calls are the only entry points into the kernel system.
- Main difference: system calls contains a switch from user space to kernel space, while functional calls remains in the user space.
- On x86-64 linux, system call parameters are passed to the kernel through cpu registers by signal trap.

(2) What is a hypercall in Xen?

In Xen, hypercall is a interface for providing privilege or root service for the processes in VM by Xen. The hypercall triggers a software trap into the hypervisor to perform a privileged operation, like the system calls in ordinary OSes.

(3) How does Xen virtualize exceptions on x86 (IA-32)? What modifications does Xen make to the original x86 exception handlers?

- How: A table describing the handler for exception is registered. The stack frames are unmodified in Xen's paravirtualization architecture, which remains generally the same with those for real x86 hardware. When an exceptions occurs outside ring 0, Xen's handler creates a copy of the exception stack and return proper handler to handle the exception.
- Modifications: The key modification is for the page fault handler. Since guest OS can not read fault address from the CR2 which is privileged, Xen write it into an extended stack frame.

(4) What are the challenges of virtualizing interrupts (especially regarding interrupt masking) on x86 (IA-32)?

Typically, A VMM will deny the ability to control interrupt masking. When the guest attempts to control interrupt masking, a fault will incur in the context of ring deprivileging. Then, frequently masking and unmasking interrupts will significantly affect system performance, since VMM will need to intercept every guest attempt to do so.

Moreover, if the VMM don't need to intercept every attempt of the guest to modify the interrupts, but deliver virtual interrupt to the guest when the guest has unmasked interrupts, the design of a VMM will be complicated.

(5) How does Xen virtualize interrupts on x86 (IA-32)? What is the benefit of such a design?

- How: In Xen, hardware interrupts are replaced with a lightweight event system. Rather than releasing directly control of the hardware device, Xen triggers appropriate interrupt service routine within Xen.
- benefit: In this way, Xen retains tight control of the system.

(6) What is VMCS in Intel VT-x? What are VM exits and VM entry? How are VMCS used during VM exits and VM entry

- In Intel VT-x, VMCS is the abbreviation for virtual-machine control structure, which manages VM entries and VM exits and processor behavior in VMX nonroot operations.
- VM exits represents the transition from VMX non-root operation to VMX root operation, i.e., from guest to VMM. VM entry indicates the transition from VMX root operation to VMX non-root operation—that is, from VMM to guest.
- In VMCS, two of the sections are the guest-state area and the host-state area which contains fields corresponding to different components of processor state. During VM exits, guest processor state is saved to the guest-state area and new processor state is loaded from the host-state area. During VM entry, processors state is loaded from guest-state area.

(7) How does Xen leverage Intel VT-x to virtualize interrupts?

In Intel VT-x arthitecture, interrups are configured to cause VM exits conditionally which will enter in to VMX root operation and handle the interrupts with proper interrupt handler. There are two conditions (controls) to trigger VM exits. One is when the external interrupt exiting control is set, all external interrupts will lead to VM exits. The other is when the interrupt-window exiting control is set, if the guest software is ready to receive interrupts, VM exits will occur.

(8) How does Intel VT-x support exception virtualization?

Exception is supported also by VM exits. Exception bitmap which contains 32 entries for the IA-32 exceptions is recorded for VMM to specify which exceptions will cause VM exits and which will not.

4. Address translation

(1) Explain x86 (IA-32) address translation.

In x86 (IA-32) arthitecture, both segmentation and paging are adopted. So the sequence of address translation should be: virtual address -> segmentation table -> linear address -> page table -> physical address. In the paging translation step, 2-level page table structure is adopted, where the highest 10 bits are for the first-level page table to locate the address of the second-level page table, the mid 10 bits are used to locate the base address of the physical page, and the last 12 bits are used for offset within page.

(2) Explain x86-64 address translation.

x86-64 support similar structure of segmentation + paging as that in x86 (IA-32). The major difference is that x86-64 leverages 4-level page table where the first 16 bits are unused, the middle 36 bits are divided into 9-bit groups to maintain a 4-level page table structure, and the last 12 bits are offset.

(3) Explain the relationship between guest virtual memory, guest physical memory, and machine memory.

- guest virtual memory: the virtual memory where guest processes run programs, fetch and store data.
- guest physical memory: the physical memory translated from guest virtual memory by guest OS, which is just a virtualization of machine memory by VMM.
- machine memory: the true physical memory of the hardware device, translated by VMM or host OS.

The translation procedure should be: guest virtual memory -> guest OS -> guest physical memory -> VMM -> machine memory

(4) How does Xen manage the per-process page table in the VM and the per-OS page table in the VMM?

- per-process page table: This page table is registered as read-only in Xen and stored in guest OS. guest OS will need to use hypercall to update the page table through Xen which is in charge of validation and execution.
- per-OS page table: This page table is transparent in all guest OSes (all VMs). Every guest OS can translate guest physical memory to machine memory according to this page table.

(5) What does “Address-space compression” mean?

This refers to the situation where some portions of guest virtual memory are reserved by VMM to provide service. Then how to protect these memory fields from guest OS and user processes while also providing access to them becomes a problem.

(6) How does Xen address the problem of “Address-space compression”?

In Xen architecture, those reserved memory fields are read-only to guest OSes and processes. When a guest OS attempts to write or update within the fields, the action is taken over by Xen for first validation and then execution, which protects the memory fields while still providing access.

(7) What is Intel EPT? How to do MMU virtualization with Intel EPT?

Intel EPT is the abbreviation for Extended Page Tables, is the replacement of shadow page table. It is introduced to mitigate the overheads caused by the translation of shadow page table.

When EPT is enabled and active, the extended page tables can be directed accessed by guest OS with EPT base pointer to do the translation from guest physical address to host physical address with not VM exits, which thus reducing paging related overheads.

(8) How does Xen allocate physical memory to each domain?

When each domain is created, a portion os physical memory is allocated by Xen, resulting a memory isolation from other guests. Meanwhile, states of maximum allowed memory and currency allocated memory is recorded. When a guest or a domain requires to extend its memory space, a check if the currently allocated has reached the maximum limit will be conducted to decide the allocation.