CMPSCI 677 Operating Systems **Spring 2016**

Lecture 6: February 8

Lecturer: Prashant Shenoy *Scribe: Wang Xu*

6.1 Virtualization Continued

6.1.1 Hypervisor Review

For Type 1 hypervisor, it runs on "bare metal", which means there is no operating system under it. It usually comes from a modified Linux kernel, seen as a special operating system for the machine. For Type 2 hypervisor, there is a host operating system running under it. And guest operating systems running on it are seen as normal applications for the host operating system.

6.1.2 Memory Virtualization

Virtual machines will create and modify the virtual pages using their own OS. On the other hand, the hypervisor will map those virtual pages into shadow page table. Every time the virtual machine modifies its page table, it will generate a trap passing onto the hypervisor. And then the hypervisor will deal with those sensitive instructions and make the actual changes in the shadow page table.

6.1.3 I/O Virtualization

Disk: Each guest OS thinks it "owns" the disk. Hypervisor creates a "virtual disk" in its file system like a file. Every time guest OS wants to access the virtual disk, the hypervisor will permit the action and access to the actual storage. If a VM is cleaned, the hypervisor will clean its "disk" file as well. For example, when a VM runs on VirtualBox, it shows like a normal disk from the guest OS. But for the host OS, this "virtual disk" is just a big file with the extension name ".vdi". For some file systems, there may be maximum size limit for a file. The hypervisor will divide the virtual disk into several files in the physical storage. But it is still only one logic file for the VM.

Network: A logic network card is built in VMs. From outside, it is just a normal software. When the network is working, this logic layer will be mapped into the physical network interface by translating the IP addresses.

6.1.4 Virtual Appliances and Multi-Core CPUs

Virtual Applicance is a package that has pre-configured VMs with applications in it. It can be downloaded by users from the internet and run. Today, virtual appliances have evolved into docker containers.

6.1.5 OS Virtualization

OS virtualization is a lightweight virtual technology that helps to isolate applications from each other with the host OS support. Each container or zone can be built for a specific program with limit resources.

Containers are easy to set up and delete. The process in a container is running on the host OS with very little overhead comparing virtual machines.

Different mechanisms can be used for controlling the containers. Namespaces is a process based resource isolation which provide an illusion to a contained process that it is the only process running in the system. Contained processes can not see each other. They have their own visible file system. Some permissions can be set up that if this process can be seen by certain group of users or what the container can do or not. Linux cgroup is another isolation which is resource isolation. It can limit hardware resource for a container. For example, it can set up number of cores, maximum memory space or maximum cpu usage time for a container.

There are several schemes for container scheduling. One is share-based scheduling. A weight is assigned to each process. For example, time slots are distributed to different processes according to these weights. Some unused cycles can be assigned to other processes. Another scheme is credit-based scheduling, which means one process can actually generate their credits when unused time slots appear, and then use them in the future.

Docker is a technology built based on the Linux container. It is a portable Linux container. Developers can build a container using their machine, and users can download these dockers and run it as a container in their own machines. Several dockers can be built together for microservices.

6.1.6 Use of Virtualization Today

For data center, using server consolidation, multiple virtual servers from old machine can be packed and reallocated on newer machines. Another example is that users can develop and test their programs in virtual machines on their desktop instead of using other additional machines.

6.2 Server Design Issues

Usually a client sends a request to a server and the server responses this request to the client. The client must figure out what application is running on the server and what port is open for these requests. A server can be stateful, which means a session is open when a request appears. The server has to track every session. For stateless servers, useful data can be stored in client side. Soft state can be used to improve performance. One of the soft state design is caching.

From server architecture view, the server can be pure-sequential which means the server deal with the requests one at a time. And the server can be event-based. For example, if a I/O exception happens in a process, the server can switch to another process. There are also thread-based server architecture and process-based architecture. According to the performance, the order of these 4 architectures from best to worse is event-based, thread-based, process-based and pure-sequential.