Lecture 30 - Virtual Machines - Jails and Containers

Videos

https://youtu.be/qvqJSzu6IpQ - Lect-30-2150-pt1-virtualization.mp4
https://youtu.be/iVRZ7-C4AoA - Lect-30-2150-pt2-VirtualBoxDemo.mp4
https://youtu.be/b8F1WhN47us - Lect-30-2150-pt3-Docker.mp4

From Amazon S3 - for download (same as youtube videos)

http://uw-s20-2015.s3.amazonaws.com/Lect-30-2150-pt1-virtualization.mp4
http://uw-s20-2015.s3.amazonaws.com/Lect-30-2150-pt2-VirtualBoxDemo.mp4
http://uw-s20-2015.s3.amazonaws.com/Lect-30-2150-pt3-Docker.mp4

Vitalization of systems

My first experience with vitalization of systems was the post-office conversion from system IBM 360 to IBM 370. They needed more performance - rather than re-write the old software - they purchased new hardware - the new System 370 came with a “Virtual Machine” that could run 6 OS 360 systems on OS 370 - they brought up the 6 systems and - instant results.

The basic idea is to emulate the entire hardware of a system on another system.

Most people did not experience vitalized systems for a long time after this. The vitalization that was available was a different sort - it is what we call “containers” to day but it was built into Unix systems - specifically it was a thing called a “jail”. It still exists in FreeBSD today - and if you have ever sued Yahoo you are using BSD/Jails. Basically the idea is to vitalize all the system calls and change files so that processes inside the container stay inside the container. This is OS Level - and it has very little overhead - with some tremendous security advantages.

I run an email-relay system for a company that uses this - purely for the added security. What is inside the “container” or “jail” is all that the process can see - so you don’t have to include stuff like password files - or - tools - or - much of anything. Just what the server / service needs to run. A high level of isolation is achieved. BSD Jails has this down to a science. A subset of this is a thing called “chroot”. In a Unix system this was originally developed for testing new operating systems. The “root” of the file system is “/” - but what if you create a new test system - with all the stuff you want to test - and - tell the OS that when this process runs it will use the “/users/pschlump/test1” as the “root” of the file system. The process sees this as “/” -the outside- sees it as

/usrs/pschlmp/test1" - so the inside can't get out - it is trapped inside and can only see what is visible to it. BSD applies this same principal to system calls - privileges - processes - network interface etc. You can put a "process" into a "jail" and it is trapped on the inside - contained from the rest of the OS so that it can not do any harm. Fantastic security.

About the time that this came along - Intel finally built enough new VTx extensions to the x86 processor so that you could build a virtual machine. VMWare was created (now a multi-billion dollar business) to take advantage of this and vitalize systems. This is where you have a "host" OS that runs multiple "virtual" systems - possibly with different OSes and allows for complete emulation of the vital system. Oracle VirtalBox is one of these that is free to download and use for non-business applications.

Full vitalization has 2 flavors. Systems that run on the same type of a processor - and interrupt some calls. - this is like VMWare/Fusion where the host hardware is the same as the guest hardware. VMWare, Parallels, KVM, VirtualBox all fall into this category. Then there are full emulators -where you emulate a different CPU- QEUM and Unicorn are in this category. For example I can run a MIPS version of FreeBSD on my x86 processor using QEMU.

When you do iOS or Android development you use a full emulator. Both of these system are running on ARM processors - chances are very high that your development environment ins on an Intel x86 based processor. The emulator will need to do more than just emulate the hardware - it has to emulate different screen sizes, different hardware setups and hardware that you don't have on your development environment.

Virtual Systems are very useful for testing. You can (and I do) run a windows test environment on my Mac. I also run multiple different Linux environments.

1. Demo of Virtual Box
2. Demo of using a Docker Container