CSE291 Virtualization - Course Overview

Yiying Zhang
Before we start

• This course is about virtualization (in different forms). If you feel that this is not for you, e.g., if you are in the wrong class (this is CSE291-J), if you prefer exams over paper reading, if you don’t like doing one big group project per semester, if you have no software development experience, if you have no knowledge of undergrad OS+architecture, please feel free to leave now.

• Class website: https://cseweb.ucsd.edu/~yiying/cse291j-winter20/

• Class time: TTh 2pm - 3:20pm in Room CSE4140
Who Am I

- Fifth-year assistant prof at UCSD CSE (at Purdue ECE before 2019 summer)
- Working on datacenter systems (OS, dist sys, architecture, networking, etc.)
- Passionate about systems building (currently building next-gen datacenters)
- Also a part-time employee at Databricks
Course Structure

• About half are lectures given by me or paper discussion led by me

• Rest are paper discussion led by student volunteers (with bonus points)

• About 2/3 of the classes require paper reading
  • Need to submit paper summaries and answer questions before class

• Rest are online material learning

• Class attendance and participation in class discussion

• 3 quizzes throughout the quarter
Grading

• 25% Paper reading and class participation/attendance (attendance tracked from next week)

• 15% Quizzes

• 60% Project
  • Project proposal 15%  Jan 26
  • Project progress report 5%  Feb 16
  • Final project presentation 7% (about 10 min per group)  Mar 12
  • Final project report 33% (3-6 pages)  Mar 20

• Up to 10pts Bonus (discussion lead)
Sample Turn-in of Paper Summary

Only PDF is accepted
Name your PDF as [month]-[day]-[your first name]-[your last name].pdf

9-27-yiying-zhang.pdf

A Comparison of Software and Hardware Techniques for x86 Virtualization

Summary and your overall feeling of the paper: 2-5 sentences

• Q1: Why is x86 un-virtualizable with trap-and-emulate? Give one example.
  • A: 1-3 sentences

• Q2: How are jump instructions translated?
  • A: 1-3 sentences

• Q3: With hardware virtualization extensions (e.g., Intel VT), do we still need binary translation? Why or why not?
  • A: 1-3 sentences
Paper Discussion Lead

To gain bonus points, you can volunteer to lead or co-lead paper discussion. Here are what you need to do for discussion lead:

• Prepare slides that you will use to lead discussion. You can use slides that are available online, build slides yourself, or extend existing online slides with your own content.

• Your should not only just explain the paper (using existing online slides) but also prepare questions that will stimulate class discussion.

• Send draft slides to me no later than noon of the day before the corresponding class date. I will make comments and suggestions, which you should incorporate in the final version. The sooner you send your draft slides to me, the more helpful feedback I can give you.
Course Project

• One research-oriented, open-ended project in groups of 1-4

• Be prepared to do substantial programming and self-learning, start early!

• Most projects have assigned topics
  • But you need to formulate your own problem and solution
  • Allow self-defined project topics, but need to discuss with me ASAP

• Form your group by the end of next Friday!
Sample Project Topics

- Study: Profiling light VMs and/or library OS
- Build+Study: Serverless vs. Container vs. VM
- Build: Fine-grained resource mgmt and billing for VM/container/serverless
- Attack: VM/container/serverless security holes

- Full project topic description will be out by the end of this week

More Exciting Research at WukLab!
Tentative Schedule

• 1 week: intro and virtualization overview
• 1.5 weeks: virtualizing CPU, memory, and I/O
• 1.5 weeks: container, Docker, Kubernetes
• 1 week: serverless
• 1 week: Unikernel and Light VM
• 1 week: para-virtualization
• 0.5 week: VM migration/replication
• 1.5 weeks: hardware support and virtualizing non-traditional hardware
• 0.5 week: security
• 1 week: course summary and project presentation
Diversity and Inclusion

• Respect for Diversity: It is my intent that students from all diverse backgrounds and perspectives be well served by this course, that students’ learning needs be addressed both in and out of class, and that the diversity that students bring to this class be viewed as a resource, strength and benefit.

• All people have the right to be addressed and referred to in accordance with their personal identity. In this class, we will have the chance to indicate the name that we prefer to be called and, if we choose, to identify pronouns with which we would like to be addressed...I will do my best to address and refer to all students accordingly and support classmates in doing so as well.
As a student at UCSD you are subject to the UCSD POLICY ON INTEGRITY OF SCHOLARSHIP, which enjoins you to respect the highest standards of honesty and integrity. All work that you submit in this course must be your own; unauthorized group efforts are considered academic dishonesty. Academic dishonesty is a serious offense which may result in suspension or expulsion from the university. Students are encouraged to report academic dishonesty to the instructor directly, or to the Academic Integrity Office.
Another Chance To Leave Now
Why Study Virtualization?

- Almost all cloud applications run in the virtualization environment
- Most IT infrastructures run in the cloud or on-prem virtualization env
- Understanding virtualization is key to building cloud infrastructures
- Understanding virtualization will help application design
Adoption of Virtualization


2% - 71% - 82%

PROPRIETARY INFORMATION © 2014 Tintri Inc. All Rights Reserved.
Regular Machine Stack

vi  gcc  firefox

OS

Hardware
What is an OS?

• A piece of software that manages and virtualizes hardware for applications
  • An indirection layer between applications and hardware
  • Provides a high-level interface to applications,
  • while interact with hardware devices with low-level interfaces
  • Runs privileged instructions to interact with hardware devices

• Applications
  • Can only execute unprivileged instructions
  • Perform system calls or faults to “trap” into OS
  • OS protect applications from each other (to some extent) (e.g., address space)
Virtualization

• Adding another level of indirection to run OSes on an abstraction of hardware

• Virtual Machine (Guest OS)
  – OS that runs on virtualized hardware resources
  – Managed by another software (VMM/Hypervisor)

• Virtual Machine Monitor (Hypervisor)
  – The software that creates and manages the execution of virtual machines
  – Runs on bare-metal hardware
Virtualization

- vi
- gcc
- firefox

Guest OS
Virtual HW
OS (VMM)
Hardware
Virtualization: A Brief History
Mainframes and IBM

• Before we have datacenters or PCs, there were giant metal frames

• Support computational and I/O intensive commercial/scientific workloads

• Expensive (IBM 704 (1954) costs $250K to millions)

• “IBM and the seven dwarfs” – their heyday was the late ‘50s through ‘70s
Issues with Early Mainframes

• Different generations were not architecturally compatible

=> Headache to port software

• Batch-oriented

• Meanwhile, ideas started to appear towards a time-sharing OS

• The computer was becoming a multiplexed tool for a community of users, instead of being a batch tool for wizard programmers
IBM’s Response

• IBM bet the company on the System/360 hardware family [1964]
  • S/360 was the first to clearly distinguish architecture and implementation
  • Its architecture was virtualizable

• The CP/CMS system software [1968]
  • CP: a “control program” that created and managed virtual S/360 machines
  • CMS: the “Cambridge monitor system” -- a lightweight, single-user OS
  • With CP/CMS, can run several different OSs concurrently on the same HW

• IBM CP/CMS is the first virtualization system. Main purpose: multiple users can share a mainframe
IBM’s Mainframe Product Line

- System/360 (1964-1970)
  - Support virtualization via CP/CMS, channel I/O, virtual memory…
  - Reimplementation of CP/CMS as VM/370
- System/390 (1990-2000)
- zSeries (2000-present)

- Huge moneymaker for IBM, and many business still depend on these!
PCs and Multi-User OSes

• 1976: Steve Jobs and Steve Wozniak start Apple Computers and roll out the Apple I, the first computer with a single-circuit board

• 1981: The first IBM personal computer, code-named "Acorn," is introduced. It uses Microsoft's MS-DOS

• 1983: Apple's Lisa is the first personal computer with a GUI

• 1985: Microsoft announces Windows

• The PC market (1980-90s): ship hundreds of millions of units, not hundreds of units

• Cluster computing (1990s): build a cheap mainframe out of a cluster of PCs
Multiprocessor and Stanford FLASH

• Development of multiprocessor hardware boomed (1990s)

• Stanford FLASH Multiprocessor
  • A multiprocessor that integrates global cache coherence & msg passing

• But system software lagged behind

• Commodity OSes do not scale and cannot isolate/contain faults
A Real FLASH Multiprocessor

Can it run Windows?

Photo taken by Yiyong Zhang at Mendel Rosenblum's Office
Stanford Disco and VMware

- Stanford Disco project (SOSP’97 Mendel Rosenblum et al.)
  - Extend modern OS to run efficiently on shared memory multiprocessors
  - A VMM built to run multiple copies of Silicon Graphics IRIX OS on FLASH

- Mendel Rosenblum, Diane Greene, and others co-founded VMware in 1998
  - Brought virtualization to PCs. Main purpose: *run different OSes on different arch*
  - Initial market was software developers for testing software in multiple OSes
  - Acquired by EMC (2003), which later merged with DELL (2016)
Diane Greene

- Co-founder and the CEO of VMware from 1998 to 2008
- The CEO of Google's cloud businesses from 2015 until early 2019
- An Alphabet board of directors member from 2012 to 2019
- 2017 Abie Award for Technical Leadership from AnitaBorg
- 2018 Forbes' America's Top 50 Women In Tech
Server Consolidation

- Datacenters often run many services (e.g., search, mail server, database)
  - Easier to manage by running one service per machine
  - Leads to low resource utilization
- Virtualization can “consolidate” servers by hosting many VMs per machine, each running one service
  - Higher resource utilization while still delivering manageability
The Cloud Era

• The cloud revolution is what really took virtualization on

• Instead of renting physical machines, rent VMs
  • Better consolidation and resource utilization
  • Better portability and manageability
  • Easy to deploy and maintain software
  • However, raise certain security and QoS concerns

• Many instance types, some with specialized hardware; all well maintained and patched
  • AWS: 241 instance types in 30 families (as of Dec 2019)
The Virtuous Cycle for Cloud Providers

1. Lower prices attract more customers
2. More customers utilize more resources
3. Greater utilization of resources requires more infrastructure
4. Buying more infrastructure in volume leads to lower unit costs
5. Lower unit costs allow for lower customer prices

Source: CloudBlogs.Microsoft.com
Container

• VMs run a complete OS on emulated hardware
  • Too heavy-weighted and unnecessary for many cloud usages
  • Need to maintain OS versions, libraries, and make sure applications are compatible
• Containers (e.g., Docker, LXC)
  • Run multiple isolated user-space applications on the host OS
  • Much more lightweight: better runtime performance, less memory, faster startup
  • Easier to deploy and maintain applications
  • But doesn’t provide as strong security boundaries as VMs
Managing Containers

- Need a way to manage a cluster of containers
  - Handle failure, scheduling, monitoring, authentication, etc.
- Kubernetes: the most popular container orchestration today
- Cloud providers also offer various container orchestration service
  - e.g., AWS ECS, EKS
Serverless Computing

- VMs and containers in cloud still need to be “managed”

- Is there a way to just write software and let the cloud do all the rest?

- Serverless computing (mainly in the form of Function as a Service)
  - Autoscaled and billed by request load
  - No need to manage “server cluster” or handle failure
  - A lot less control and customization (e.g., fixed CPU/memory ratio, no direct communication across functions, no easy way to maintain states)
Summary of Virtualization History

• Invented by IBM in 1960s for sharing expensive mainframes
• Popular research ideas in 1960s and 1970s
• Interest died as the adoption of cheap PCs and multi-user OSes surged in 1980s
• A (somewhat accidental) research idea got transferred to VMware
• Real adoption happened with the growth of cloud computing
• New forms of virtualization: container and serverless, in the modern cloud era