Resource Virtualization

Syllabus

Web Page
http://www.cs.northwestern.edu/~pdinda/virt

Instructor
Peter A. Dinda
Technological Institute, Room L463
847-467-7859
pdinda@northwestern.edu
Office hours: Thursdays, 2-4pm or by appointment

Location and Time
Lecture: Ford Third Floor Conference Room (tentative)
or Systems Lab (SB-340)
Tuesdays and Thursdays, 12:30-2

Prerequisites
Required CS 213 or (ECE 205 and ECE 231)
or equivalent computer systems course
Highly recommended CS 343 or equivalent operating systems course
Highly recommended CS 340 or equivalent networking course
Highly recommended Familiarity with a systems programming language
such as C or C++. Familiarity with a scripting
language such as Perl or Python.
Highly recommended Familiarity with computer architecture, ideally to
the level of ECE 361. (CS 213 is the minimum
sufficient level)

The purpose behind these prerequisites is to assure that you understand the
principles of how processors, computer systems, and networks work coming into
the class and are able to do systems-level programming. *If you do not meet some
of these prerequisites, but feel you are prepared to take the course, please contact
me.*
Readings
For the most part, readings for the course will be in the form of research papers. A separate reading list is provided. **There is no textbook, but you should buy the following reference book:**


Objectives, framework, philosophy, and caveats
A basic principle in computer science is that of indirection, creating a new layer between two existing layers. By introducing a new layer into a software system, it often becomes straightforward to do many things and the system becomes more flexible. However, there is a tension: a new layer may make the system slower.

A classic example is adding a programming language to a system. For example, the EMACS text editor has a core data manipulation library written in C coupled with a Lisp interpreter, through which the library can be used. Much of the editor, and the many extensions people have added to it are written in Lisp, which is arguably much easier than writing them in C and sufficiently fast for text editing.

Currently, there is considerable excitement in the operating systems, networking, and distributed systems communities over what we shall call **resource virtualization**. The basic idea is to add a software layer that provides virtual machines, virtual networks, and even virtual services that are implemented on top of the existing physical resources and services in the network. Because these resources are virtual, we can potentially create a great many of them, make them private to their users, customize them to particular purposes, simplify their administration by making them user- or group-specific, and even inspect them from the outside to monitor their performance or detect intrusions.

This course will examine resource virtualization, from the highly influential early work in the 1970s to the present. In particular, it will include:

- Architectural support for virtualization
- Traditional OS-level virtualization
- Paravirtualization
- Virtual servers
- Emulation and binary translation
- Language-level virtualization
- Virtual networking and overlays
- Virtual devices
- Virtual storage and SANs
- Virtual services
- Virtual machine migration
• Remote display
• Virtualization-based computing environments
• Measurement, inference, adaptation and reservation
• Security and virtual machines

Almost all of the readings for the course will be in the form of research papers, with some experience report papers added as well. We will generally read 1-3 papers or equivalent materials for each session, covering fundamental ideas and important recent results. Each paper will be formally presented to the group by a student and then discussed in a round-table manner. A reading list will be available that includes the papers to be read, as well as other related papers.

This is a graduate course and all students in it will be treated like graduate students. I will assume that you are interested in this material, that you can motivate yourself to learn about it, and that you will not be afraid to venture into uncharted territory (i.e., do research). The undergraduate section will differ primarily in that the expectations for the project will be slightly lower.

Projects in the previous iterations of the course resulted in several papers published at high quality workshops and conferences.

Tools

Students in the course will be given account on the VLab. The VLab consists of a high end cluster that support virtual machines using VMware Virtual Server. You can find out more about the VLab at http://www.cs.northwestern.edu/~pdinda/vlab/.

Project

Over the course of the quarter, you will apply what you learn to a project of your choice, and then document your project in a high quality paper and open presentation. Project topics will be chosen in consultation with me. Projects may be done individually or in groups. Project complexity and expectations will be tied to group size. There are specific projects that I have in mind that would be well-geared to groups of two or three. I will expect weekly project reports.

The expectation for graduate students is that the project will be quality work that the students would not be embarrassed to submit to a workshop. The expectation for undergraduates is that the project be something they would be proud to list on their resumes, although all students are encouraged to aim high. There is a related long-term research project to this course, Virtuoso (more information at http://virtuoso.cs.northwestern.edu/), so there is a potential for projects in this course to turn into longer-term research efforts.

All projects will be presented at a public colloquium.
Example project ideas will be provided in a separate paper handout. Because of the high expectations placed on the project, it is vital that you choose to work on something that interests you deeply and that I can advise strongly.

Exams
There will be no exams

Grading
50% Project, including weekly progress reports
10% Project paper and presentation
20% In-class paper presentations of papers
20% General classroom participation

Schedule

Thursday, 1/4 Introductory Material
  What are virtual machines and what can we do with them?
  Reviewing how an operating system works
  Required Reading: Collection-Intro; Collection-Smith&Nair; Figueiredo03 (3)

Tuesday, 1/9 Traditional OS-level virtual machines
  Goldberg’s seminal work
  The VM/370 operating system
  Required Reading: Goldberg74 (21), Seawright79 (22), Creasy81 (23)

Thursday, 1/11 Architectural support for traditional OS-level VMs
  What hardware features are needed
  Why it used to be hard to virtualize an IA32 or IA64 machine
  Required Reading: Popek74 (46), Robin00 (48)
  PROJECT PROPOSAL DUE

Tuesday, 1/16 Modern architectural support for traditional OS-level VMs
  AMD’s Pacifica and Intel’s VT extensions
  Required Reading: Nieger06 (51), AMD (50)

Thursday, 1/18 Modern VMMs
  How something like VMware or Virtual PC works
  Required Reading: Collection-Rosenblum; Waldsburger29 (31), Dong06 (35)
  Other reading: Hall91 (47), Karger91 (27), Virtual 8086 mode
  PROJECT UPDATE DUE

Tuesday, 1/23 Paravirtualization
  Blurring the line between VMM and microkernel
  Very fast VMMs that you have to port your operating system to
  Xen, Denali, UML
Thursday, 1/25  
Virtual Servers  
Making one operating system look like many  
Required Reading: Linux V-Server (68), Kam00 (69), Price04 (70)

PROJECT UPDATE DUE

Tuesday, 1/30  
Emulation and binary translation  
Running on a different architecture  
Binary Translation, SIMICS, QEMU  
Required Reading: Sites93 (53), Magnusson02 (54), Bellard05 (57)

Thursday, 2/1  
Language-level virtual machines  
Make your own architecture  
UNCOL, p-System, JVM, Jit-compilers  
Required Reading: p-System (60), Lindholm (61), Meijer (63)  
Other reading: Adl-Tabatabai98 (62), Dinda04 (125)

PROJECT UPDATE DUE

Tuesday, 2/6  
Virtual devices  
Making peripherals remote, virtual, or both, while keeping them fast  
Required Reading: Sugarman01 (71), Menon06 (78), Liu06 (79)  
Other Reading: Abramson06 (80)

Thursday, 2/8  
Virtual storage and SANs  
Putting your disks in someone else’s care  
Required Reading: Wikipedia (85), Pfaff06 (86), Nath06 (87)  
Other Reading: Beck02 (82), Huang04 (84)

PROJECT UPDATE DUE

Tuesday, 2/13  
Virtual networking and overlays  
Building a new, better network on top of the old network  
Required Reading: Chu02 (97), Ferguson98 (91)  
Other reading: Walfish04 (103)

Thursday, 2/15  
Virtual networking and overlays  
Building a new, better network for virtual machines on top of the old network  
Required Reading: Sundararaj04 (101), Jiang03 (102), Ganguly06 (106)  
Other Reading: Tsugawa06 (107)

PROJECT UPDATE DUE

Tuesday, 2/20  
Remote display  
Delivering the console to the user  
Thin client computing  
Required Reading: Richardson98 (136), Baratto05 (140)
Other reading: Lai02 (139), Microsoft00 (138), Rossoff06 (141)

Thursday, 2/22  Migration
Delivering the computer to the user
Required Reading: Sapuntzakis02 (130), Clark05 (133), Nelson05 (134)
Other Reading: Kozuch02 (129)
PROJECT UPDATE DUE

Tuesday, 2/27  Virtualization-based computing environments
High-end and grid computing and Enterprise
Required Reading: Ganguly06 (18), Fabian06 (19), Ruth05 (13)
Other reading: Keahey05 (16), Maccabe05 (17), Shoykhet (9)

Thursday, 3/1  Virtualization-based computing environments
Desktop replacements and servers
Required Reading: Chandra05 (13), Satyanarayanan05 (15)
Other Reading: Garfinkel03 (7), Jiang03 (8), Sapuntzakis03 (6)
PROJECT UPDATE DUE

Tuesday, 3/6  Measurement, inference, adaptation, and reservation
Automagically making existing, unmodified application run better or cheaper
Required Reading: Sundararaj05 (110), Lange05 (111) Gupta06 (114)

Thursday, 3/8  Security and virtual machines
VMM isolation and codesize arguments
VM introspection
Required Reading: Karger91 (27), Garfinkel03 (124)
Other reading: Dunlop02 (123)

PROJECT DUE

Project Presentations In Finals Week