Resource Virtualization

Syllabus

Web Page

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

Instructor

Peter A. Dinda
1890 Maple Avenue, Room 338
847-467-7859
pdinda@cs.northwestern.edu
Office hours: Thursdays, 2-4pm or by appointment

Teaching assistants

Jason Skicewicz
1890 Maple Avenue, Room 332
847-491-7150
jskitz@cs.northwestern.edu
Office hours:

Location and Time

1890 Maple Avenue, Bliss Conference Room (340),
Mondays and Wednesdays, 2 pm.

Prerequisites

Required	CS 343 or equivalent operating systems course
Required	CS 340 or equivalent networking course
Highly recommended	CS 213 or equivalent computer systems course
Highly recommended	Familiarity with a systems programming language such as C or C++. Familiarity with a scripting language such as Perl or Python.
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

Readings for the course will be in the form of research papers. A separate reading list will be provided.

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 (as in IBM's 370 architecture and soon to be included in the Intel architecture, also HW/SW co-designed VMs, such as Crusoe.)
- OS-level virtualization (examples include IBM's VM operating system, VMWare, Plex/86, Microsoft's Virtual Server, Virtual PC, etc)
- Virtual servers (examples include Ensim, Virtuozzo, Free VSD, V-server, UML, Xeno, etc)
- Emulation (examples include Virtual PC on a Mac, or SIMICS)
- Language-level virtualization (examples include UNCOL, the UCSD p-system, the Java VM, and Microsoft's .NET VM)
- Virtual networking (VLANs, VPNs, overlays for virtual machines at layers 2 and 3)
- Virtualized services (Proxies, etc) and storage (SANs, etc)
- Distributed computing using resource virtualization (Grid computing, assorted current research projects, including one of ours)

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.

This is the first iteration of this course.

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://www.cs.northwestern.edu/~plab/Virtuoso/), 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 are listed in a separate 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

Session	Date	Topics	Reading
1	1/5	Introductions, class mechanics schedule reading for 1st half of quarter. Kinds of virtual resources (machines (architectural-, language-, os-level), devices, networking). Services built around them *Smith's Taxonomy and our variants*	Smith01 (2)
2	1/7	Why OS-level Virtual Machines? (1970s) *Goldberg summary of SotA Note: IBM VM/370 has whole lecture later*	Goldberg74 (1)
3	1/12	Why Language-level Virtual Machines (1960s-Today) *UNCOL to CLR*	Bagly62 (13), Vogels03 (15)
4	1/14	Why Virtual Machines (1980s) *Golden Age of DOS multiplexing Also, VAX security kernel.*	VMM/386 (3), V8086 (4), Karger91 (5)
5	1/19 (MLK)	Why Virtual Machines (1990s-today) *VMWare, Virtual Appliances, Secure computing platforms*	Nieh(11), Sapuntzakis03 (9), Garfinkel03 (10)
6	1/21	Why Virtual Machines (1990s-today) *Virtual grid computing, Virtual Services, Security*	Figueiredo03 (6), Hand03 (7), Chen01 (8)
7	1/26	Emulation *SIMICS and classic alpha binary translation work*	Magnussen02 (32), Sites93 (34)
8	1/28	OS-level Virtual Machines *IBM VM/370: The most successful virtual machine system ever*	Seawright79 (17), Creasy81 (18) and other VM/370 papers
9	2/2	OS-level Virtual Machines *VMWare patent, Lawton discussion about virtualizing the IA32 (may change this), the Disco paper, and another analysis of the IA32's VMM capabilities (may change this)*	Devine02 (22), Lawton (24), Bugnion00 (26), Robin (30)
10	2/4	OS-level Virtual Machines	Devine02 (22), Lawton (24),

		VMWare patent, Lawton discussion about virtualizing the IA32 (may change this), the Disco paper, and another analysis of the IA32's VMM capabilities (may change this) (continuation)	Bugnion00 (26), Robin (30)
11	2/9	Virtual Servers *Iimplementations of virtual servers, focusing on UML (May also focus this on VSD or V-server)*	implementations (Ensim, FreeVSD, Linux V-server and UML (Dike00 (49), Hoxer (50))
12	2/11	Virtual Servers *Denali and Xeno*	Whitaker02 (42), Barham03 (47)
13	2/16	Language-level Virtual Machines *UCSD p-system, Java VM Spec (Read the intro, SKIM the book)*	UCSD p-system (37), Lindholm (38)
14	2/18	Language-level Virtual Machines *MS CLR, JITS*	Meijer (39), MS (40), Adl-Tabatabai98 (41)
15	2/23	Virtual Devices *VMWare device virtualization*	Sugarman01 (51), King03 (52)
16	2/25	Virtual Networking *VPNs and VLANs* *We'll try to find a better thing for you to read about VLANs.*	Ferguson98 (61), Italiano02 (63), IEEE (64)
17	3/1	Virtual Networking *Several of virtual machine specific proposals (for virtual servers and OS-level virtual machines)*	Sundararaj03 (67), Jiang03 (68), Jinag03 (59)
18	3/3	Virtual Storage *Loki, Façade, planning a SAN*	Beck02 (55), Lumb03 (56), maybe Alvarez01 (54)
19	3/8	Migration *OS-level and virtual server migration, We may also briefly talk about migration in language-level VMs*	Sapuntzakis02 (74), Boyd02 (76), maybe Osman02 (75)
20	3/10	Security *IDS based on introspection, Using VMs to replay attacks*	Garfinkel03 (72), Dunlop02 (73)