Syllabus:  
Total virtualization

**Course name:** IT Product Development

**Code discipline:** ????

**Subject area:** Software Engineering

## P.1 Short Description and prerequisites

### Virtualization has wide practical applicability nowadays. A shift from local desktop applications towards software/desktop as a service is a paradigm shift. Scale-out instead of scale-up is another trend served by virtualization technologies. Clouds are everywhere while the fog computing is coming. As the epoch of personal computers passes, we see the dawn of new era.

### This era will be populated with a wide variety of devices and virtualized applications. More and more services would choose microservices model. And while many app developers and home users don’t understand virtualization, professionals who would use or develop a cloud solution must have the deep knowledge of cloud and virtualization technologies. That is the basics for an argued choice of a particular solution within a given business context. That is the key for «leaky abstractions» problem solving.

### Additionally, virtualization is one of the more complicated technologies that are in practical use today. Solving problems in this domain would train good problem solving skill. In particular, it is important to understand how lower-level issues can impact those experienced at a higher level. For example, a slow-starting Office application might be caused by host operating system file caching algorithms. Understanding this interplay requires students to switch their point of view quickly from low-level details to the larger picture -- a skill that is a key success factor for an IT professional.

### The Total Virtualization course will be interesting for aspiring software architects, since it promotes a system-wide perspective (a must-have skill for a software architect). The course uncovers numerous aspects of cloud architecture that should be considered while choosing a project development path (e.g., the risks and benefits of cloud infrastructure usage, the choice of a particular cloud or virtualization solution, virtual infrastructure management issues, etc.). These aspects could also be helpful for future CTOs. The course is also aimed at system-level developers, since it gives opportunity for integrating system-level knowledge (CPU architecture, OS architecture, resource management algorithms) into popular applied technologies. The course is also essential for anyone who is planning to develop an app in the cloud (e.g., considering issues such as the price of migration between clouds, cloud risks, virtualization overhead, etc.). Although some of the technologies described in the course may be of less interest to application developers, it is critical to be aware of the leaky abstractions problem: without understanding the underlying virtualization stack some application-level problems can’t be fixed at all

### P.1.1 Prerequisite subject codes from [the li](https://docs.google.com/spreadsheets/d/1LGrcNu_mAeOr9K2QehDgHSVrRL5P2IcaGFOvQqEaGUE/edit#gid=0)

* CSE101
* СSE105
* CSE112 or CSE118 or CSE420
* CSE130 or CSE512 or CSE128
* 22 or CSE804 or CSE809 or CSE812

### P.1.2 Prerequisite topics

* Basic programming skills.
* Software architecture
* Computer architecture
* Basics of Operating systems

## P.2 Course Topics

Going deeper, the course will cover the following topics:

* The need for virtualization: clouds, IaaS, DaaS, SaaS, PaaS.
* A virtualization introduction: basic terms, history
* CPU virtualization approaches: emulation, binary recompilation, stabbing, para-virtualization, hardware-assisted technologies
* Physical memory virtualization issues: compression, swap files, balloons
* Virtual paging, interrupts and devices, virtual video
* VM performance
* VM migration
* Introduction to containers
* Dockers
* IaaS: Resource distribution (memory, CPU)
* IaaS: Dynamic power management
* IaaS: Failovers and backups
* PaaS: a view from inside
* PaaS: a revolution driven by microservices
* Migrating to the cloud: cost, risks and deployment models, choice of cloud providers
* Security issues of clouds and virtualisation
* Cloud storages

## P.3 Intended Learning Outcomes (ILOs)

### P.3.1 What is the main purpose of this course?

The course is aimed to give vision of the full virtualization/cloud stack: from processor registers to law risks of migrations, from performance of virtual machines to security of public clouds. This vision would allow students to have high competence in developing their cloud solutions or even participating in cloud platforms.

### P.3.2 ILOs defined at three levels

**Level 1:** What **concepts** should a student **know/remember/explain?**

By the end of the course, the students should be able to ...

* Differ PaaS from IaaS and SaaS
* Explain the basic principles of virtualization
* Understand difference between containers and virtual machines
* Know criteria of choice between containers and virtual machines for different workloads
* Know memory management strategies and techniques
* Remember diffetent virtualization approaches
* Understand internals of PaaS and argue what is SaaS and what is PaaS

**Level 2:** What **basic practical skills** should a student be able to perform?

By the end of the course, the students should be able to ...

* Read and understand the source code of Linux kernel
* Create a container based on cgroups interface
* Deploy an application into the cloud infrastructure
* Use PaaS API for debug and monitoring
* Tune performance of a virtual machine
* Estimate cost of public IaaS use vs on-premise deployment

**Level 3:** What **complex comprehensive skills** should a student be able to **apply in real-life scenarios?**

By the end of the course, the students should be able to ...

* Find real technical information under tones of marketing stuff
* Make weighted and reasonable architectural choices in virtualization and cloud domain

## P.4 Grading

**Table 1:** Course grading range

|  |  |  |
| --- | --- | --- |
| **Grade** | **Range** | **Description of performance (optional)** |
| A. Excellent | 80-100 |  |
| B. Good | 60-79 |  |
| C. Satisfactory | 40-59 |  |
| D. Fail | 0-39 |  |

**Table 2:** Course activities and grading breakdown

|  |  |
| --- | --- |
| **Activity Type** | **Percentage of the overall course grade** |
| Assignment/Labs | 70 |
| Final quiz | 30 |

### P.4.1 Recommendations for students on how to succeed in the course (optional)

* Participation is important. Showing up is the key to success in this course.
* If you don’t have a corresponding technical background, please do not hesitate to ask lecturer. If you feel that the gap is deep, request for extra reading.
* Reading the recommended literature is optional, and will give you a deeper understanding of the material.

## P.5 Resources, literature and reference materials

### P.5.1 Open access resources

* Timekeeping in VMware virtual machines <http://www.vmware.com/files/pdf/Timekeeping-In-VirtualMachines.pdf>
* Intel Software Developer Manual, volume 3c, charter 23 <http://www.intel.com/content/dam/www/public/us/en/documents/manuals/64-ia-32-architectures-software-developer-vol-3c-part-3-manual.pdf>
* Carl A. Waldspurger. “Memory Resource Management in VMware ESX Server” <http://www.waldspurger.org/carl/papers/esx-mem-osdi02.pdf>
* Carl A. Waldspurger et al, “VMware distributed resource management: design, implementation and lessons learned” <http://www.waldspurger.org/carl/papers/drs-vmtj-mar12.pdf>
* Ian Pratt et al, “Xen and the art of virtualization”, <http://www.cl.cam.ac.uk/research/srg/netos/papers/2003-xensosp.pdf>
* Yoshi Tamura, “Kemari: the fault tolerant VM synchronization based on KVM”, <http://www.linux-kvm.org/wiki/images/0/0d/0.5.kemari-kvm-forum-2010.pdf>
* “Increasing virtual machine density with KSM”, <http://www.linux-kvm.org/wiki/images/0/0e/KvmForum2008$kdf2008_12.pdf>
* Microsoft Hyper-V overview <http://technet.microsoft.com/library/hh831531.aspx>
* OpenStack architecture <http://docs.openstack.org/arch-design/arch-design.pdf>
* Understanding the cloud computing stack <http://broadcast.rackspace.com/hosting_knowledge/whitepapers/Understanding-the-Cloud-Computing-Stack.pdf>
* “Analysis of the Intel Pentium’s Ability to Support a Secure Virtual Machine Monitor” <https://www.usenix.org/conference/9th-usenix-security-symposium/analysis-intel-pentiums-ability-support-secure-virtual>
* <https://docs.openstack.org/yoga/api/>
* <https://azure.microsoft.com/en-us/blog/inside-azure-file-storage/>
* <https://opencontainers.org/>
* <https://www.virtuozzo.com/fileadmin/user_upload/downloads/White_Papers/Virtuozzo-VMvsContainerSecurity-WP-EN-Ltr.pdf>
* <https://aws.amazon.com/blogs/aws/firecracker-lightweight-virtualization-for-serverless-computing/>
* <https://resources.coreos.com/white-papers/6-pitfalls-of-do-it-yourself-kubernetes>
* <https://www.blackhat.com/docs/us-16/materials/us-16-Sharkey-Breaking-Hardware-Enforced-Security-With-Hypervisors.pdf>

### P.5.2 Closed access resources

* Carl Waldspurger, Mendel Rosenblum, “I/O virtualization”, Communications of ACM, 2012

### P.5.3 Software and tools used within the course

* Under discussion.

Teaching Methodology:  
Methods, techniques, & activities

## P6. Activities and Teaching Methods by Sections

Mark what techniques and methods are used in each section (1 is used, 0 is not used).

**Table A1:** Teaching and Learning Methods within each section

|  |  |  |
| --- | --- | --- |
| **Teaching Techniques** | **Section 1** | **Section 2** |
| Problem-based learning (students learn by solving open-ended problems without a strictly-defined solution) | 1 | 1 |
| Project-based learning (students work on a project) | 1 | 1 |
| Modular learning (facilitated self-study) | 1 | 1 |
| Differentiated learning (provide tasks and activities at several levels of difficulty to fit students needs and level) | 0 | 0 |
| Contextual learning (activities and tasks are connected to the real world to make it easier for students to relate to them); | 1 | 1 |
| Business game (learn by playing a game that incorporates the principles of the material covered within the course). | 0 | 1 |
| inquiry-based learning | 0 | 0 |
| Just-in-time teaching | 0 | 0 |
| Process oriented guided inquiry learning (POGIL) | 0 | 0 |
| Studio-based learning | 0 | 0 |
| Universal design for learning, | 0 | 0 |
| Task-based learning | 0 | 0 |

**Table A2:** Activities within each section

|  |  |  |
| --- | --- | --- |
| **Learning Activities** | **Section 1** | **Section 2** |
| Lectures | 1 | 1 |
| Interactive Lectures | 1 | 1 |
| Lab exercises | 1 | 1 |
| Experiments | 0 | 0 |
| Modeling | 0 | 0 |
| Cases studies | 0 | 1 |
| Individual Projects | 0 | 0 |
| Group projects | 0 | 0 |
| Flipped classroom | 0 | 0 |
| Quizzes (written or computer based) | 0 | 1 |
| Peer Review | 0 | 0 |
| Discussions | 1 | 1 |
| Presentations by students | 0 | 0 |
| Written reports | 1 | 1 |
| Simulations and role-plays | 0 | 1 |
| Essays | 0 | 0 |
| Oral Reports | 0 | 0 |

## P.7 Formative Assessment and Course Activities

### P.7.1 Ongoing performance assessment

Both graded and non-graded activities during the semester (before the exam)

|  |  |  |
| --- | --- | --- |
| **Activity Type** | **Content** | **Is Graded?** |
| Individual Assignments | A1: Find the response in Linux kernel source code? Find the VT-x virtualization loop. Specify related functions  Find the PF handler | 1 |

### P.7.2 Final assessment

|  |  |  |
| --- | --- | --- |
| **Activity Type** | **Content** | **Is Graded?** |
| Quiz | 1. Which type of virtualization would you recommend for the following workload?  2. Why SMP workload could be slow in virtual machines?  3. Which type of vulnerability is more “popular” for virtual machines then for containers? | 1 |