https://folk.ntnu.no/michaeng/tdt4186_22/
michael.engel@ntnu.no

Theoretical exercises
Spring 2022

Theoretical Exercises 8

Virtual machines, the Cloud and Embedded Systems

Please submit solutions on Blackboard by Monday, 4.4.2022 12:00h

Notice: Finding solutions to most of the questions below requires research beyond the material given in lectures 17 and 18!

8.1 Emulation and JIT compilation

With Apple's M1 CPUs, the processor architecture of Macs was switched from x86-64 to Aarch64 (commonly called ARM64). The instruction sets of these two architectures are not compatible, so there are two ways to keep old software running:

- Recompile all programs for the Aarch64 target instruction set
- Translate the compiled binary programs from x86-64 machine code to Aarch64

Apple provides such a translator with macOS, the Rosetta 2 system. Rosetta 2 is a combination of static translation (translating parts of the binary program, like translating a text from Norwegian to English, without executing it) and dynamic just-in-time compilation (tracing the execution of the program and translating the program piece by piece as it is executed).
Find out why, in the common case, it is *not* possible to *statically* translate a binary executable from one instruction set to the other. In other words, why is a certain amount of interpretation or just-in-time compilation of the original program at runtime required?
Hint: This is a very advanced question that requires knowledge about computer architecture and machine language. Finding out the answer will teach you more about the interaction between a binary program code, its data, and the hardware, than you ever wanted to know...

8.2 Virtual memory ballooning

When running multiple operating systems on top of a hypervisor, it is possible to configure *memory ballooning*, a technique that allows to allocate more memory for all VMs taken together than what is available as RAM in the computer running the hypervisor.
Assuming no swap space is configured, explain why the memory ballooning approach can work.
Hint: look at the memory use of a Linux or MacOS system using e.g. `htop`.

8.3 Containers

The earliest support for a very simple container-like functionality in Unix was provided by the `chroot(2)` system call and the Unix tool with the same name.
Give three examples of missing functionality in `chroot` to implement container functionality as provided by Docker (you might need to read up on Docker) and explain why the respective functionality is required (e.g. in terms of security).

8.4 Popek and Goldberg Criteria

The original Motorola 68000 processor did not fulfill the Popek and Goldberg criteria for a virtualizable architecture.
The critical instruction that could be executed in the nonprivileged mode (user mode) is
`MOVE from SR`
This instruction copies the contents of processor status register into a regular data register.
Find out why this instruction is problematic and how a virtual machine could exploit this instruction.

8.5 Privilege Modes

Some processor architectures (e.g. Intel/AMD x86 and x86-64, DEC VAX and RISC-V) support more than two privilege levels.
Find out why this could be useful and in which ways more than two privilege levels were used in the Windows NT 3.5 and VMS operating system.

8.6 Shadow Page Tables

Describe what happens in case of a page fault in a virtual machine on a system that employs a hypervisor using shadow page tables (see slide 19 of lecture 18 for an overview). Which lookups are performed, which parts of which of the page tables are accessed?