HÁSKÓLI ÍSLANDS
Iðnaðarverkfræði-, vélaverkfræði- og tölvunarfræðideild

TÖL401G: Stýrikerfi / Operating Systems · Vormisseri 2022

Assignments 3–4 · To be solved until 01.02.2022, 13:00

Note: choose *either* Assignment 3a *or* 3b: Assignment 3a, if you prefer theoretical work; Assignment 3b, if you prefer practical work. have a fast Internet connection and 3 GB space left on your disk in order to download/install a virtual machine running Linux as guest OS.

Examples from each chapter will be explained in videos that are available in Canvas via "Panopto Video". On Thursday you will have opportunity to work on assignment with assistant from teacher. Report via Canvas at any questions that you have, so that we can clarify this during class.

Assignment 3a

Consider a VM hypervisor (see section 2.8 of the lecture slides) for running operating systems OS_1 and OS_2 on top of it inside two virtual machines on a single processor/single core system:

1. The scheduler of the VM hypervisor has just granted OS_1 CPU time for 20ms. While OS_1 is running in the granted time slot, the timer of OS_2 for triggering the CPU scheduler of OS_2 expires. Describe two different options of how the VM hypervisor could deal with this!

   Note: Due to the assumed single processor/single core system, OS_2 is suspended while the VM hypervisor is executing OS_1. Also assume that a native hypervisor and no para-virtualisation are used and therefore, the guest OSes do not know that they are running in a VM nor does the hypervisor know internals of the guest OSes.
   Do not discuss CPU scheduling algorithms in general, but consider just two different options for the particular scenario above (=what to do if an external interrupt occurs which is intended to be serviced by a VM which has currently not the CPU).

2. In a virtual machine scenario, the different virtual machines are really completely separated from each other: OS_1 running on VM_1 can neither access the main memory nor the files[1] of OS_2 running on VM_2. – Describe one possible solution that allows you nevertheless to exchange data between different virtual machines.

Assignment 3b

Install and use the hosted hypervisor *VirtualBox* as follows to answer the questions below.

- If possible, enable CPU virtualisation support in your BIOS/UEFI (may be hard to find, e.g. hidden in Security settings) – in particular if VirtualBox gives you an error message that "VT-x is not available".
- Download (Windows, Mac OS, Linux) and install via `https://www.virtualbox.org`

[1] To virtualise a disk that provides a file system, typically a file (as huge as the virtual disk) is created on the host file system as replacement for a physical disk. Each VM then uses a different file ("VM image") from the host file system as virtual disk.

- If you are using Linux as host OS: you need to have kernel sources installed (do a web search using your Linux distribution name + "virtualbox install kernel sources" to find out how to achieve this). Note that your Linux distribution may anyway have a VirtualBox package that works out of the box (but might be outdated, which should however not be a problem).

- Finally, follow the steps described here:
  Note: VM image (=a virtual partition that contains a Linux installation to be ran inside a VM), namely the file `OSC-2016.ova`

  – Download this 2.7 GB file from
  `http://notendur.hi.is/helmut/OSC-2016.ova`

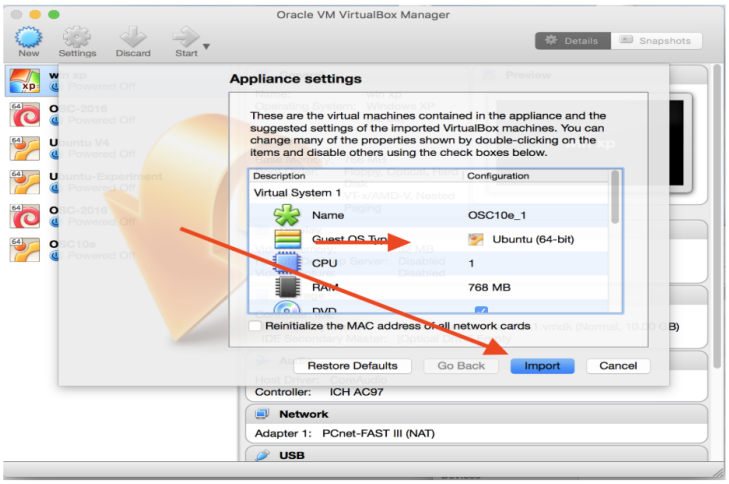
  To check the integrity of the download: calculate the MD5 checksum of your download, it needs to start with `93` and end with `84` (do not change filename of VM image):

  – on Linux: `md5sum "`*local path to OSC-2016.ova*`"`
  – on Mac: `md5 "`*local path to OSC-2016.ova*`"`
  – on Windows: `CertUtil -hashfile "`*local path to OSC-2016.ova*`" MD5`

  To import that file to VirtualBox, either double-click on it or start VirtualBox and use the "Import" icon or "File" → "Import Appliance" in VirtualBox (even though only `OVF` format is mentioned there, `OVA` is supported as well.)

1 Installation as a VirtualBox Appliance

Double-click on the downloaded file. This will open the following window



Choose the Import button.

Figure 1: An image of Oracle Virtual Box Manager

- On the Virtual Box Manager choose the OSC2016 VM and click the green `Start` button.

- For logging into the VM machine the

  1. Linux User id is: `oscreader`
  2. Linux password is: `osc`

  The Linux installation in the VM image uses US, keyboard layout, e.g.:/ is next to your right shift key, where þ is on an Icelandic keyboard.

1. While you should not download and run unknown software, in particular not OSes, from untrusted sources, why is it not a problem to download and run the VM image for solving this assignment? (Think about possible threats such as code contained in the VM images that might format your disk or encrypt the data stored on your host file system and give you the decryption key only after paying some ransom.)

2. Create CPU load inside the guest OS, e.g. by opening a command line shell and running on the command line: `yes >/dev/null` (stop later using Ctrl-C): describe briefly how this CPU load affects your host system in terms of change of the CPU load as displayed by some system monitor running on the host system (e.g. `top` on Linux or Mac OS command line or the MS Windows task manager)?

3. Now, do the opposite by creating load on your host system: e.g., using `yes >/dev/null` on Linux or Mac OS command line – for MS Windows, you can create a file that contains the following lines, save it into, e.g., your home directory in a file of type `.BAT`, and execute that file:
   ```
   @echo off
   :loop
   goto loop
   ```
   Describe briefly how this CPU load affects your guest system in terms of change of the CPU load as displayed by, e.g., the command line system monitor `top` on the guest system.

4. Are the effects reported by the system monitors the same in the two above cases? If not, how can the different monitoring results be explained?

Assignment 4

Describe the steps of the system boot process until the login prompt of the operating system appears. While slide 2-51 to 2-53 focuses mainly on the first half of system boot, you should focus on what happens after the bootstrap program/boot code has loaded the kernel into RAM and has just handed over control to the kernel. Slide 2-53 describes this only in three sub bullet points, you shall do it in more detail!

- Assume that the loaded kernel has a design based on *modules*!

  Note: this leads to another bootstrapping problem: for loading the modules from a storage device, a file system module and a device driver module are needed. However these needed modules are also located on a storage device and can thus not be accessed by the plain kernel. (Chicken-egg problem: the kernel needs a device driver module to access the device, but that device driver module is located on that device.) **Take care to cover the solution to this problem!**

- You can stop your description at the point when the operating system has decided which program to start as first process (with process id 1 – this is the process that is in charge of taking care that the login prompt is displayed). **Describe this final step as well, i.e. how is it decided what program to start as first process.**
- As the boot process is better described for open-source operating systems like Linux, describe the boot process of a Linux-based operating system. While doing some research on the net you will find a lot of descriptions, e.g., on web pages. **As always: mention your sources!**