Verkefni 2

brj46

vor 2025

Verkefni 3a

Spurning 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!

Svar:

The VM hypervisor could either:

1. Delay the handling of the interrupt from OS_2 until OS_1 has finished its time slot. once OS_1 has finished its time slot, the hypervisor can handle the interrupt from OS_2 and grant it CPU time.

   This option ensures that OS_1 gets its full time slot but OS_2 will experience a delay.

2. Pause OS_1 and immediately switch to OS_2 to handle the interrupt. After OS_2 has finished its interrupt, the hypervisor can resume OS_1 or continue with OS_2 depending on the scheduling policy.

   This option ensures that OS_2 gets its interrupt handled in a timely manner but OS_1 will experience an interruption.

Spurning 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 files1 of OS_2 running on VM_2. – Describe one possible solution that allows you nevertheless to exchange data between different virtual machines

Svar:

A possible solution to exchange data between different virtual machines is to use a shared file system.

The hypervisor could provide a shared virtual disk or file system that both virtual machines can access. This shared storage would act as a medium for data exchange between the virtual machines.

Verkefni 3b

Spurning 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?

Svar:

Well this software is provided by the university and is part of the course material. The main concern with running unknown software is the risk of malicious code, such as viruses or ransomware, which could harm your host system. However, since the VM image is provided by a trusted source, the risk is minimal.

Spurning 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)?

Svar:

When running the command `yes >/dev/null` in the guest OS, the CPU load on the host system increased as I saw in my task manager. but only from around 30% to around 41%.

Spurning 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.

Svar:

The CPU usage in the guest OS increased because the host system was consuming CPU resources, leaving fewer resources available for the VM. When I created CPU load on the host system using the .Bat file, the host system's CPU usage increased significantly. As a result, the guest OS had less access to the CPU resources, causing its CPU usage to increase as well.

Spurning 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

Svar:

The effects are similar but not the same.

1. When you create CPU load in the guest OS the host system's CPU usage increases because the VM is consuming CPU resources allocated to it by the host.
2. When you create CPU load on the host system the guest OS's CPU usage increases because the host system is consuming CPU resources, leaving fewer resources available for the VM.

The differences in the monitoring results can be explained by how VirtualBox allocates CPU resources between the host and guest systems. The host system has direct control over the physical CPU and dynamically allocates resources to the VM. When the host system is under heavy load, it prioritizes its own processes, reducing the resources available to the VM.