# OpSys: Oblig #3


120912 – Victor Rudolfsson

# Contents

*Chapter 11.6: Theory*

## 11.6.1. Hva er forskjellen på memory mapped og isolated I/O? Angi fordeler og ulemper med disse to prinsippene.

Isolated I/O means the device is accessed through a port, which makes the information from that device isolated to the assigned port. With memory mapped I/O, information from the I/O device is mapped to memory, and for data to be cached – this does, however, require a portion of the memory to be allocated for I/O. Memory mapped I/O can also lead to some issues with caching, such as problems arising from caching of stateflags.

## 11.6.2. På en harddisk, hvor mange bytes finnes som regel i en sektor? Hva er en sylinder? Hva er typisk gjennomsnittlig aksesstid for en disk i dag? Hva er overføringsraten (ca, i MB/s) mellom diskplate og buffer?

A sector generally consists of 512 bytes, and a cylinder is a collection of sectors consisting of a sector and all sectors directly below or above it vertically on a harddrive.

## 11.6.3. Hva oppnår vi med å koble diskene som RAID disker? Hvordan er diskene organisert på RAID-level 1. Forklar hvordan diskene er organisert på RAID-level 5.

By running harddrives in RAID, we get a more reliable storage medium. In RAID level 1, this is achieved by using two disks, in which one acts as a mirror. In RAID level 5, data is distributed over several disks, and it can operate even if one disk is missing. By distributing the parity information over all disks, RAID 5 can calculate the missing pieces from the parity information stored on the other disks.

## 11.6.4. Hvilke fire kriterier definerer et presist interrupt?

**1.** The program counter is stored in a known location
**2**. All instructions leading up to the program counter location have been executed
**3**. No instructions following the program counter has been executed
**4**. The state of the PC-instruction is known

## 11.6.5. Forklar forskjellen mellom HDD og SSD når det gjelder lesing, skriving/overskriving og sletting av filer. Hva er poenget med TRIM kommandoen?

A regular harddrive prefers reading *sequentially*. This is because bits are located on a rotating disk, and read by a mechanical arm. If bits are stored in sequence, it requires less time to locate all bits

required to complete the read (or write) instruction. A solid state drive, however, has no mechanical parts, and in contrast to a regular harddrive, prefers bits to be as evenly distributed as possible. This is because a solid state drive has multiple *channels* able to read from different places *simultaneously*, and retrieving bits from different locations is thus faster than if all bits were located in the same place. However, a solid state drive cannot write less than one page at the time, nor can it delete less than one block at the time. This means that to remove data from one page, the whole block must be deleted as a result of the higher voltage needed to perform an erasure.

TRIM allows the OS to tell a solid state drive which data blocks are no longer in use and can be removed.

### 11.6.6. Hva betyr det at et operativsystem er tilpasset SSD disker (slik som f.eks. Windows 7 er).

It supports TRIM.

## *Chapter 11: Labs*

```
$stats = get-wmiobject -class "Win32_PerfFormattedData_PerfOS_System" -computer .


function whoami

{

  $file = Get-ChildItem $MyInvocation.ScriptName

  write-host "Hi I'm" $file.Name

}


function uptime

{

    $lastBootTime = (Get-WmiObject -Class Win32_OperatingSystem -computername
localhost).LastBootUpTime

    $systemUpTime = (Get-Date) -
[System.Management.ManagementDateTimeconverter]::ToDateTime($lastBootTime)


    write-host "Uptime since last boot: " $systemUpTime.days "days " $systemUpTime.hours
"hours " $systemUpTime.minutes "minutes" $systemUpTime.seconds "seconds"

}


function nProcThreads

{

    $processes = get-process | measure

    $threads = get-process | %{$_.Name;$_.Threads | %{"`t{0}" -f $_.ID}} | measure


    write-host "Currently," $processes.Count "processes running with" $threads.Count
"threads"

}


function nContextSwitchesLastSec

{

    write-host "Context switches per second:"
$stats.ContextSwitchesPersec.tostring("###,##.0.0")

}


function cpuModeUsage

{

    # Okay seriously this is just ridiculous -- counters have _language specific_ names?

    # Apparently, since I made the mistake of installing Winblow$ in my native language,

    # counters aren't named in english. What incompetent software clown came up with this
craptastic idea?

    $privilegedTime = $($(Get-Counter -Counter "\Processor(_Total)\privilegierad tid i
```

```
procent" -SampleInterval 1 -MaxSamples 1).countersamples).cookedvalue

    $userTime = $($(Get-Counter -Counter "\Processor(_Total)\användartid i procent"
-SampleInterval 1 -MaxSamples 1).countersamples).cookedvalue

    #$privilegedTime = $($(Get-Counter -Counter "\Processor(_Total)\%Privileged Time"
-SampleInterval 1 -MaxSamples 1).countersamples).cookedvalue

    #$userTime = $($(Get-Counter -Counter "\Processor(_Total)\%User Time" -SampleInterval 1
-MaxSamples 1).countersamples).cookedvalue

    write-host $privilegedTime "% of time in kernel mode"

    write-host $userTime "% of time in user mode"

}


function nInterruptsLastSec

{

    # write-host $($(Get-Counter -Counter "\Processor(_Total)\interrupts/sec"
-SampleInterval 1 -MaxSamples 1).countersamples).cookedvalue "interrupts per second"

    write-host $($(Get-Counter -Counter "\Processor(_Total)\avbrott per sekund"
-SampleInterval 1 -MaxSamples 1).countersamples).cookedvalue "interrupts per second"

}






# Set up array with all options

$menuOptions = @("What's the name of this script?", "How long since last boot?", "How many
threads and processes are running?", "How many context switches per second?", "How much time
has the CPU spent in kernel/user mode per second?","How many interrupts per second?",
"Exit")

# And keep count for listing

$menuCount = 1


write-host "What I don't understand is why anybody who misses the terminal on Windows so
much they use PowerShell, is still using Windows?"


# List all options in array

foreach($menuOption in $menuOptions)

{

    write-host $menuCount "-" $menuOption


    # And increase count. Increase by two if its the last one.

    $menuCount++

    if($menuCount -eq $menuOptions.Count)

    {

        $menuCount = $menuCount + 2

    }

}
```

```
switch(read-host "Velg en funksjon: ")
{
    1{ whoami }
    2{ uptime }
    3{ nProcThreads }
    4{ nContextSwitchesLastSec }
    5{ cpuModeUsage }
    6{ nInterruptsLastSec }
    9{ exit }
}
```

*Chapter 12.3: Theory*

## 12.3.1. Det kreves at fire ulike betingelser alle må være oppfylt for at deadlock skal inntre. Beskriv disse kort.

**1. Mutual exclusion**: Each resource is assigned to *one* process, or available

**2. Hold and wai**t: The process which claimed a resource can request access to other resources

**3. No preemption**: Resources cannot be forcibly claimed but must be released first

**4. Circular wait**: Two or more processes is waiting for a resource held by one of the others which cannot be released until said resource is available.

## 12.3.1. Hva vil det si at en tilstand er "unsafe" i forbindelse med deadlock? Vil det garantert tilsi at en får en deadlock?

An *unsafe* state does not guarantee deadlock; however, it does mean that there is a *risk* of deadlock. In a guaranteed state, the process is not in deadlock and there's a guarantee that they will finish with their maximal resource requirements – a guarantee not present in an unsafe state.

## 12.3.3. Forklar kort hva en ressursgraf er og hvordan den kan benyttes til å avsløre om vi har en deadlock situasjon.

A resource graph plots the resources required by different processes in relation to one another, and a deadlock situation can be identified by a loop in the graph.

## 12.3.4. Tanenbaum oppgave 6.1 : Students working at individual PCs in a computer laboratory send their files to be printed by a server which spools the files on its hard disk. Under what conditions may a deadlock occur if the disk space for the print spool is limited? How may deadlock be avoided?

If both PCs' processes write to the directory and the disk becomes full, in which case disk space can only be freed up by either PCs' process, they will end up deadlocked as the printer will not print until the job is buffered on the disk in its entirety.

## 12.3.5. A computer has six tape drives, with *n* processes competing for them. Each process may need two drives. For which values of *n* is the system deadlock free?

## 12.3.6 What is the smallest value of *x* for which this is a safe state?

## *Chapter 12.4: Labs*

**1.**

```
# Get the last provided argument
foreach($i in $input)
{
    $num = $i
}

# Otherwise request input
if([string]::IsNullOrEmpty($num))
{
    $num = read-host
}

if($num -lt [math]::pow(2,10))
{
    write-output $num"B"
}
elseif($num -lt [math]::pow(2,20))
{
    write-output ([math]::round(($num/[math]::pow(2,10)),2))"KB"
}
elseif($num -lt [math]::pow(2,30))
{
    write-output ([math]::round(($num/[math]::pow(2,20)),2))"MB"
}
elseif($num -lt [math]::pow(2,40))
{
    write-output ([math]::round(($num/[math]::pow(2,30)),2))"GB"
}
```

**2.**

```
# Get the last provided argument
foreach($i in $input)
{
    $num = $i
}

if([string]::IsNullOrEmpty($num))
```

```
{
    $num = read-host
}

if($num -lt [math]::pow(10,3))
{
    write-output $num "ns"
}
elseif($num -lt [math]::pow(10,6))
{
    write-output ([math]::round(($num/[math]::pow(10,3)),2))"µs"
}
elseif($num -lt [math]::pow(10,9))
{
    write-output ([math]::round(($num/[math]::pow(10,6)),2))"ms"
}
else
{
    write-output ([math]::round(($num/[math]::pow(10,9)),2))"seconds"
}
```

**4.**

```
# Iterate through arguments
foreach($i in $input)
{
    $name = $(get-process -id $i |sort vm | select name,vm).Name
    $virtualMemory = (write-output $(get-process -id $i |sort vm | select name,vm).VM  |
.\human-readable-bytes.ps1)
    $workingSet = (write-output $(get-process -id $i).WorkingSet | .\human-readable-
bytes.ps1)
    $out = "******** Minne info om prosess med PID $i *********"
    $out += ([Environment]::NewLine)
    $out += "Total bruk av virtuelt minne: $virtualMemory"
    $out += ([Environment]::NewLine)
    $out += "Størrelse på Working Set: $workingSet"
    $filename = "$i--$(Get-Date -format yyyyMMdd-HHmmss).meminfo"
    write-output $out | out-file $filename



}
```

*Chapter 13.6: Theory*

## 13.6.1. Forklar påstanden til Popek og Goldberg fra 1974: A machine is virtualizable only if the sensitive instructions are a subset of the privileged instructions.

For a machine to be virtualizable, instructions that would otherwise be executed by the OS/kernel must be relayed through the hypervisor for execution, because otherwise it would be running natively. By adding a layer between the host OS and the guest OS in the form of a type 2 hypervisor (or between the hardware and the guest OS in the case of a type 1 hypervisor), which relays sensitive instructions for execution to and from the guest OS system, we achieve virtualization. If the guest OS would do this directly, it would have to be running natively.

## 13.6.2. VMware does binary translation one basic block at a time, then it executes the block and starts translating the next one. Could it translate the entire program in advance and then execute it? If so, what are the advantages and disadvantages of each technique?

Yes, it could, and this is called paravirtualization, where the code is modified in advanced and sensitive instructions are replaced with calls to the hypervisor. On one hand, this is useful to allow the guestOS to run on a type 1 hypervisor, but it makes the guestOS much less portable, whereas binary translation is a lot more flexible in that it translates instructions dynamically and requires no modification to the code beforehand.

## 13.6.3. Forklar hvordan datamaskinarkitekturens beskyttelsesringer (protection rings) benyttes ved virtualisering når virtualiseringsteknikken er binæroversetting (Vmware's teknologi).

When using binary translation, code is scanned for sensitive instructions and these are then replaced with calls to the hypervisor. The guest OS can thus be run in userland and all attempts at performing a privileged instruction will be trapped to the hypervisor, which in turn can send it further to the host OS (in case of type 2 hypervisor). In this case, the VMM can run in a more privileged protection ring than the guest OS, by running as modules to the host OS kernel.

## 13.6.4. Hva karakteriserer en applikasjon som vil være utfordrende/problematisk å kjøre på en virtuell maskin?.

It causes a lot of traps to the kernel, which is expensive.

## 13.6.5. Forklar kort fordeler og ulemper med shadow page tables i forhold til nested/extended page tables.

Modifications in the guest page table must be noticed, and such modifications lead to further modifications in the Shadow Page Table and the Physical Page Map which causes significant

overhead. This is also true in reverse – changes to the Shadow Page Table lead to a need to update the page map and guest page table.

## *Chapter 13.7: Lab*

## 1.

```
# If full path, save. Else, append path of current working dir
if($args[0] -Match "\\")
{
   $dir = $args[0]
}
else
{
   $dir = $MyInvocation.MyCommand.Path | split-path
   $dir += "\$args"
}


$disk = $dir | split-path
$disk = $disk[0]
$disk += ":"
write-host $disk


write-host "Directory $dir"
$space = Get-WmiObject Win32_LogicalDisk -ComputerName localhost -Filter "DeviceID='$disk'"
| Select-Object Size,FreeSpace
$used = 100-(100/($space.Size/$space.FreeSpace))


$files = (get-childitem .\Adobe -rec |where {!$_PSIsContainer} | select-object FullName,
Length | sort Length -Descending)


$fileStats = ($files | measure Length -ave -max -min)
$largestFile = $files[0].FullName
$largestFileSize = (write-output $files[0].Length | .\human-readable-bytes.ps1)
$averageFileSize = (write-output $fileStats.Average | .\human-readable-bytes.ps1)



write-host "Partisjonen $args befinner seg på er $used% full"
write-host "Det finnes "($fileStats.Count)" filer."
write-host "Den største er $largestFile som er $largestFileSize stor"
write-host "Gjennomsnittlig filstørrelse er $averageFileSize"
```

## 2.

```
# If full path, save. Else, append path of current working dir
if($args[0] -Match "\\")
{
  $dir = $args[0]
}
else
{
  $dir = $MyInvocation.MyCommand.Path | split-path
  $dir += "\$args"
}

$regex = '\.*[åøæÅÆØ].*\b'

get-childitem $dir -rec |where {!$_PSIsContainer} | select-object FullName | select-string  $regex
```