LA 3.1 CREATING A VM USING ORACLE VIRTUALBOX

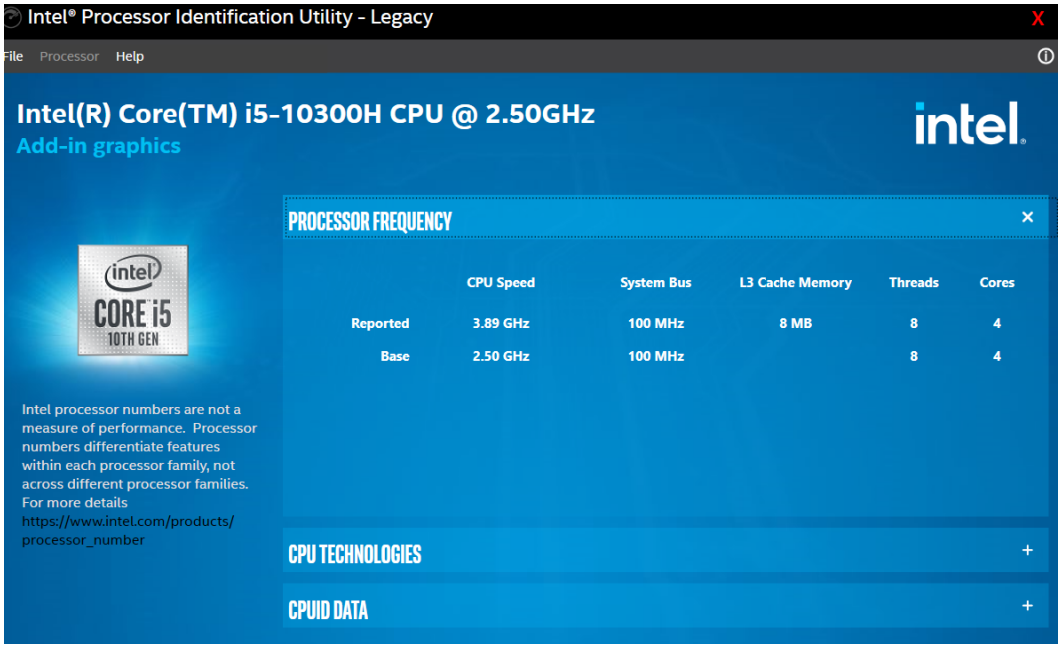
- by Group 1

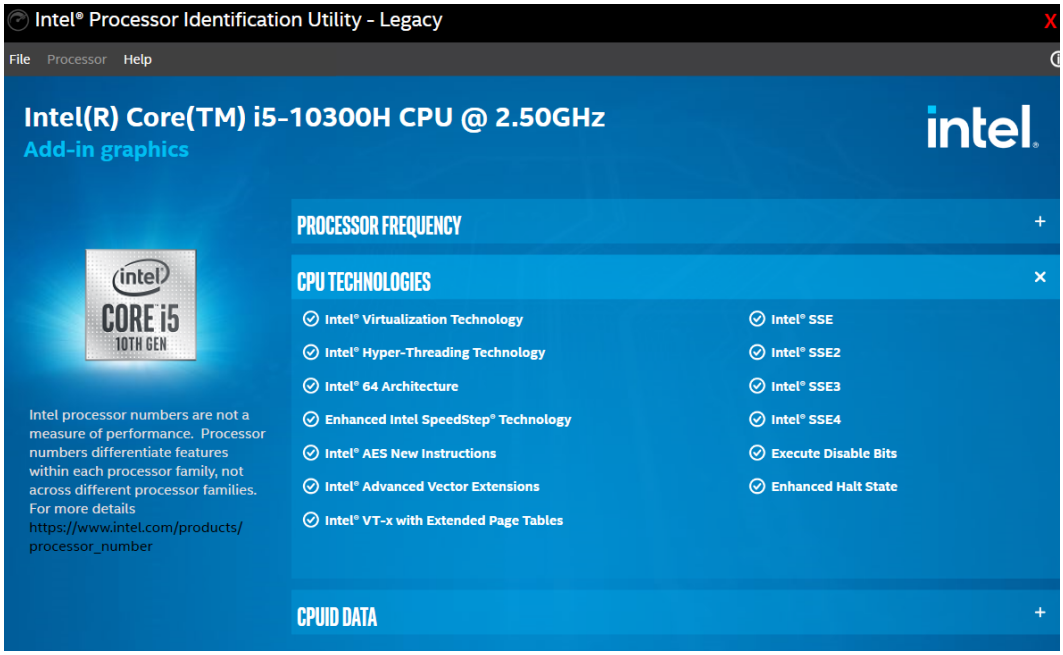
- ARCHIS SAHU - 2020A7PS1692H
- SIDDHANT PANDA - 2020A7PS0264H
- ADITYA DHANEKULA - 2020A7PS0205H
- AYUSH KALRA - 2021A7PS2222H

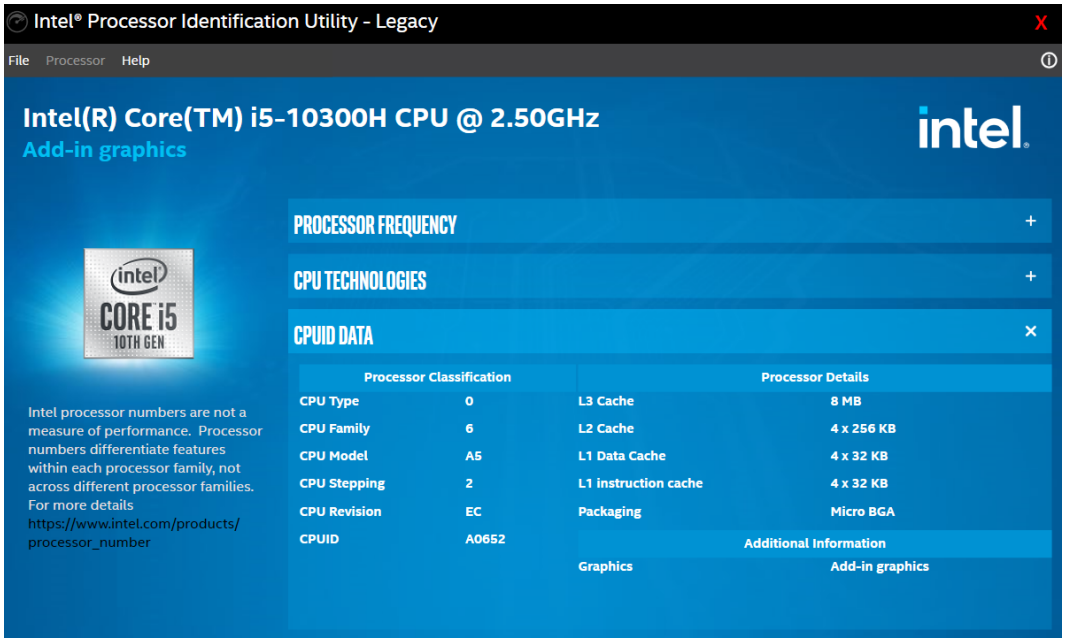
Module 1 : To check if the machine is virtualizable

- To check if the machine is virtualizable we use the **Intel Processor identification tool**

- Intel virtualization technology and HyperThreading technology are both enabled. This means that the machine is virtualizable.
    - **Hyperthreading** - It enables a single physical CPU core to function as more than one logical CPU core.
    - This machine has 8 threads and 4 cores(each core can function as 2 logical cores, as a result of hyperthreading).

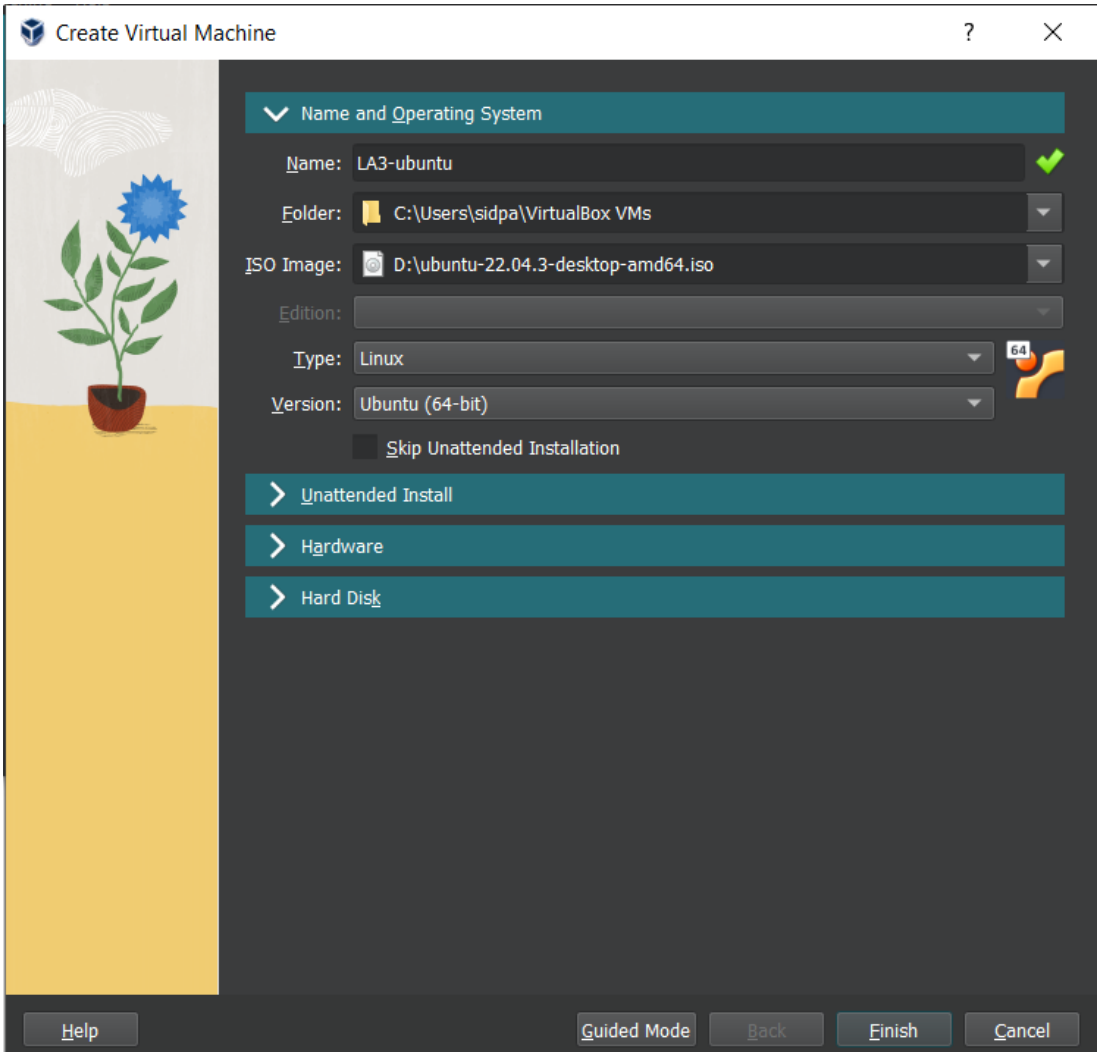

Intel® Processor Identification Utility - Legacy
File Processor Help
Intel(R) Core(TM) i5-10300H CPU @ 2.50GHz
Add-in graphics
intel
PROCESSOR FREQUENCY
CPU Speed System Bus L3 Cache Memory Threads Cores
Reported 3.89 GHz 100 MHz 8 MB 8 4
Base 2.50 GHz 100 MHz 8 4
CORE i5 10TH GEN
Intel processor numbers are not a measure of performance. Processor numbers differentiate features within each processor family, not across different processor families. For more details https://www.intel.com/products/processor_number
CPU TECHNOLOGIES
CPUID DATA


Intel® Processor Identification Utility - Legacy
File Processor Help
Intel(R) Core(TM) i5-10300H CPU @ 2.50GHz
Add-in graphics
intel
PROCESSOR FREQUENCY
CPU TECHNOLOGIES
Intel® Virtualization Technology
Intel® Hyper-Threading Technology
Intel® 64 Architecture
Enhanced Intel SpeedStep® Technology
Intel® AES New Instructions
Intel® Advanced Vector Extensions
Intel® VT-x with Extended Page Tables
Intel® SSE
Intel® SSE2
Intel® SSE3
Intel® SSE4
Execute Disable Bits
Enhanced Halt State
CORE i5 10TH GEN
Intel processor numbers are not a measure of performance. Processor numbers differentiate features within each processor family, not across different processor families. For more details https://www.intel.com/products/processor_number
CPUID DATA

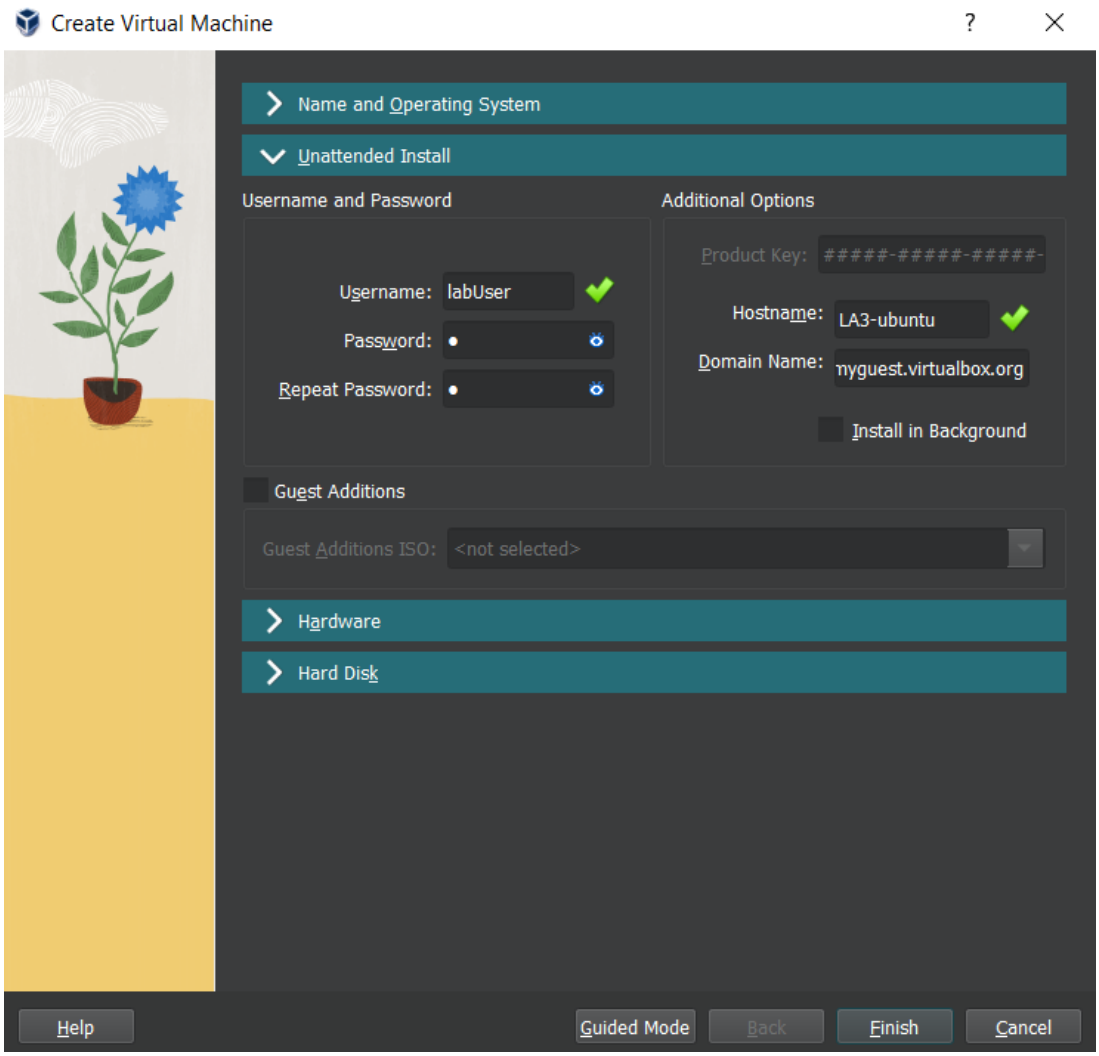


Module 2 : To configure a VM

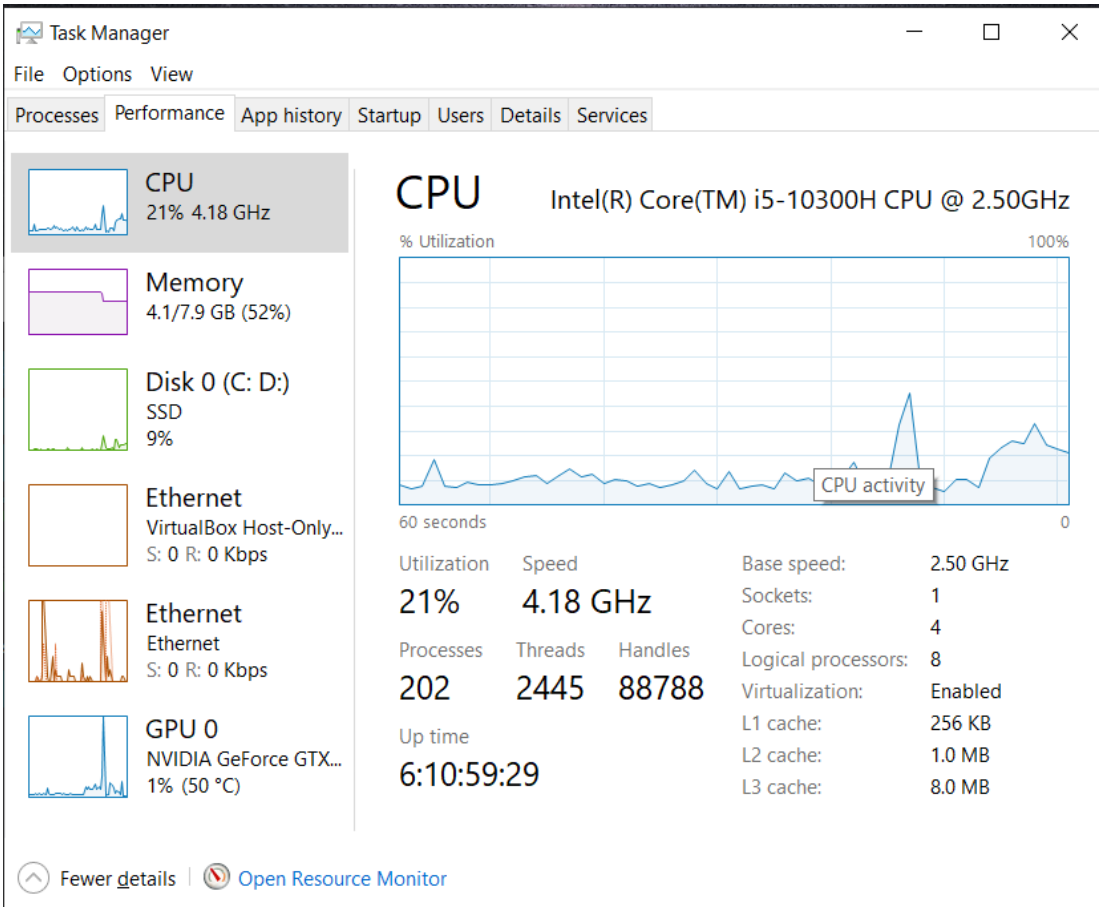
- **Step 1** : Open Oracle VirtualBox and click on *"New Virtual Machine".* The following screen appears.
    a. Name the VM
    b. Select the folder where the files related to this VM are stored
    c. Select the ISO image from which the guest OS shall be installed



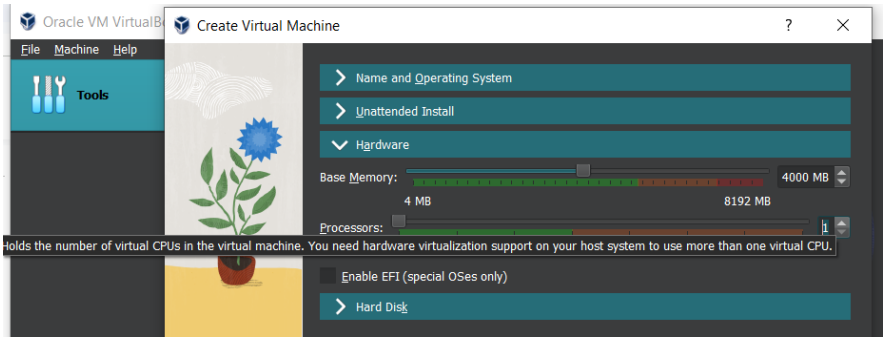
- **Step 2** : Unattended installation - Provide with the necessary details like the username and password for the VM.
    - On starting the VM, unattended installation is performed automatically. It changes the boot device order to boot from the virtual hard disk first
    - We see a checkbox called “**Guest Additions**”. It is used for installation inside the VM after the guest OS has been installed. They have
        - Device drivers
        - System applications
        - It is used to optimize the system’s performance and usability. For ex : Shared folders, Seamless windows, etc



- **Step 3** : Choosing memory



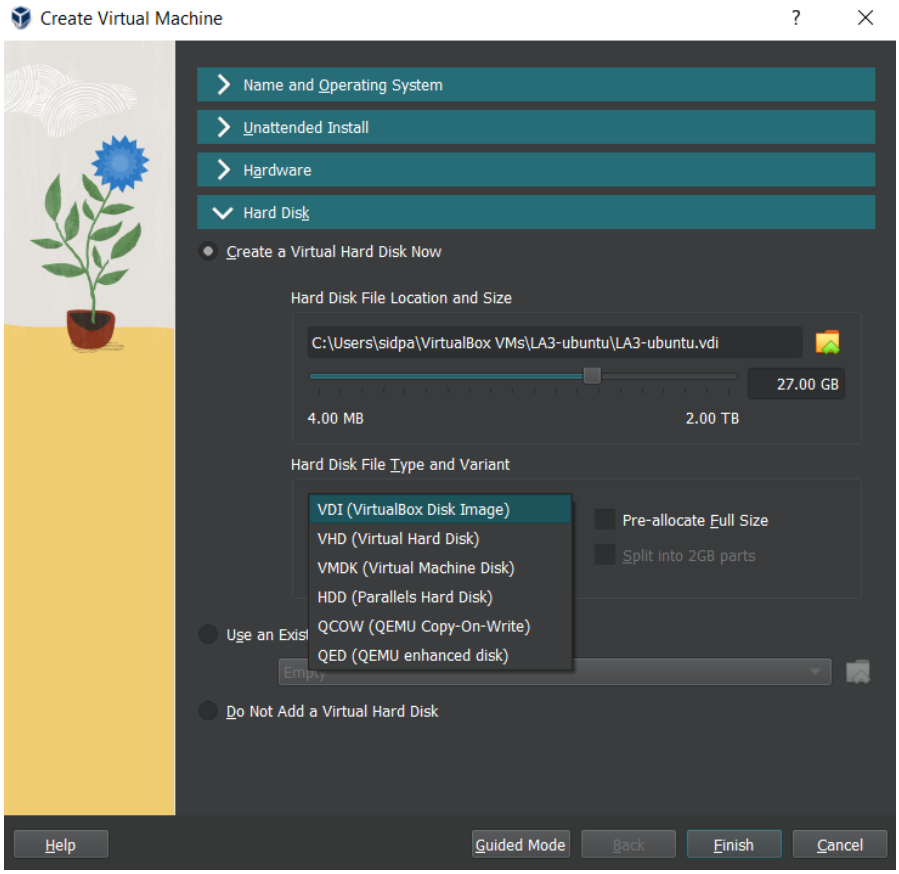
- We notice by checking the performance of the host OS that in normal use, about 4 / 8 GB of RAM is used. This tells us that another 4 GB (in general) can be allotted to our VM, for its efficient usage.



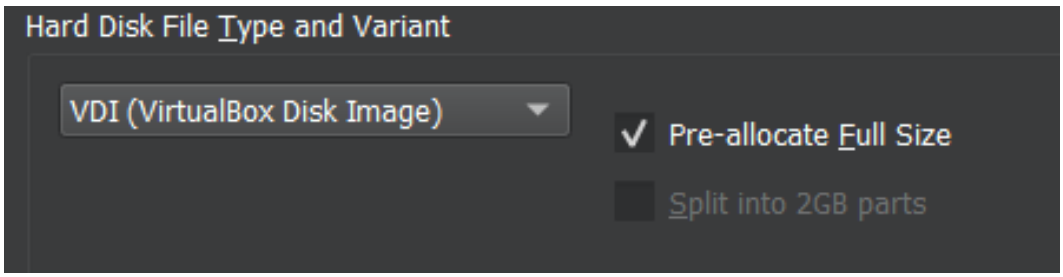
- We choose the number of processors to be 4, this shows us that the number of vCPUs in my VM can be 4 at max

- **Step 4** : Configuring the Hard disk
  - We allot a total hard disk space of about 27 GB to the VM, in the shown directory.

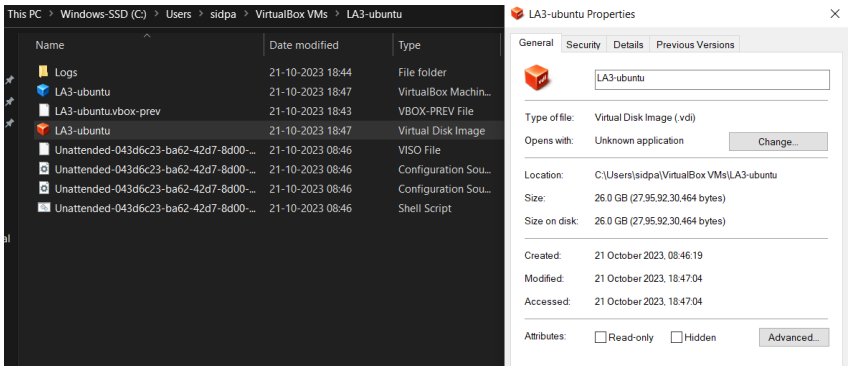
- We chose to create a new virtual Hard Disk, but we could have used a pre-made .vdi file to create a virtual hard disk from an image.



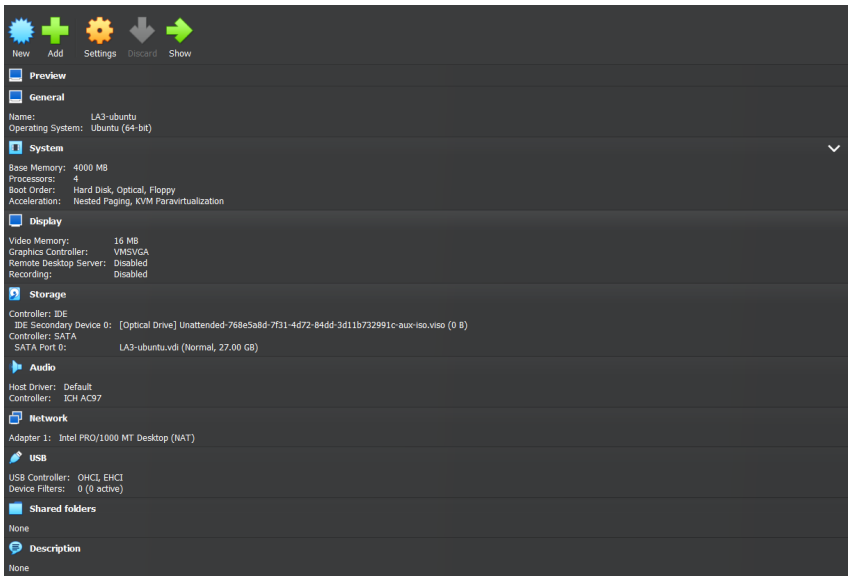
- The Hard Disk storage to be created can be done in two ways :
    i. **Pre Allocating the Full size** - entire space allocated upfront ; better performance
    ii. **Dynamically Allocating the disk** - space allocated as used ; slower performance



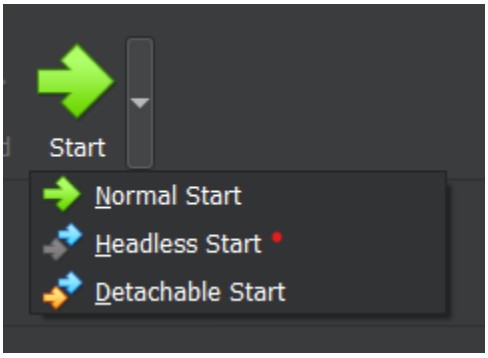
- We check in the directory of the Virtual machine that the .vdi image file is of about 27 GB, which is equal to the size of the pre-allocated disk. Everything is allocated up-front, hence taking up the whole 27 GB of space.



- A summary of the configuration of the VM to be created :

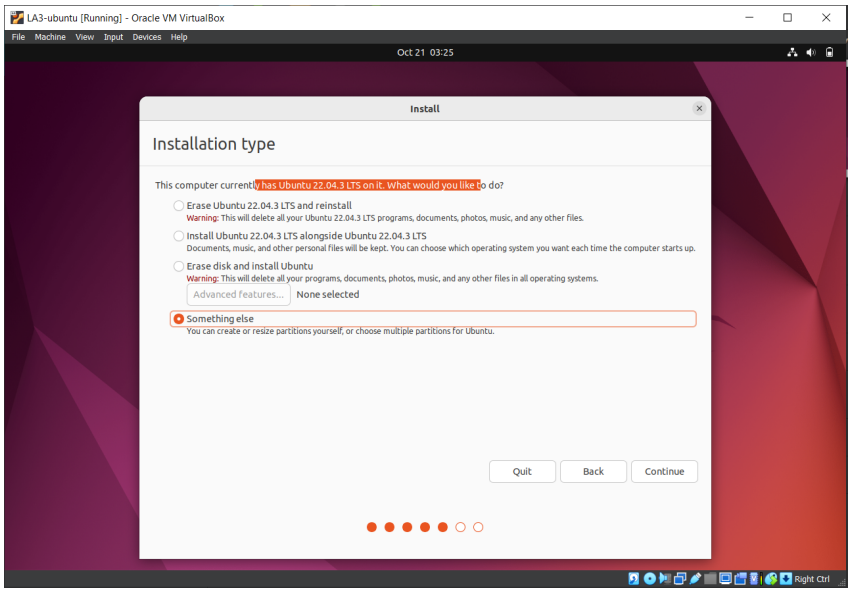


- **Step 5** : Start the VM

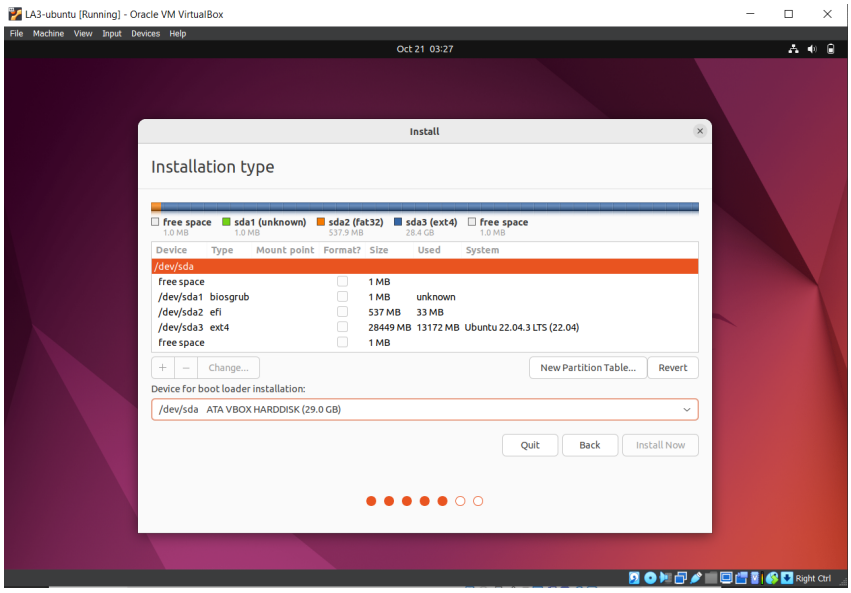


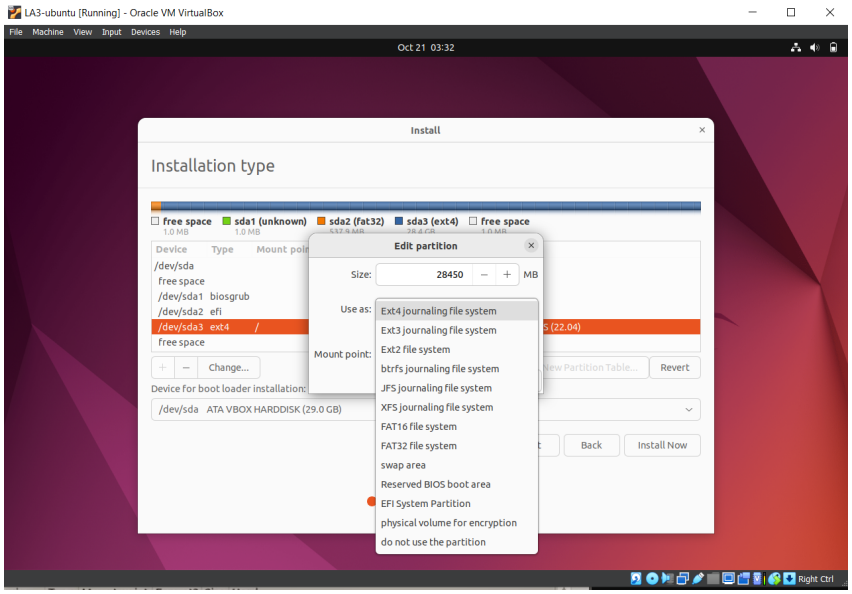
  - **Normal Start** - runs with a GUI
    Headless Start - runs without a GUI
    Detachable Start - runs without a GUI, but can be attached at any point in time

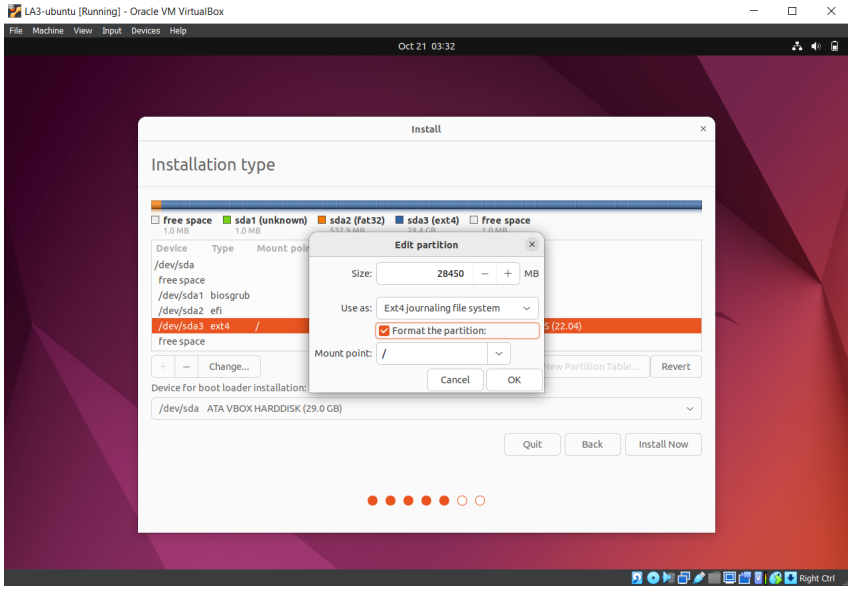
- Complete the installation of the OS. Choose “*Something else*” to create your own partitions for the guest OS



- We create a partition and mount it on the root directory (in our case the /dev/sda3) and use an ext4 journaling file system

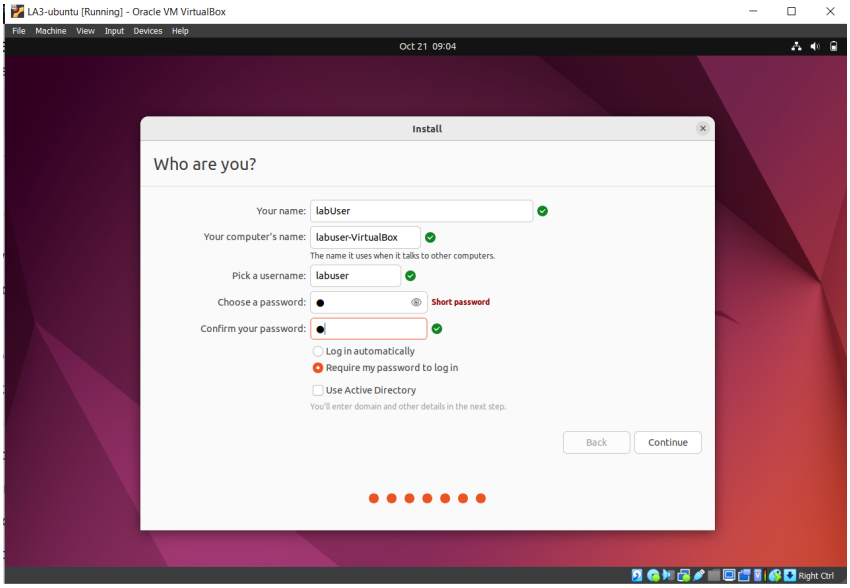






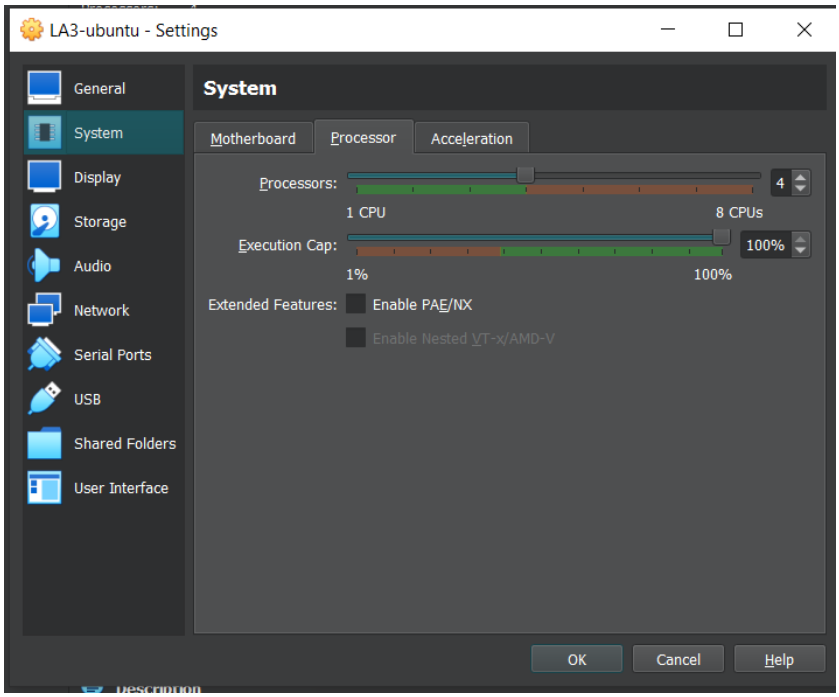
    - Using the format partition checkbox - all data on the mount point gets deleted, hence giving a fresh new partition

- Finally give the username and password to finish the installation



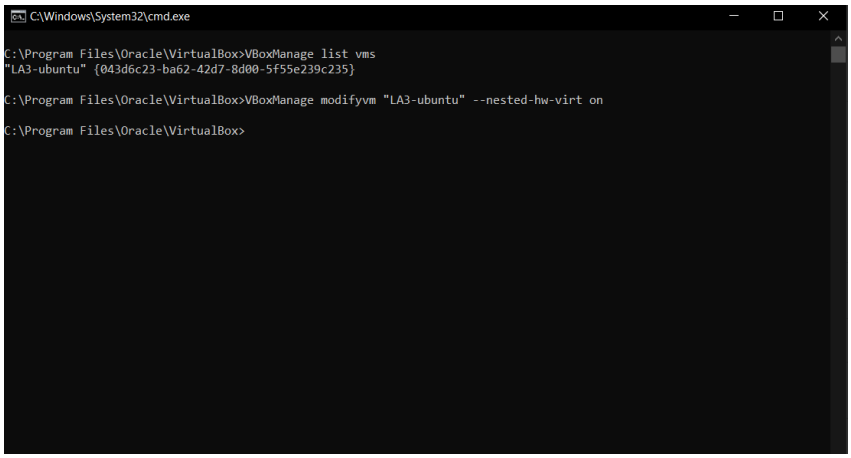
Module 3 : To check if nested virtualization is possible

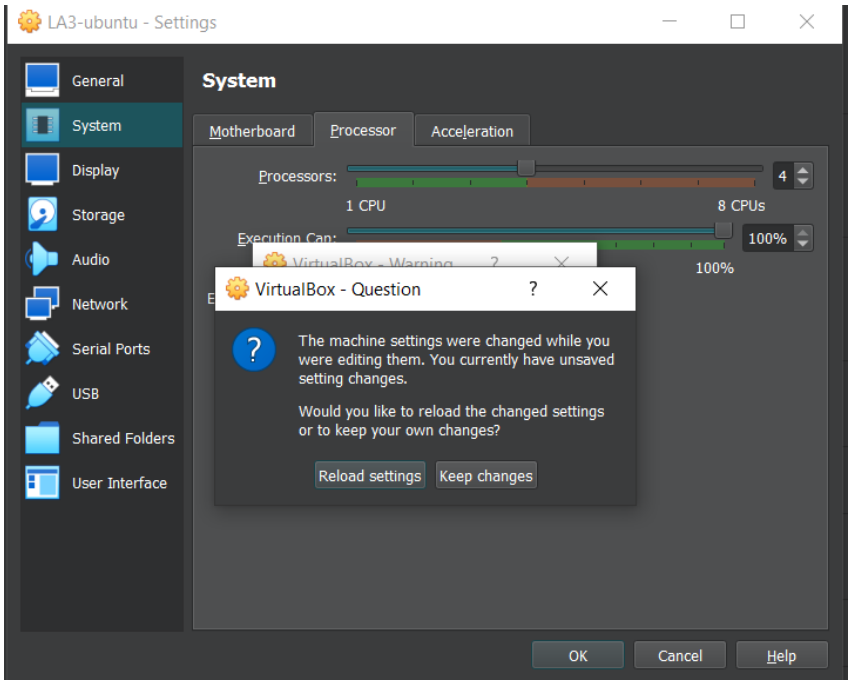
- **Step 1** : Now that the VM is up and running perfectly, let us peek on the processor settings again:

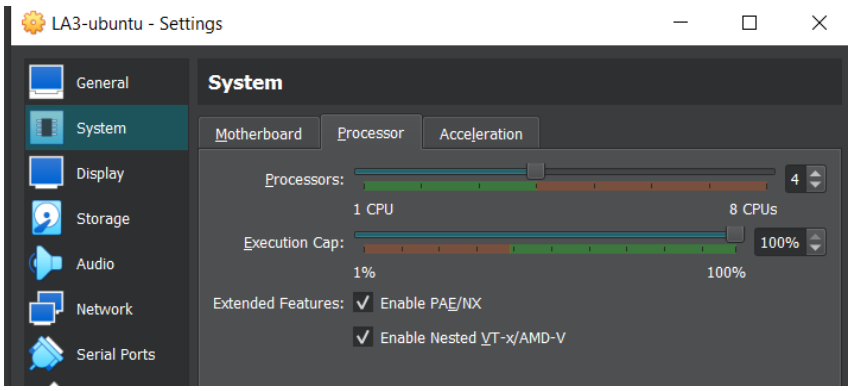


- In the extended features, we see two options, let us discuss each :
    i. **PAE** - Physical Address Execution :
        - PAE is a memory management feature introduced in x86 and x86-64 architectures to allow 32-bit processors to access more than 4 GB of physical memory (RAM). The traditional 32-bit architecture is limited to addressing a maximum of 4 GB of RAM because of the way memory addresses are structured.
        - PAE extends the physical memory addressing from 32 bits to 36 bits, allowing the processor to access up to 64 GB of RAM (though practical limits are usually lower, depending on the operating system and hardware). It does this by adding an extra layer of addressing, enabling the CPU to access a larger range of physical memory addresses.
        - PAE is especially relevant in virtualization when running 32-bit guest operating systems on a 64-bit host. Enabling PAE allows the guest OS to access more than 4 GB of RAM if it's available on the host.
    ii. **NX** - No Execute :
        - NX is a security feature that helps prevent the execution of malicious code in memory. It marks certain areas of memory as non-executable, ensuring that data can't be executed as if it were a program.
        - It helps isolate and protect each virtual machine from executing potentially malicious code from the other VMs.
    iii. Enable VT-X/AMD-V virtualization :
        - This technology is what enables virtualization in a machine. Our host machine had it enabled, hence we were able to install VirtualBox and use it. Likewise, allowing this functionality for the VM we create will let us create more VMs inside itself - Nested Virtualization can be enabled.

- **Step 2** : Grayed out VT-X ; enabling it using the CLI **[Look at VT-X functioning at the end of the document for greater detail]**
    - The option to enable VT-x is grayed out by default because the CPUs we use lack a certain feature called VMCS shadowing which improves nested HW virtualization performance tremendously, it will be terrible otherwise.
    - However, it is possible to do it using VBoxManage, because it is meant as a tool to even make VM config changes which can break the VM and it is assumed the user knows what he/she is doing.


C:\Windows\System32\cmd.exe
C:\Program Files\Oracle\VirtualBox>VBoxManage list vms
"LA3-ubuntu" {043d6c23-ba62-42d7-8d00-5f55e239c235}
C:\Program Files\Oracle\VirtualBox>VBoxManage modifyvm "LA3-ubuntu" --nested-hw-virt on
C:\Program Files\Oracle\VirtualBox>

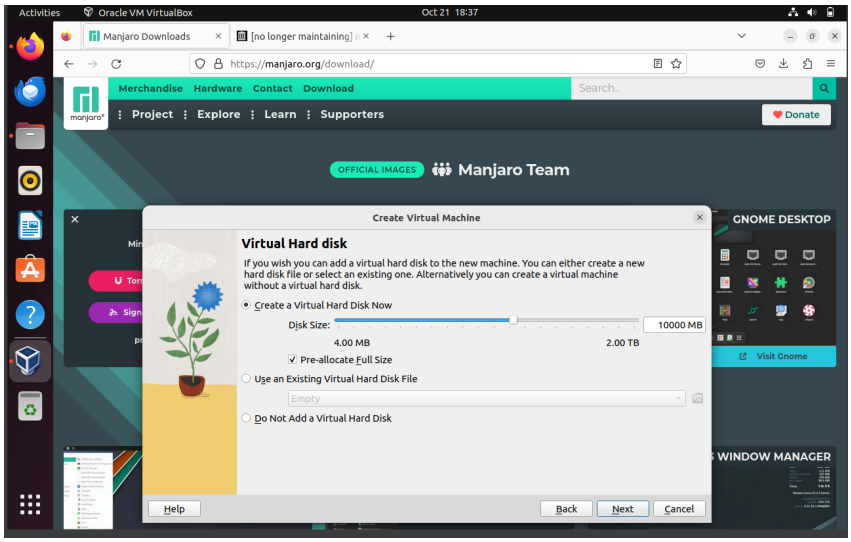

LA3-ubuntu - Settings
General
System
Display
Storage
Audio
Network
Serial Ports
USB
Shared Folders
User Interface
System
Motherboard
Processor
Acceleration
Processors:
4
1 CPU
8 CPUs
Execution Cap:
100%
100%
VirtualBox - Question
The machine settings were changed while you were editing them. You currently have unsaved setting changes.
Would you like to reload the changed settings or to keep your own changes?
Reload settings
Keep changes
OK
Cancel
Help


LA3-ubuntu - Settings
General
System
Display
Storage
Audio
Network
Serial Ports
System
Motherboard
Processor
Acceleration
Processors:
4
1 CPU
8 CPUs
Execution Cap:
100%
1%
100%
Extended Features:
Enable PAE/NX
Enable Nested VT-x/AMD-V

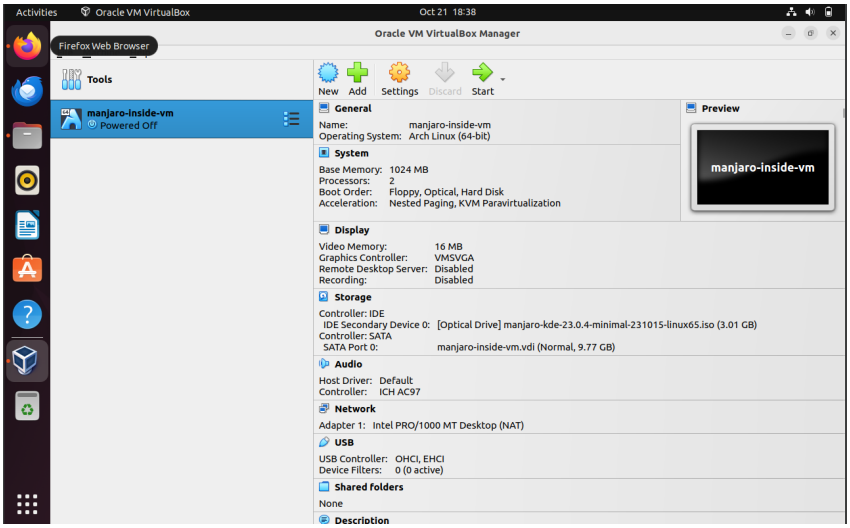
- Now we can see that the checkbox is ticked, hence nested virtualization can now be possible.

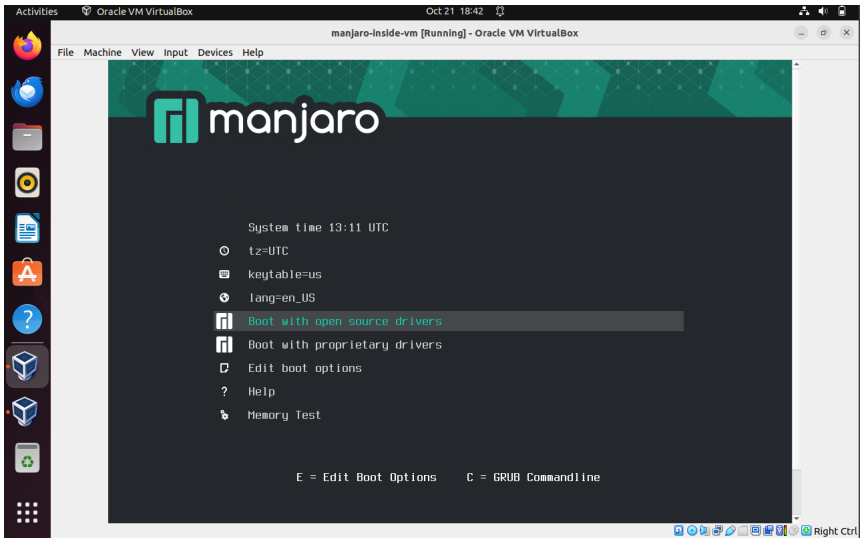
- **Step 3** : Making a VM inside a VM
  - Install VirtualBox in the newly created Ubuntu VM
  - Download the .iso file for the desired guest OS - in our case Manjaro Linux (Arch based distro.)

  - Allocate the disk space according to the needs, here we allocate 1 GB memory.

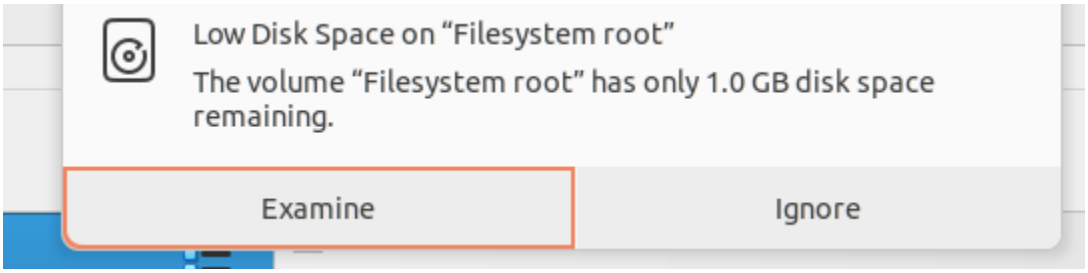


  - Start the new vm-inside-vm and use it like you normally would





- But we notice that the system says that it has very little space left, and starts malfunctioning. The performance of nested virtualization is abysmal, as expected due to the lack of resources in the host VM itself.



APPENDIX : VMCS and VT-x

VT-x (Virtualization Technology for x86) is a hardware virtualization technology developed by Intel for x86 and x86-64 processors. It is a set of processor extensions and features designed to enhance virtualization capabilities, making it more efficient, secure, and reliable to run virtual machines (VMs) on the same physical hardware. VT-x technology is commonly used in virtualization software like VMware, VirtualBox, and Microsoft Hyper-V to create and manage VMs.

Here's a more detailed explanation of VT-x virtualization:

1. **Virtual Machine Monitor (VMM) Support**: VT-x enables the creation of a Virtual Machine Monitor (VMM), also known as a hypervisor. The VMM is a software layer that manages multiple VMs on a single physical host. VT-x helps the VMM efficiently control hardware resources and isolate the execution of multiple VMs.

2. **Hardware-Assisted Virtualization**: VT-x provides hardware-level support for virtualization. This means that the processor itself is designed with features that make it easier to create and manage virtual machines. These features include:
    a. **Virtual Machine Control Structure (VMCS)**: VT-x processors have a VMCS that allows the VMM to control and manage VMs. The VMCS contains information about the state of a VM, its resources, and execution control.
    b. **VMX Root and VMX Non-Root Operation**: VT-x introduces two operation modes: VMX Root Mode (for the hypervisor) and VMX Non-Root Mode (for the VMs). The transition between these modes is done securely and efficiently.
    c. **Privilege Level and Execution Control**: VT-x allows VMs to run at different privilege levels, such as Ring 0 (kernel mode) and Ring 3 (user mode), with controls over which VM can execute privileged instructions.

3. **Improved Isolation**: VT-x improves the isolation of VMs from each other and from the host system. This isolation is critical for security and reliability. Each VM operates as if it has its own dedicated hardware resources.

4. **Performance Enhancements**: VT-x provides performance benefits by reducing the overhead of virtualization. This includes optimized instruction execution and improved context switching between VMs.

5. **Security Features**: VT-x helps enhance the security of VMs by enabling features like Extended Page Tables (EPT) and Second Level Address Translation (SLAT). These

features help in better memory management and prevent VMs from accessing each other's memory.

6. **Compatibility**: VT-x technology ensures better compatibility with various operating systems and software. It allows VMs to run a wide range of operating systems, including Windows, Linux, macOS, and others.

7. **Nested Virtualization**: VT-x also supports nested virtualization, which means you can run VMs within VMs. This is particularly useful for testing and development environments.

8. **I/O Virtualization**: VT-x includes features that enable better handling of input/output (I/O) devices in virtualized environments, improving performance and scalability.

In summary, VT-x is a set of hardware features integrated into Intel processors to enhance virtualization on x86 and x86-64 systems. It enables more efficient and secure virtualization by providing direct hardware support for virtualization, improved isolation, and better performance. VT-x is a key component in running virtualization software and creating virtualized environments for various purposes, including server consolidation, software testing, and development.

—--

VMCS (Virtual Machine Control Structure) shadowing is a technique used in hardware-assisted virtualization to optimize the management of virtual machines (VMs) by reducing the overhead of managing VMs. It is a feature often associated with Intel VT-x and AMD-V virtualization technologies. VMCS shadowing improves the performance and efficiency of virtualization platforms by allowing a virtual machine monitor (VMM) to delegate certain aspects of VM management to a host or hypervisor while retaining control over others.

Here's a more detailed explanation of VMCS shadowing:

1. Purpose: VMCS is a data structure associated with each VM, containing information about its state, configuration, and execution control. It is managed by the VMM. VMCS shadowing allows the VMM to offload some of its responsibilities to the host or hypervisor, which can handle certain VMCS-related tasks on its behalf. This reduces the workload on the VMM and improves performance.

2. Two Levels of VMCS: VMCS shadowing essentially creates two levels of VMCS: the "shadow VMCS" and the "VMCS." The shadow VMCS resides at the hypervisor or host level, and the VMCS is specific to the guest VM. The shadow VMCS shadows some of the fields and settings from the VMCS.

3. Offloaded VMCS Fields: Certain VMCS fields can be offloaded to the shadow VMCS. These fields are typically those that do not need frequent changes during VM execution.

Fields related to the basic configuration and setup of the VM can be shadowed, such as the guest's register state, the guest's physical memory address space configuration, and initial control settings.

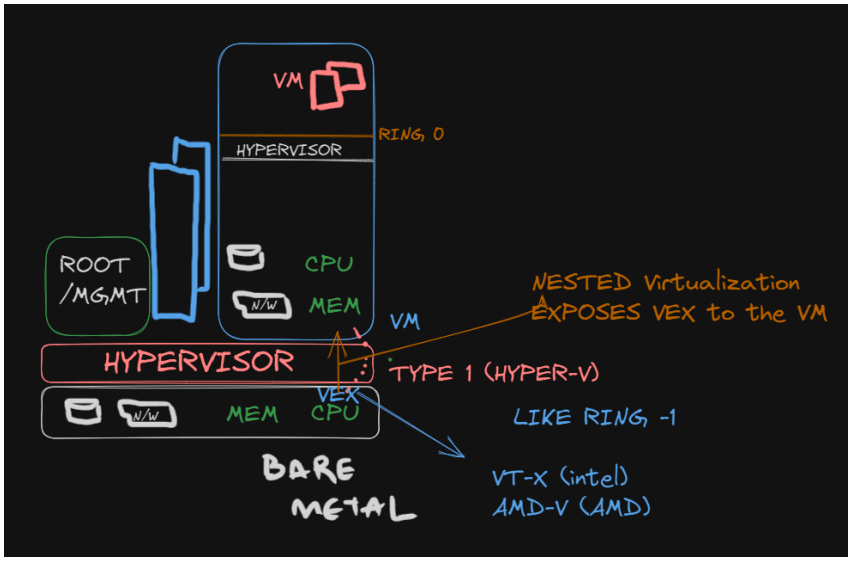
4. Improved Efficiency: VMCS shadowing improves the efficiency of VM management by reducing the need for frequent VM exits and re-entries. VM exits are events where the VMM takes control from the VM to perform certain tasks, such as handling exceptions, I/O operations, or other VM management tasks. Minimizing VM exits is crucial for optimizing VM performance.

5. Reduced Overhead: By delegating the management of certain VMCS fields to the shadow VMCS, VMCS shadowing reduces the overhead associated with VM control and context switching. This results in improved VM performance and better resource utilization.

6. Simplified VM Management: VMCS shadowing simplifies VM management for the VMM by allowing it to focus on the dynamic and frequently changing aspects of VM execution while delegating the static and setup-related configurations to the shadow VMCS.

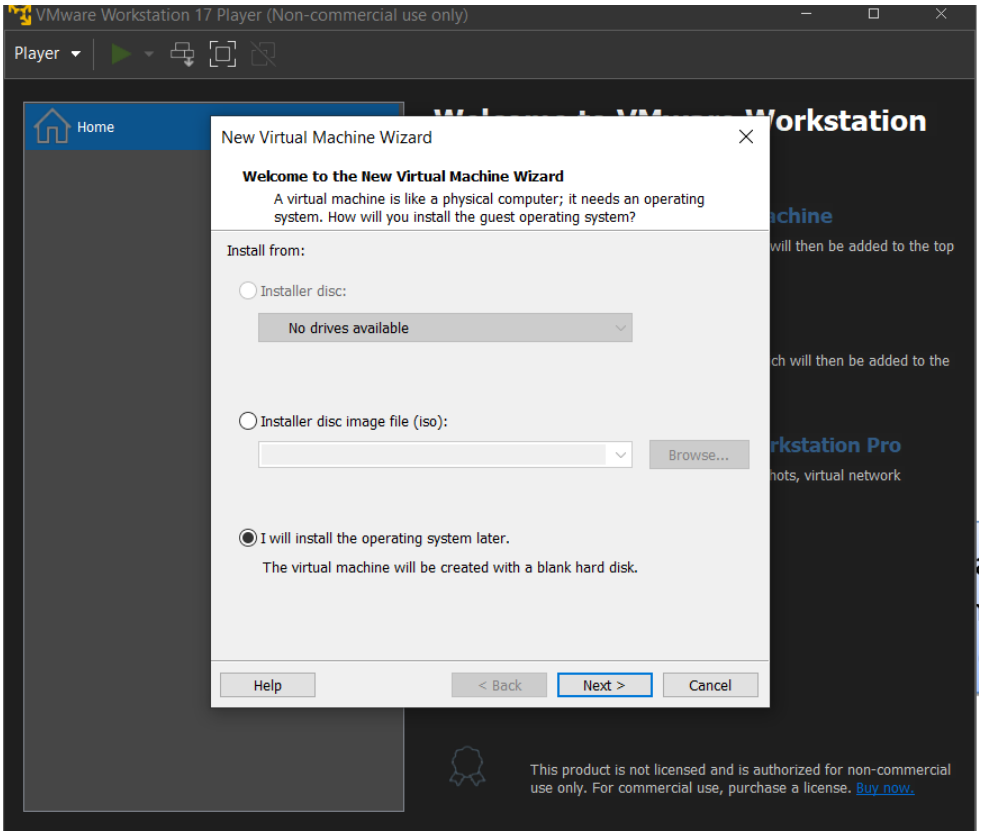
7. Example Use Cases: VMCS shadowing is particularly useful in scenarios where virtualization efficiency is critical, such as data centers, cloud computing environments, and virtualization platforms that host numerous VMs. It allows for better consolidation of VMs on a physical host while maintaining good performance.

In summary, VMCS shadowing is a hardware-assisted virtualization technique that allows a virtual machine monitor (VMM) to offload certain aspects of VM management to a shadow VMCS at the host or hypervisor level. This reduces the overhead associated with VM control and context switching, resulting in improved VM performance and efficiency, especially in environments where resource consolidation is essential.

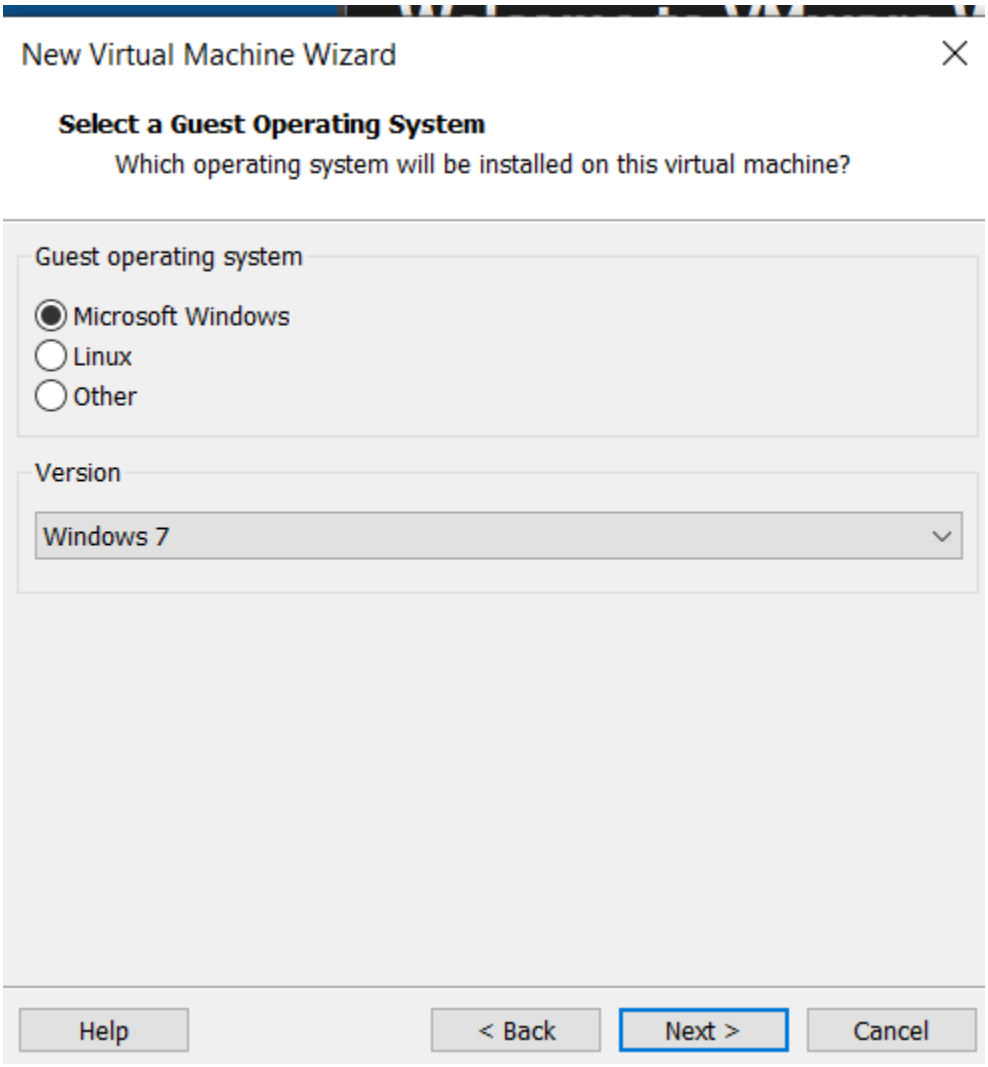


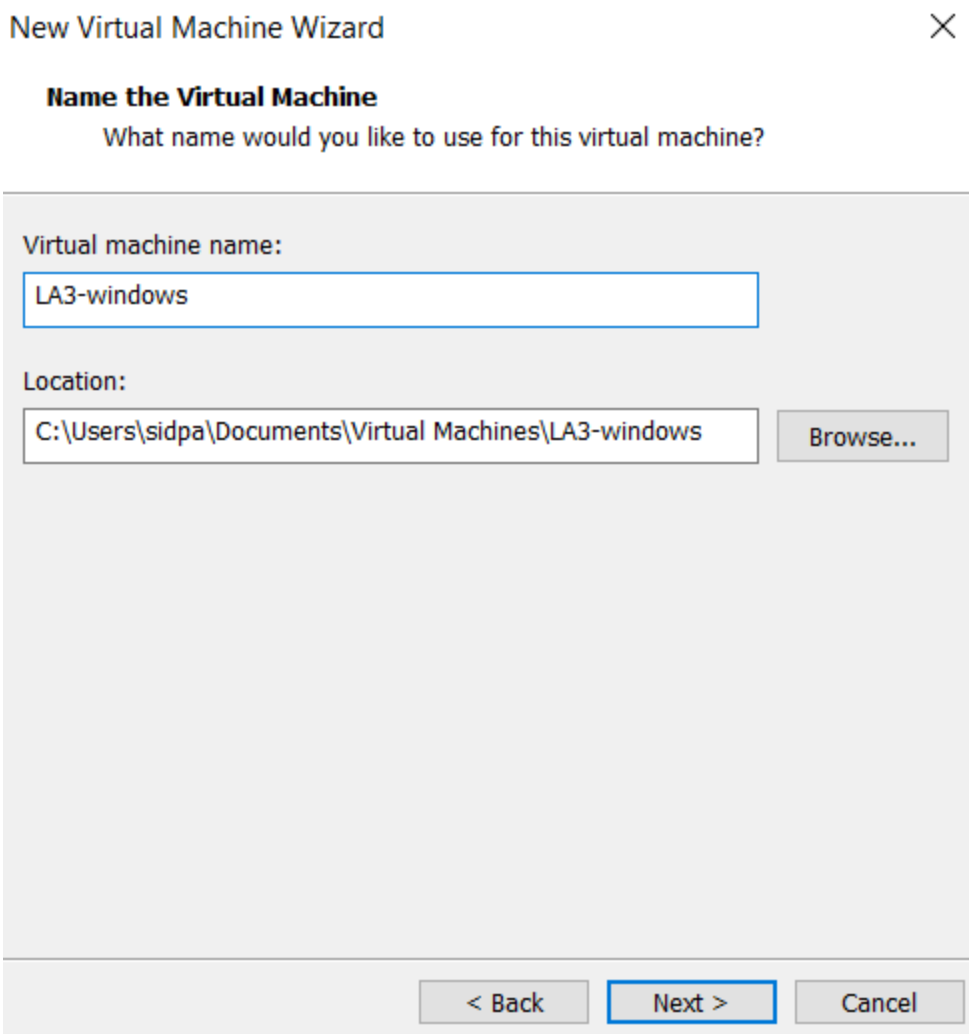
LA 3.2 - CREATING A VM USING VMWARE WORKSTATION

- **Step 1** : Create a new VM, and the following screen appears :

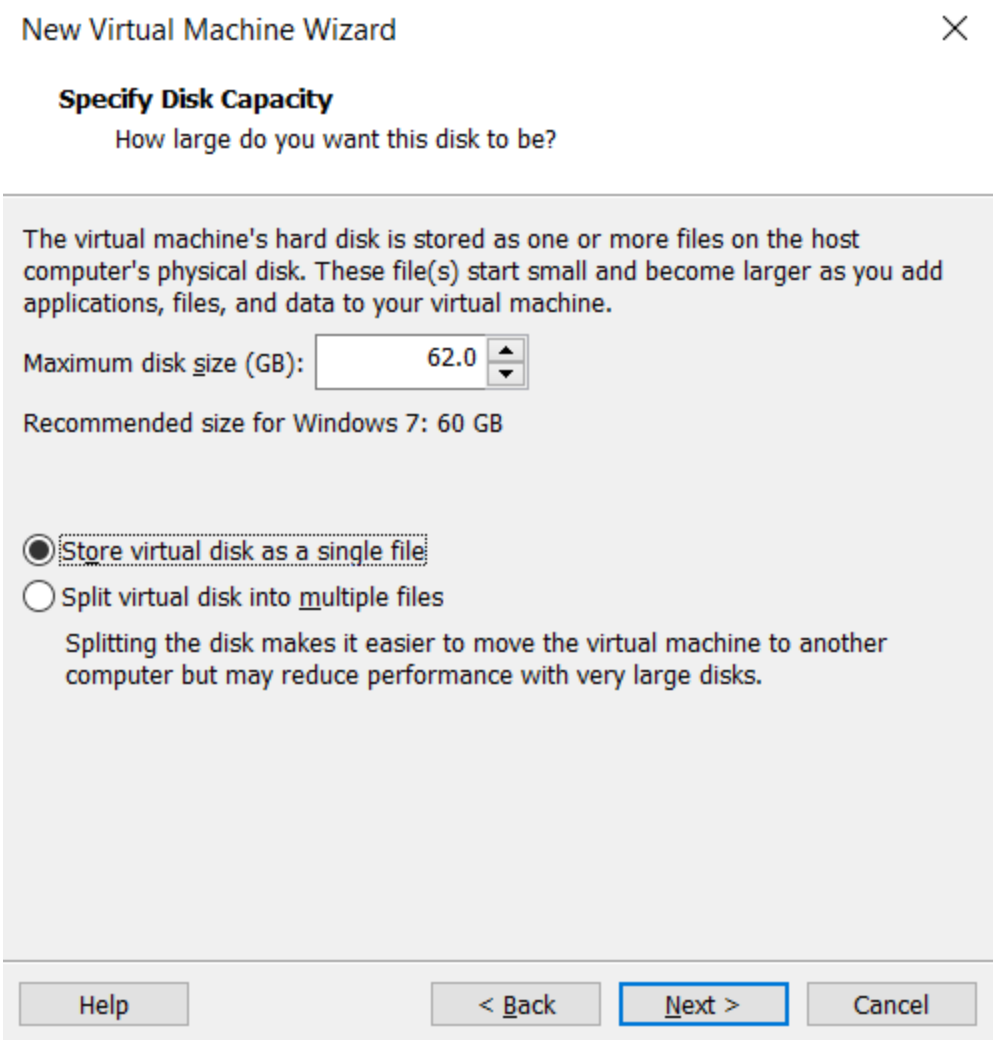


- **Step 2** : Select the guest OS that you would like to install (can be edited later), and name your VM.

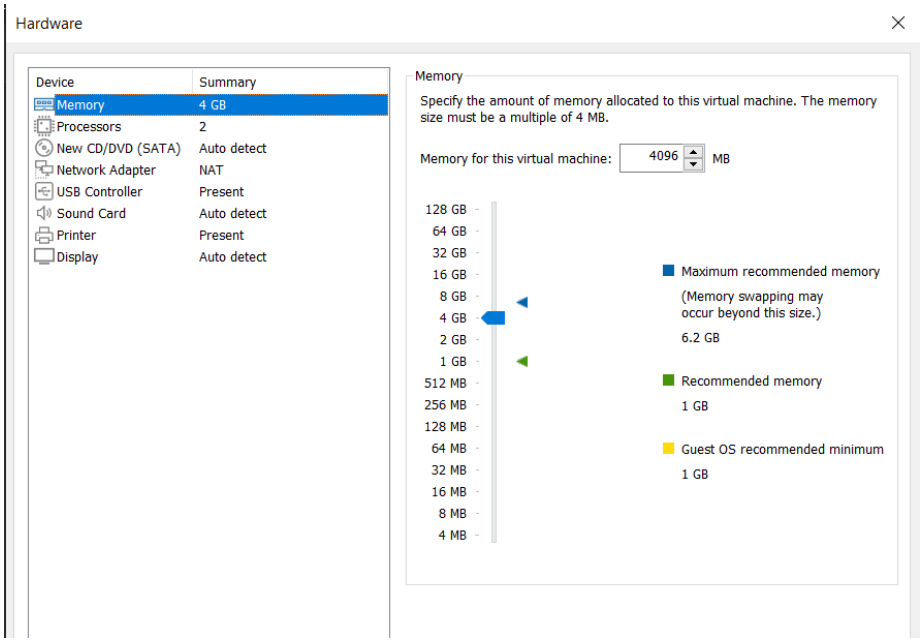


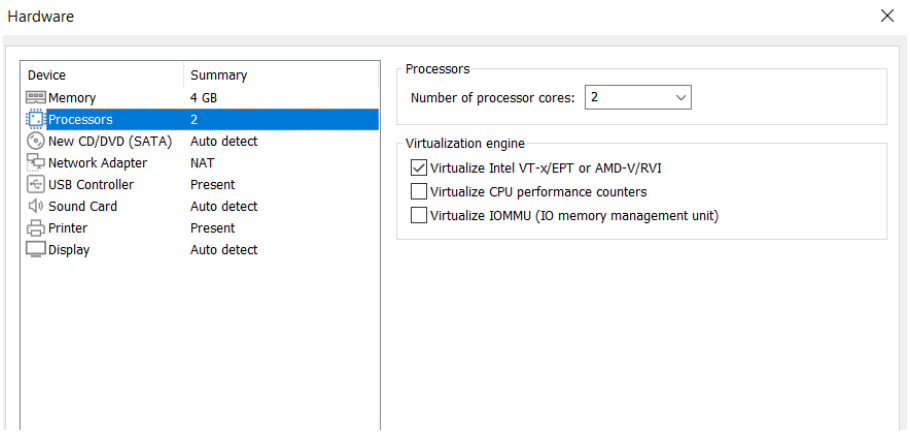


- **Step 3** : Specify the disk capacity (in our case an arbitrary 62 GB)
    - By default, the disk created is dynamically allocated as shown
    - We can store the disk as a single file, or as split files. Split files disk storage are easier to transfer while transferring VMs, but if the disks are too large, there will be too many splits and hence inefficient to transfer.

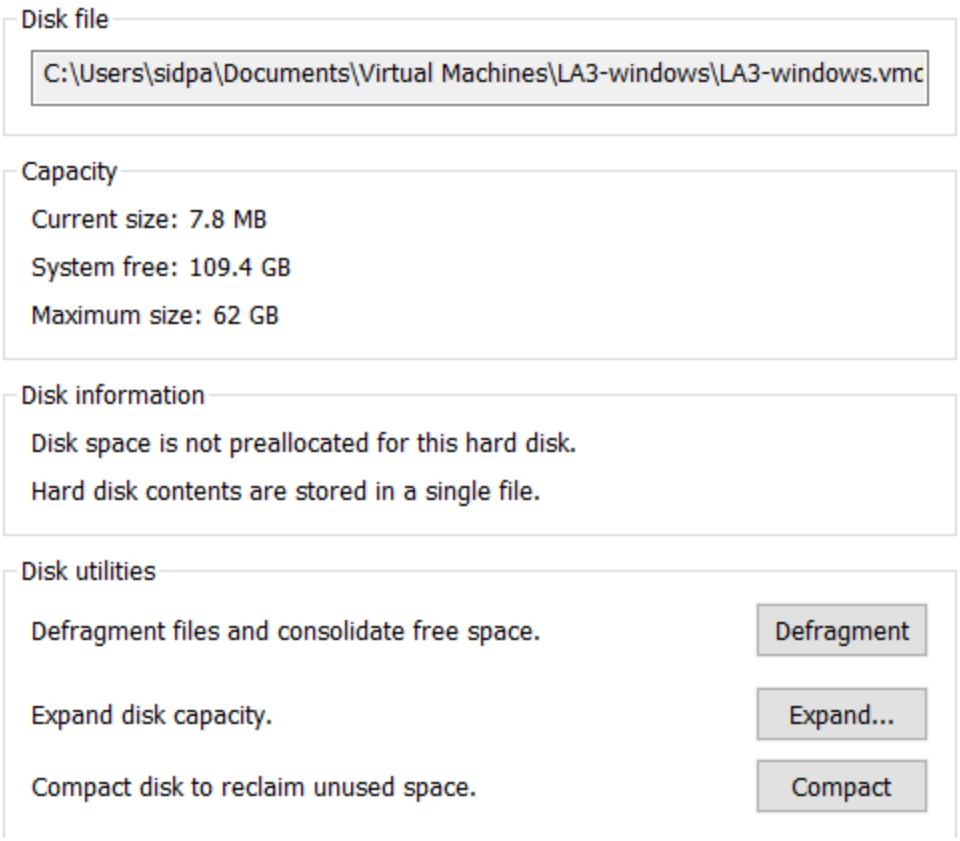


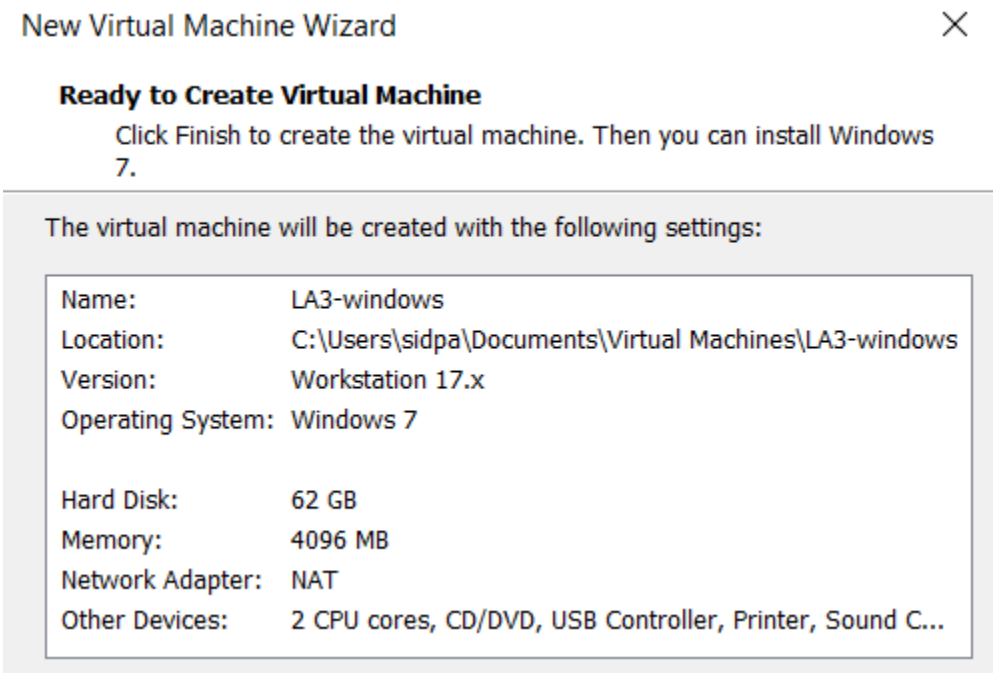
- Let us now look at the advanced hardware settings, we can configure the memory, the number of processors and other characteristics we wish to provide for the VM



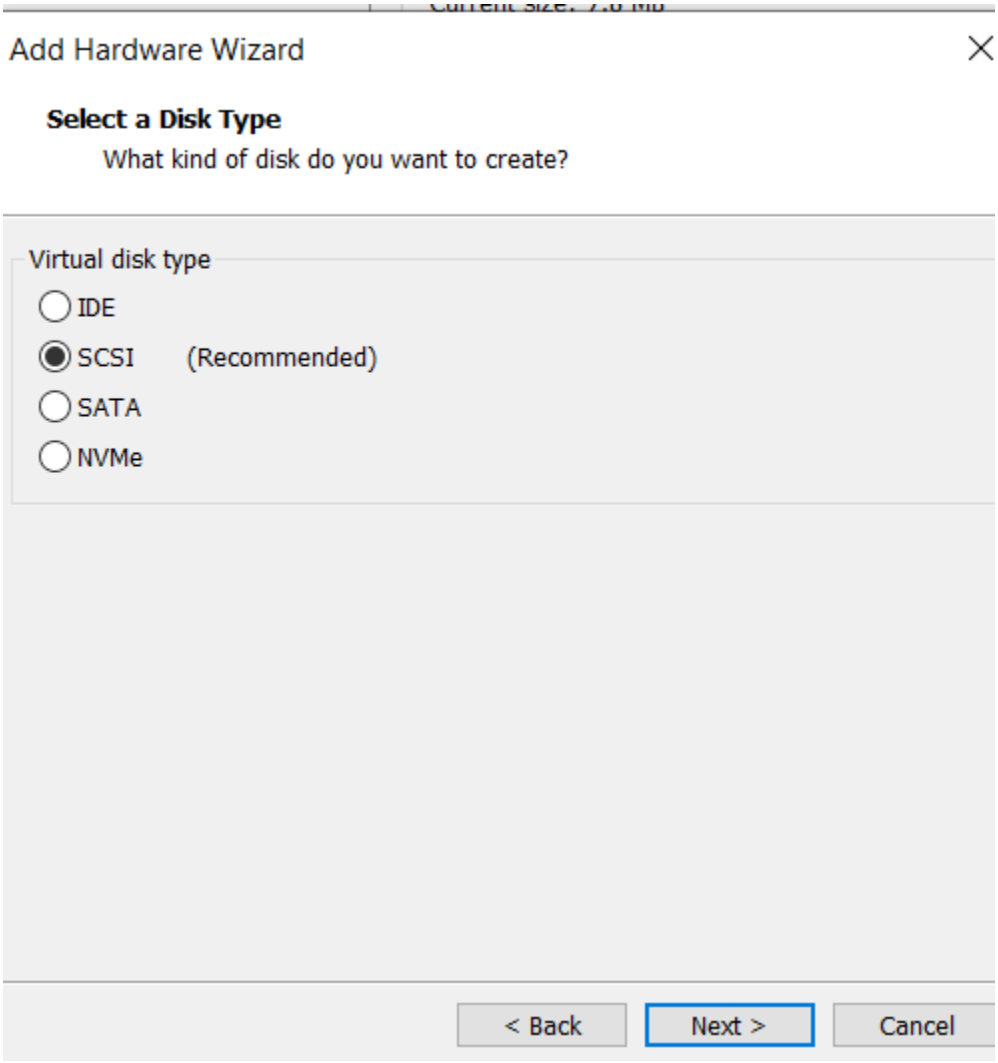


- Analyze the virtual hard disk, as it is dynamically allocated we can defragment, expand and compact the disk as and when required.

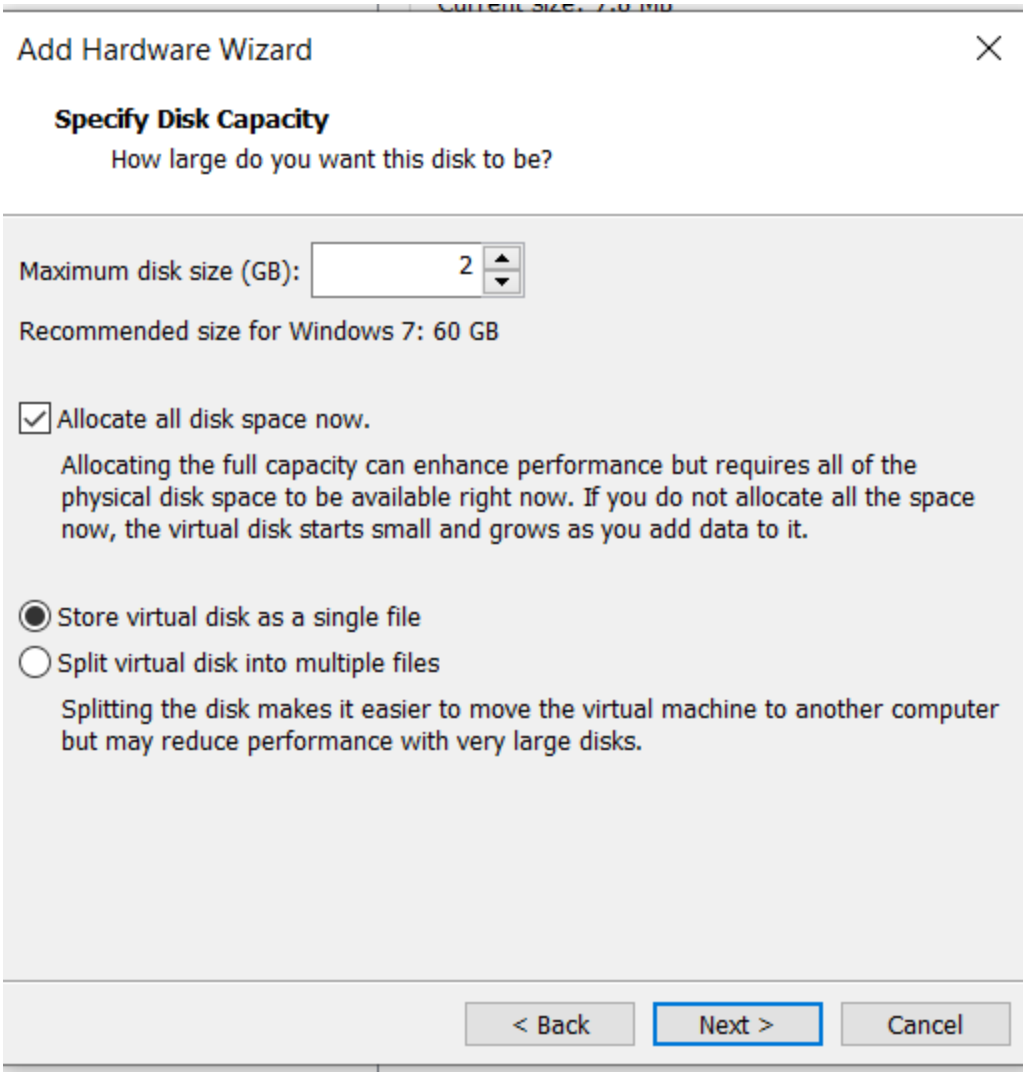


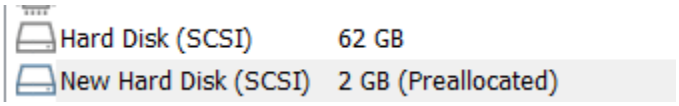


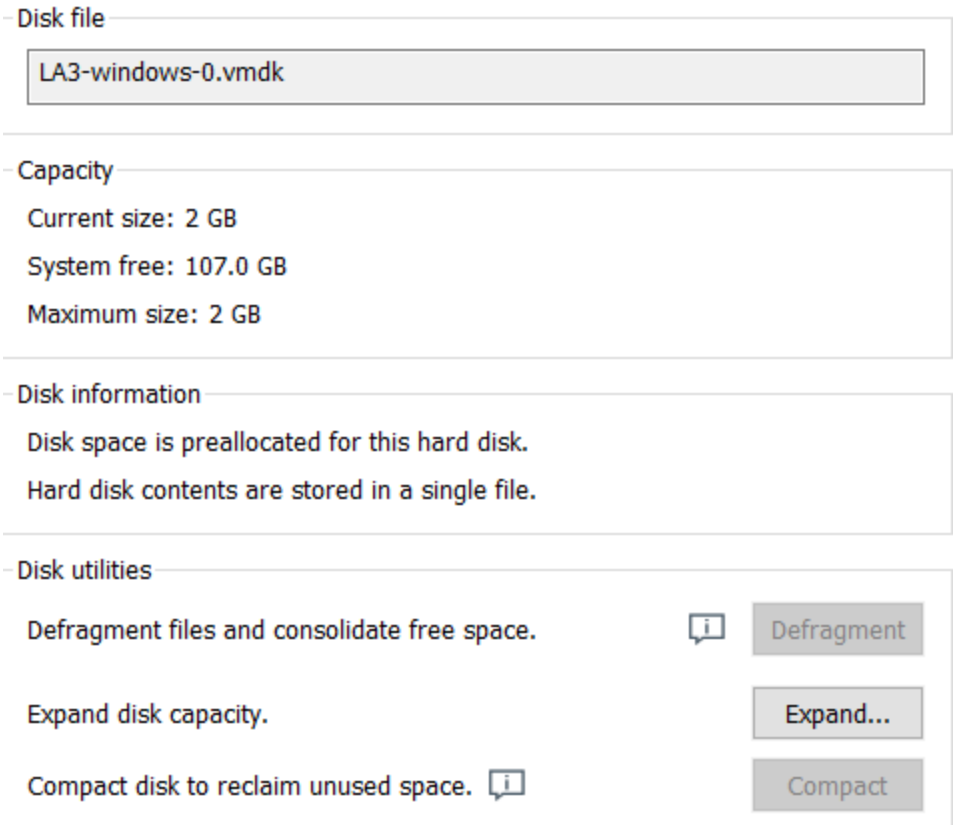
- **Step 4** : Add an extra disk, as shown below :



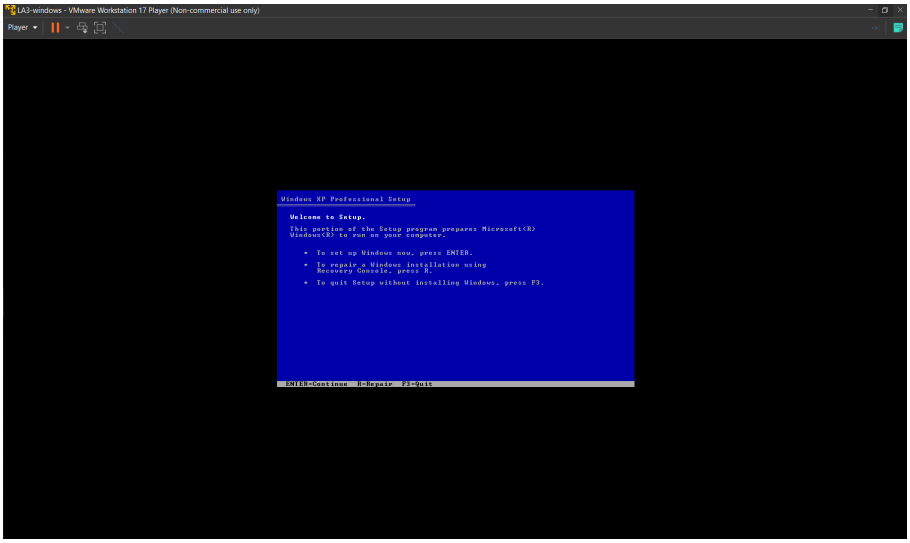
- We allocate all the disk space upfront, and further notice that now we do not have options to fragment, compact, etc. enabled

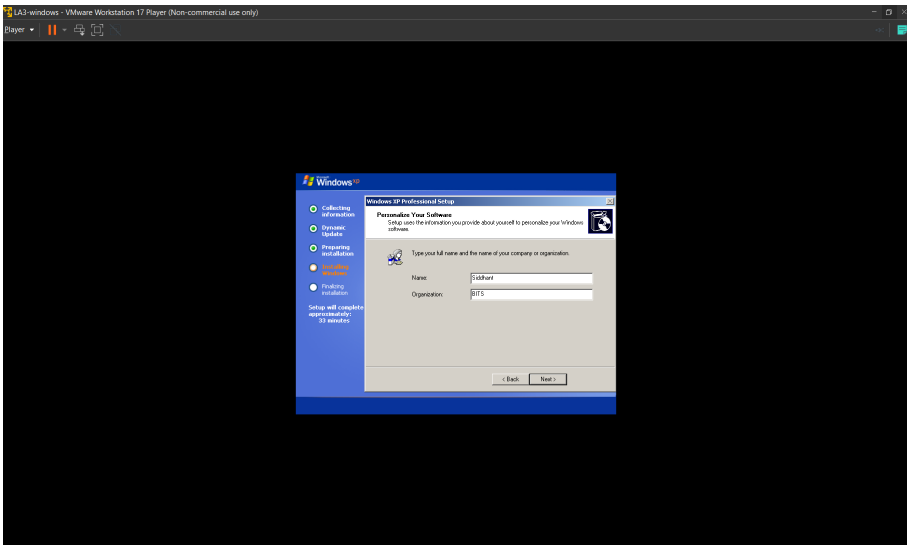

Add Hardware Wizard
Specify Disk Capacity
How large do you want this disk to be?
Maximum disk size (GB):
2
Recommended size for Windows 7: 60 GB
Allocate all disk space now.
Allocating the full capacity can enhance performance but requires all of the physical disk space to be available right now. If you do not allocate all the space now, the virtual disk starts small and grows as you add data to it.
Store virtual disk as a single file
Split virtual disk into multiple files
Splitting the disk makes it easier to move the virtual machine to another computer but may reduce performance with very large disks.
< Back
Next >
Cancel


Hard Disk (SCSI) 62 GB
New Hard Disk (SCSI) 2 GB (Preallocated)



- Step 5 : Start the VM and configure the setup as required





- We can also add shared folders using the “Options” menu in the VM settings. Shared folders can be used to act as a common channel of sharing data between the host and the guest OSes.