

# Basics Lab 1

## Requirements :

- 1) Any Virtualization Software (e.g., VMWare)
- 2) Any Windows ISO (e.g., Windows 11 ISO)
- 3) Any programming language (e.g., Python)

## The Goal :

- You need to write a script that can distinguish between running on a physical computer and in a virtual machine environment.

## The Objective :

- You need to understand how the attacker can detect virtual honeypots.

### The Steps

1 You have to install virtual windows on virtual machine software to test the differences between normal windows and virtual windows.

1) Install any virtualization software, like VMware.

2) Install Windows ISO on this virtualization software.

2 Google about signs to discriminate between virtual machine windows and real windows (e.g., How to tell if VM or physical Windows?).

3 Implement a script with any programming language to detect these signs.

# Solution

## Step 1: Setting Up the Virtual Machine

### 1- Install Virtualization Software:

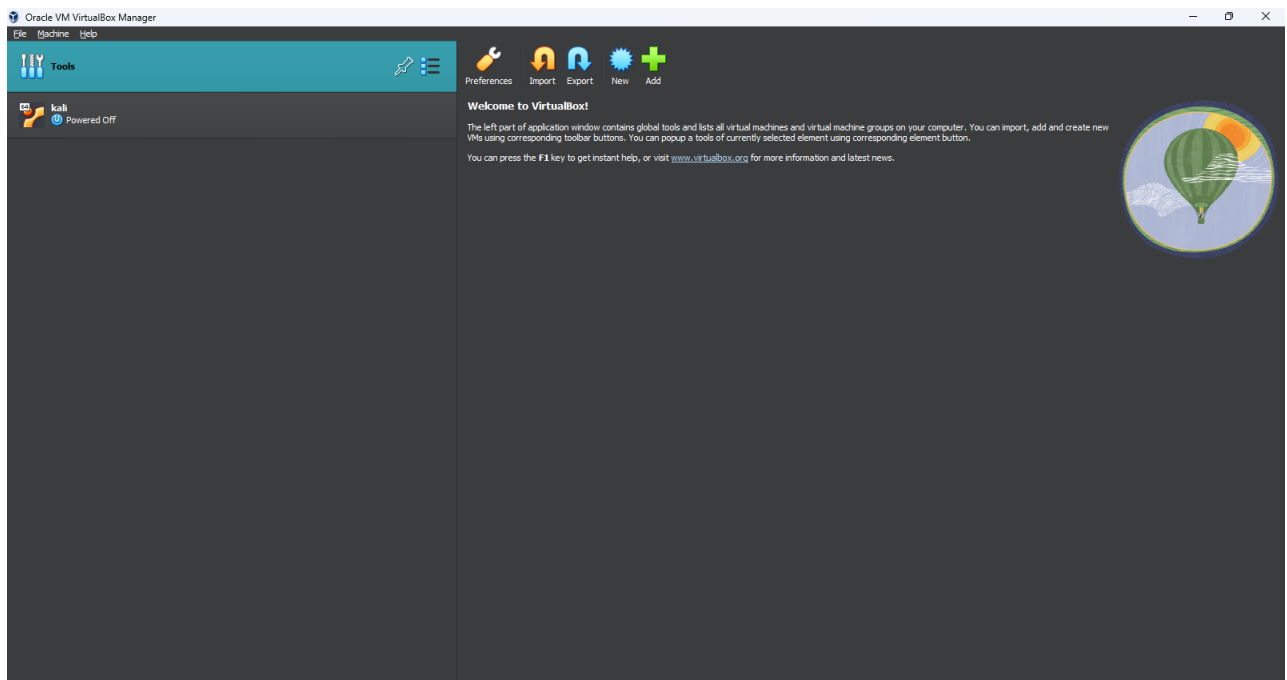
Download and install virtualization software like [VMware](#) or [Virtual Box](#).

### 2- Download Windows ISO:

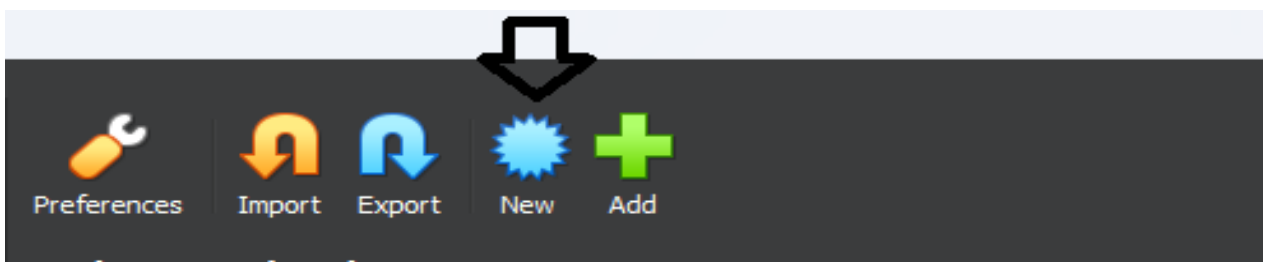
Obtain a Windows ISO file from the [Microsoft website](#).

### 3- Install Windows on the Virtual Machine:

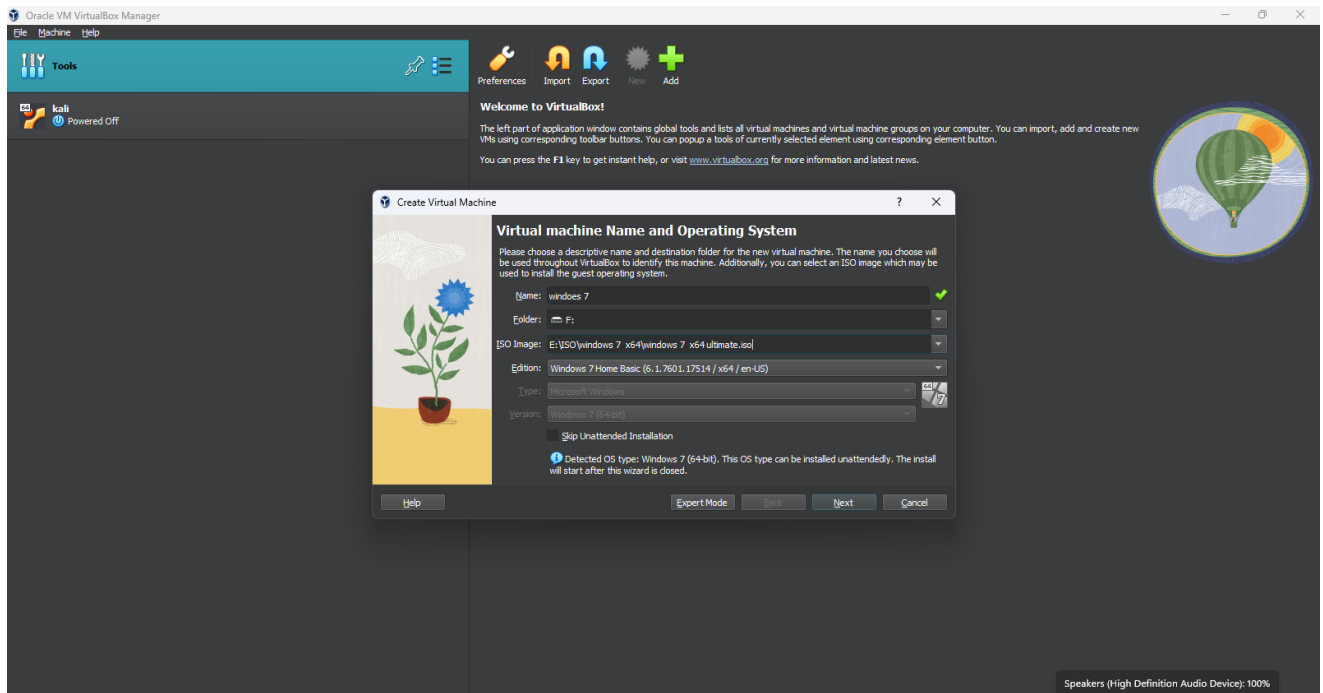
- Open Virtual Box .



- create a new virtual machine.



- Follow the prompts to install the Windows ISO on the virtual machine.



## Step 2: Researching VM Detection Signs

Research ways to differentiate between virtual and physical machines. Here are a few common methods:

- **Hardware Abstraction Layer (HAL) Checks:**
  - Virtual machines often have specific HAL implementations that differ from physical machines.
- **BIOS Information:**
  - VM-specific BIOS strings (e.g., "VMware", "VirtualBox") can be checked.
- **Network Adapter Information:**
  - Virtual machines may have network adapters with MAC addresses that follow specific patterns.
- **CPU Information:**
  - Virtual machines may have specific CPU features or flags.
- **Registry Keys:**
  - Specific registry keys may indicate a virtual environment.
- **Installed Drivers:**
  - Check for VM-specific drivers.

## Step 3: Writing the Detection Script

(Note: We will write the script in the Batch language because the Batch language does not require any requirements in order to work unlike Python.)

Let's get started →

**@echo off:** Disables the display of commands as they are executed in the command prompt.

**setlocal:** Starts a new local environment so that changes to variables do not affect the main environment.

```
1  @echo off
2  setlocal
3
```

**set "downloadsPath=%USERPROFILE%\Downloads":** Sets the path to the Downloads directory for the current user.

**set "outputFile=%downloadsPath%\SystemTypeResult.txt":** Defines the name and path of the file where the results will be written.

```
4  :: Define the Downloads directory path
5  set "downloadsPath=%USERPROFILE%\Downloads"
6  set "outputFile=%downloadsPath%\SystemTypeResult.txt"
7
```

These commands use the **wmic** tool to get system information:

- **manufacturer:** The manufacturer of the computer.
- **model:** The model of the computer.
- **bios manufacturer:** The manufacturer of the BIOS.
- **baseboard product:** The product name of the motherboard.
- **chassistypes:** The type of chassis (system enclosure).

```
8  :: Collect system information
9  for /f "tokens=2 delims==" %%i in ('wmic computersystem get manufacturer /value') do set Manufacturer=%%i
10 for /f "tokens=2 delims==" %%i in ('wmic computersystem get model /value') do set Model=%%i
11 for /f "tokens=2 delims==" %%i in ('wmic bios get manufacturer /value') do set BIOSManufacturer=%%i
12 for /f "tokens=2 delims==" %%i in ('wmic baseboard get product /value') do set BaseBoardProduct=%%i
13 for /f "tokens=2 delims==" %%i in ('wmic systemenclosure get chassistypes /value') do set ChassisType=%%i
14
```

## These commands collect additional information:

- **caption:** The type of video card.
- **name:** The name of the network adapter that is enabled.
- **screenheight, screenwidth:** The dimensions of the screen.

```
:: Collect additional information
for /f "tokens=2 delims==" %i in ('wmic path win32_videocontroller get caption /value') do set VideoCard=%i
for /f "tokens=2 delims==" %i in ('wmic path win32_networkadapter where "NetEnabled=true" get name /value') do set NetworkAdapter=%i
for /f "tokens=2 delims==" %i in ('wmic desktopmonitor get screenheight,screenwidth /value') do set ScreenDimensions=%i

:: Collect CPU information
```

- This command collects information about the CPU.

```
:: Collect CPU information
for /f "tokens=2 delims==" %i in ('wmic cpu get caption /value') do set CPU=%i
```

- This command collects information about the Hardware Abstraction Layer (HAL).

```
:: Collect HAL information
for /f "tokens=2 delims==" %i in ('wmic computersystem get systemtype /value') do set HAL=%i
```

- This command uses reg query to collect registry information and writes it to a temporary file.

```
:: Collect registry information
reg query "HKLM\HARDWARE\DESCRIPTION\System" > "%downloadsPath%\RegistryInfo.txt"
```

- These commands write the collected system information to the output file **SystemTypeResult.txt**

```
:: Display collected information
echo System Information: > "%outputFile%"
echo ----- >> "%outputFile%"
echo Computer System: %Manufacturer% %Model% >> "%outputFile%"
echo BIOS: %BIOSManufacturer% >> "%outputFile%"
echo Base Board: %BaseBoardProduct% >> "%outputFile%"
echo System Enclosure: %ChassisType% >> "%outputFile%"
echo Video Card: %VideoCard% >> "%outputFile%"
echo Network Adapter: %NetworkAdapter% >> "%outputFile%"
echo Screen Dimensions: %ScreenDimensions% >> "%outputFile%"
echo CPU: %CPU% >> "%outputFile%"
echo HAL: %HAL% >> "%outputFile%"
echo Registry Info: >> "%outputFile%"
type "%downloadsPath%\RegistryInfo.txt" >> "%outputFile%"
```

These commands check for signs indicating that the system is running inside a virtual machine by looking for specific keywords in the collected information. If any of these keywords are found, the variable `isVM` is set to `true`.

```
:: Check for VM indications in system information
set "isVM=false"
echo %Model% | findstr /i "VMware VirtualBox VirtualPC Hyper-V" >nul && set "isVM=true"
echo %BIOSManufacturer% | findstr /i "VMware VirtualBox VirtualPC Hyper-V" >nul && set "isVM=true"
echo %BaseBoardProduct% | findstr /i "VMware VirtualBox VirtualPC Hyper-V" >nul && set "isVM=true"
echo %CPU% | findstr /i "Virtual" >nul && set "isVM=true"
echo %HAL% | findstr /i "Virtual" >nul && set "isVM=true"
type "%downloadsPath%\RegistryInfo.txt" | findstr /i "VMware VirtualBox VirtualPC Hyper-V" >nul && set "isVM=true"
if %ChassisType%==1 set "isVM=true"
```

Based on the value of the `isVM` variable, this section of the code writes the final result (whether the system is running inside a virtual machine or on a physical machine) to the output file.

```
:: Output result
if "%isVM%"=="true" (
    echo This system is running inside a Virtual Machine. >> "%outputFile%"
) else (
    echo This system is running on a Physical Machine. >> "%outputFile%"
)
```

This command opens the output file automatically once the script finishes executing.

**endlocal** : This command ends the local environment that was created by `setlocal`, meaning any changes to variables will not affect the main environment.

```
:: Open the output file
start "" "%outputFile%"

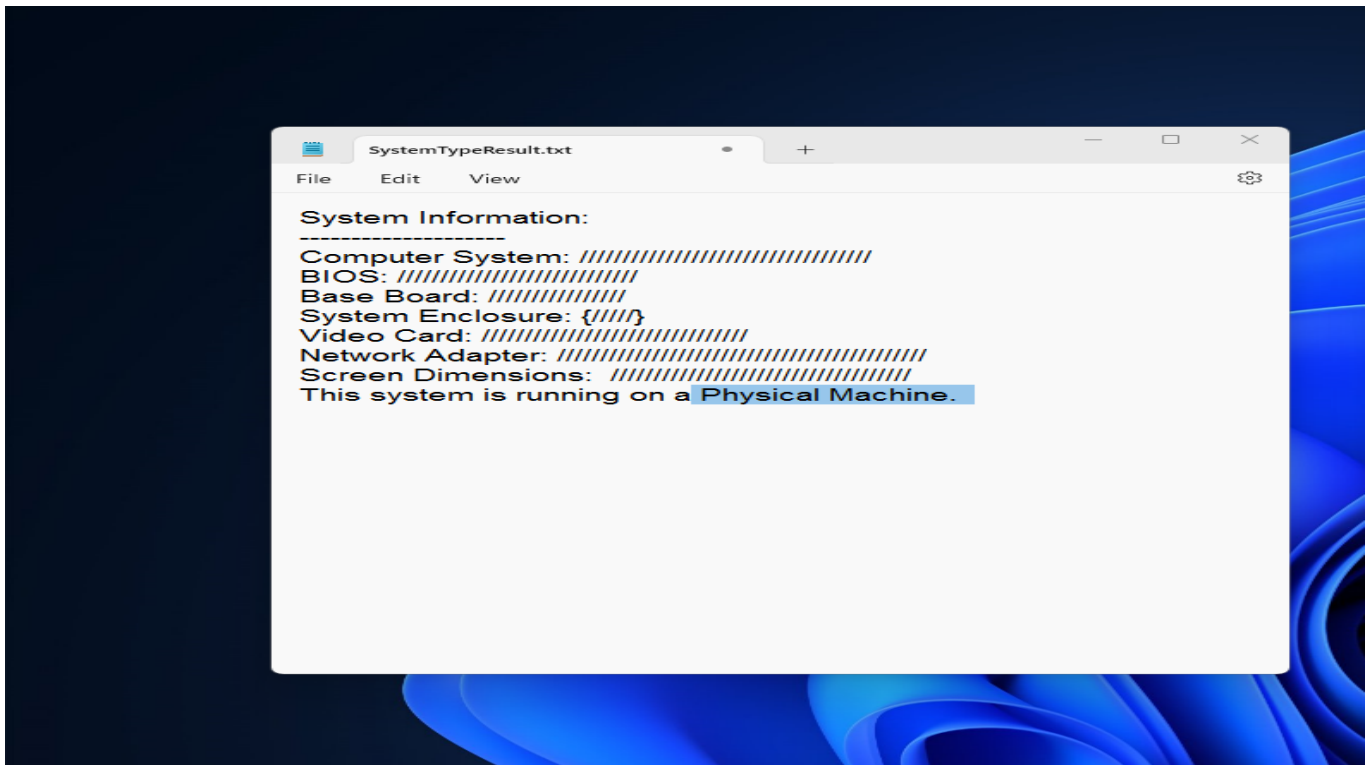
endlocal
```

After finishing, save the file with the path `(.bat)`

# Testing the Script & Proof Of Concept

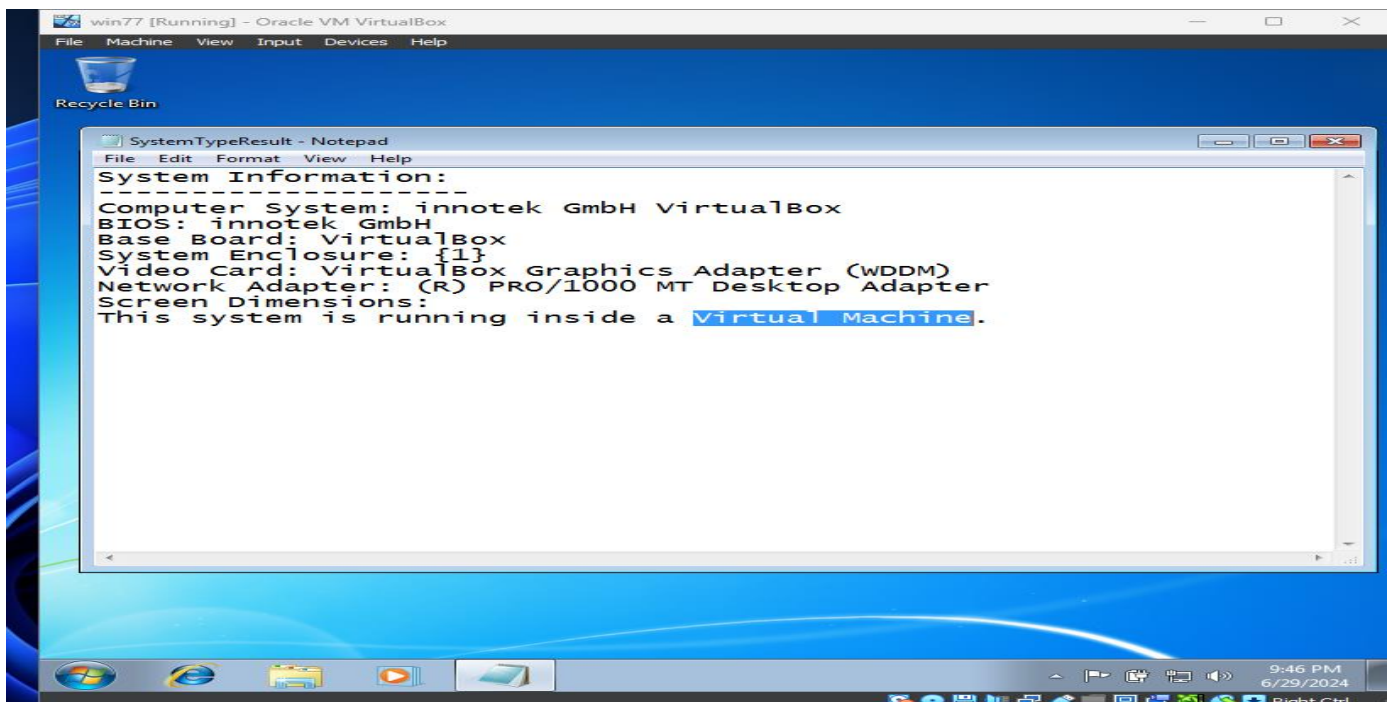
Run the script on both your physical machine and the virtual machine you set up earlier. Compare the outputs to verify that it correctly identifies the environment.

Run the script physical machine :



```
SystemTypeResult.txt
File Edit View
-----
System Information:
Computer System: //////////////////////////////////
BIOS: //////////////////////////////////
Base Board: //////////////////////////////////
System Enclosure: {/////}
Video Card: //////////////////////////////////
Network Adapter: //////////////////////////////////
Screen Dimensions: //////////////////////////////////
This system is running on a Physical Machine.
```

Run the script virtual machine :



```
win77 [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Recycle Bin
SystemTypeResult - Notepad
File Edit Format View Help
-----
System Information:
Computer System: innotek GmbH VirtualBox
BIOS: innotek GmbH
Base Board: VirtualBox
System Enclosure: {1}
Video Card: VirtualBox Graphics Adapter (WDDM)
Network Adapter: (R) PRO/1000 MT Desktop Adapter
Screen Dimensions:
This system is running inside a Virtual Machine.
```