Hacker Academy

CHEESE ALERTY SYSTEM

FORENSIC MODULE-2

Prepared by
Keerthi TR

Mentored By
Ayush Singh

Hacker Academy

INDEX

Hacker Academy

INRODUCTION

In today's digital age, the use of webcams has become ubiquitous, spanning from virtual meetings and online education to security and surveillance. However, with this increased reliance on webcams comes the need for enhanced security and monitoring to ensure that these devices are being used appropriately and securely. The Automated Webcam On/Off Alert System with Email Notifications addresses this need by providing real-time alerts whenever a webcam is activated or deactivated.

This system is designed to monitor the status of a webcam and instantly send an email notification to a predefined recipient whenever there is a change in the webcam's state. Whether the camera is turned on for a scheduled video conference or turned off to ensure privacy, the system ensures that the user is promptly informed. This functionality is particularly beneficial for individuals concerned about unauthorized access to their webcams, as it offers a layer of security by keeping users aware of their webcam activity.

The system operates seamlessly in the background, requiring minimal user intervention once configured. By leveraging this technology, users can maintain greater control over their digital environment, receive timely alerts about webcam usage, and take necessary actions to safeguard their privacy and security.

Hacker Academy

SYSTEM INFORMATION

Host System: Windows 11
Operating System: Windows 11
Processor: Intel 6
RAM:8GB
Storage: SSD (256GB)
Virtualization Support: BIOS/UEFI

Virtualization Software: VirtualBox
Version: Version 7.0.14
Base Memory: 2GB (2048MB) of RAM .
Processors: 2 CPU cores.

Guest System: Ubuntu
Operating System: Ubuntu 20.04 LTS
Processor: Virtual CPU as assigned by VirtualBox
RAM: 2GB
Storage: 20GB
Network Adapter: NAT or Bridged Adapter

INSTALLATION

1.Install VirtualBox on Windows 11:
Download the latest version of VirtualBox from the official Oracle website.
Follow the installation wizard to install VirtualBox on your Windows 11 host system.
After downloading the virtual box downloaded Vb extensions and installed it in the virtual box
2.Create a New Virtual Machine:
Open VirtualBox and click on "New".
Named the virtual machine "Ubuntu (64-bit)" as the version.
Provided details
3.Install Ubuntu:
Start the virtual machine and follow the prompts to install Ubuntu.
During installation, set up your username and password, configure partitions, and complete the installation process.
4.Install Guest Additions:
Once Ubuntu is installed, go to "Devices" in the VirtualBox menu and select "Enabled webcam".
5.Python3
Installed in Ubuntu terminal

Hacker Academy

INSTALLING NECESSARY TOOLS IN UBUNTU

To set up the necessary tools in Ubuntu for our Automated Webcam On/Off Alert System with Email Notifications, follow these detailed steps. This guide will cover updating and upgrading the system, installing Python 3, and installing fswebcam.

Step 1: updating & upgrading ubuntu

Step 1.1: open terminal in the ubuntu

Step 1.2: update the ubuntu by typing the command

$sudo apt-get update

Step 1.3: after update ,upgrade by typing the command

$sudo apt-get upgrade

Step 2: Installing python3

Step 2.1: install python3 by typing this command

$sudo apt-get install python3

(Why We Are Using It: Python is a versatile and widely-used programming language that is essential for scripting, automation, and developing various applications. Python 3 is the latest major version of Python and comes with many improvements and new features compared to Python 2. Installing Python 3 ensures that you can develop and run modern Python applications and scripts.)

Step 3: Installing fswebcam

Step 3.1: install fswebcam by typing this command

$sudo apt-get install fswebcam

(`fswebcam` is a simple and effective command-line tool for capturing images from a webcam. It is lightweight and supports a variety of options for configuring the capture settings, such as resolution, delay, and output format. We use `fswebcam` to automate the process of taking snapshots from the webcam, which can be useful for monitoring, security, and alert systems. By integrating `fswebcam` with other tools like Python, we can create scripts that trigger alerts and notifications based on webcam activity.)

Hacker Academy

SETTING UP SMTP SERVER TO SEND THE MAIL

STEPS TO SET UP AND GET AN APP PASSWORD IN OUR GOOGLE ACCOUNT

Step 1: open Your google account in You are respected browser& signup to your Google account with the mail.

Step 2:In the left sidebar, click on security Go down to see the 2-step verification under “how you sign in to Google”.

Step 3:Click on two step verification, If in case 2-step verification is OFF Make it ON by Providing the necessary details of your mail.

Step 4: Go to app password if you cannot find the app password, go to search button and search for app password Click on it.

Step 5: In app password name your app And generate the password Which contains 16 digit password copy the password(eg: fjiwqpgkurffhmga[remove the gaps in the password]).

SMTP SERVER NAME AND PORT NUMBER

Name:Smtp.gmail.com

Port number:587

Hacker Academy

PYTHON SCRIPT TO GENERATE ALERT MESSAGE AND SEND MAIL

```
import os
import time
import smtplib
from email.mime.multipart import MIMEMultipart
from email.mime.text import MIMEText

# Function to send email
def send_email():
  from_address = "storytalks7@gmail.com"
  to_address = "mamatha.g202020@gmail.com"
  msg = MIMEMultipart()
  msg['From'] = from_address
  msg['To'] = to_address
  msg['Subject'] = "Alert: Webcam is turned on"
  body = "The webcam on your system has been turned on."
  msg.attach(MIMEText(body, 'plain'))
  server = smtplib.SMTP('smtp.gmail.com', 587)
  server.starttls()
  server.login(from_address, "fjiwqpgkurffhmga")
  text = msg.as_string()
  server.sendmail(from_address, to_address, text)
  server.quit()

while True:
  # Check if webcam is in use
  result = os.popen("lsof | grep /dev/video0").read()
  if result:
    print("Webcam is on!")
    send_email()
    time.sleep(60) # Wait for 60 seconds before next check
  else:
    print("Webcam is off")
    time.sleep(5) # Check every 5 seconds
```

Hacker Academy

CREATING A PYTHON FILE , FILE PERMISSION AND EXECUTING FILE

01 Creating a python File

Step 1: open a terminal(ctrl+alt+T).
Step 2:locate the directory where you want to save the file by by cd command.
Step 2:use text editor to create a python file, type this command and file name Monitor_camera.py

$sudo Monitor_camera.py

Step 3:write Your Python script.
Step 4:save & exit: To save the file and exit nano, press Ctrl + X, then Y, and finally "Enter".

02 Give Permissions to the File

Step 1: check the current permission of your file by typing ls-l command.
Step 2:if the file doesn't have execution permission enter this command to get the file to run.

$chmod +x Monitor_camera.py

Step 3: after giving execution permission to file, permission of the file will be this.

-rwxrwxr-x 1 yourusername yourusername 45 May 18 10:00 Monitor_camera.py

03 Execute The Python File

Run the Python Script Using Python Interpreter, enter the following command

python3 Monitor_camera.py

OR

Run the Python Script Directly, Since the file is now executable, you can run it directly from the terminal

./ Monitor_camera.py

Hacker Academy

TROUBLESHOOTING OF WEBCAM IN UBUNTU

If Cheese in Ubuntu is showing "No device found," it indicates that the webcam is not being detected by the system. To overcome this, ensure that the webcam is properly enabled and recognized in VirtualBox by adding a USB filter for the webcam and confirming its connection in the Devices menu of the running VM. Additionally, you can check if the necessary drivers are installed in Ubuntu , below there are step's to troubleshoot this problem:

1. Install the VirtualBox Extension Pack:

1.1-Download the Extension Pack:

· Open your web browser and go to the VirtualBox download page.

· Find the section for the "VirtualBox Extension Pack" and download the version that matches your installed version of VirtualBox.

1.2-Install the Extension Pack:

· Open VirtualBox.

· Go to File > Preferences > Extensions.

· Click on the Add new package icon (a blue diamond with a green plus).

· Attach the extension file downloaded and install it and restart virtual machine and ubuntu.

2. Enable the Webcam in VirtualBox

Select your Ubuntu VM in VirtualBox and open its Settings. Navigate to the USB section and ensure Enable USB Controller is checked. Choose either the USB 2.0 (EHCI) Controller or USB 3.0 (xHCI) Controller, depending on your host's capabilities. Click the Add Filter From Device icon and select your webcam from the list to create a filter for it. Apply the changes by clicking OK.

3.Enable the webcam in virtual machine

3.1- Start Your Ubuntu VM:

· Click on 'Start' to boot up your Ubuntu VM.

3.2- Access Devices Menu:

· Once Ubuntu is running, go to the top menu bar of VirtualBox. (if you can't find where is menu bar refer page:17 and figure 4.2).

· Click on Devices > Webcams.

· In the 'Webcams' submenu, you should see your webcam listed. Click on it to enable it for the VM.

4. Verify Webcam Functionality in Ubuntu.

4.1- Open a Terminal:

· In Ubuntu, press Ctrl + Alt + T to open a terminal.

4.2- Install Cheese (Optional):

· If you don't have a webcam application installed, you can use Cheese. Install it by running this command

$sudo apt install cheese

4.3- Launch Cheese:

· Open Cheese from the applications menu or by running cheese in the terminal. This application should detect your webcam and show the video feed.

Hacker Academy

TROUBLESHOOTING OF WEBCAM IN UBUNTU

Open the terminal and run this command $python Monitor_camera.py file
When you execute the file Monitor_camera.py using the command python3 Monitor_camera.py, the script begins its process of continuously monitoring the status of the webcam and sending notifications in mail when the webcam is turned on.

Code Break Down

1.Importing Required Modules

```
import os
import time
import smtplib
from email.mime.multipart import MIMEMultipart
from email.mime.text import MIMEText
```

The script starts by importing necessary modules:

import os
import time
import smtplib
from email.mime.multipart import MIMEMultipart
from email.mime.text import MIMEText

·os: Provides a way of using operating system-dependent functionality like reading from the command line.
·time: Allows the script to pause execution for specified intervals.
·smtplib: Defines an SMTP client session object that can be used to send mail to any Internet machine with an SMTP or ESMTP listener daemon.
·email.mime.multipart and email.mime.text: These modules are used to create email messages with multiple parts (in this case, plain text).

2. Defining the Email Sending Function

The send_email function is defined to handle sending email notifications when the webcam is detected to be turned on:

```
def send_email():
from_address = "storytalks7@gmail.com"
 to_address = "mamatha.g202020@gmail.com"
 msg = MIMEMultipart()
 msg['From'] = from_address
 msg['To'] = to_address
msg['Subject'] = "Alert: Webcam is turned on"
 body = "The webcam on your system has been turned on."
msg.attach(MIMEText(body, 'plain'))
 server = smtplib.SMTP('smtp.gmail.com', 587)
server.starttls()
server.login(from_address, "fjiwqpgkurffhmga")
 text = msg.as_string()
server.sendmail(from_address, to_address, text)
server.quit()
```

The send_email function is defined to handle sending email notifications when the webcam is detected to be turned on:

The send_email function sets up the email parameters, including the sender and receiver addresses, subject, and body of the email.

·It uses the smtplib module to connect to Gmail's SMTP server (smtp.gmail.com), logs in using the provided credentials, and sends the email.

3. Main Loop to Monitor Webcam Status
The script enters an infinite loop to continuously check if the webcam is in use:

```
while True:
  # Check if webcam is in use
  result = os.popen("lsof | grep /dev/video0").read()
  if result:
print("Webcam is on!")
send_email()
time.sleep(60) # Wait for 60 seconds before next check
  else:
print("Webcam is off")
time.sleep(5) # Check every 5 seconds
```

·The ‘os.popen("lsof | grep /dev/video0").read()’ command is used to check if the device file for the webcam (/dev/video0) is being accessed by any process.
·‘lsof’ is a command-line utility that lists open files and the processes that opened them. If the webcam is in use, it will show up in this list.
·grep /dev/video0 filters the output to only include lines that contain /dev/video0, which is the device file for the webcam.
·If the result is not empty (if result:), it means the webcam is currently in use:
·The script prints "Webcam is on!" to the console.
·The send_email function is called to send an email notification.
·The script then waits for 60 seconds (time.sleep(60)) before checking again to avoid sending multiple emails in a short period.
·If the result is empty, it means the webcam is not in use:
·The script prints "Webcam is off" to the console.
·It waits for 5 seconds (time.sleep(5)) before checking again.

This script continuously monitors the webcam's status by checking if the device file /dev/video0 is in use. When the webcam is turned on, it sends an email notification to a specified address. The script uses simple shell commands, email handling, and a loop with time delays to achieve this functionality.

Execution Flow Steps
Here is an execution flow of what might happen after running the script.

1.Script start:
·You run python3 Monitor_camera.py.
·The script initializes and enters the monitoring loop.
2. Initial Check:
·The script checks if the webcam is in use:
·If not in use, it prints "Webcam is off" and waits for 5 seconds.
3. Webcam Turned On:
·You open an application that uses the webcam (e.g., cheese).
·The script detects that /dev/video0 is in use during the next check.
·It prints "Webcam is on!" and calls the send_email function.
·The email is sent to the specified recipient notifying that the webcam has been turned on.
·The script waits for 60 seconds before the next check.
4. Webcam Still On:
·The webcam remains in use. The script detects it again after 60 seconds but does not send another email immediately because it still waits for the next interval.
·The cycle repeats as long as the webcam is in use, with checks every 60 seconds.
5. Webcam Turned Off:
·You close the application using the webcam.
·The script detects that /dev/video0 is no longer in use.

Hacker Academy

PROFE OF CONCEPT (POC)

INSTALLING PYTHON3 AND FSWEBCAM

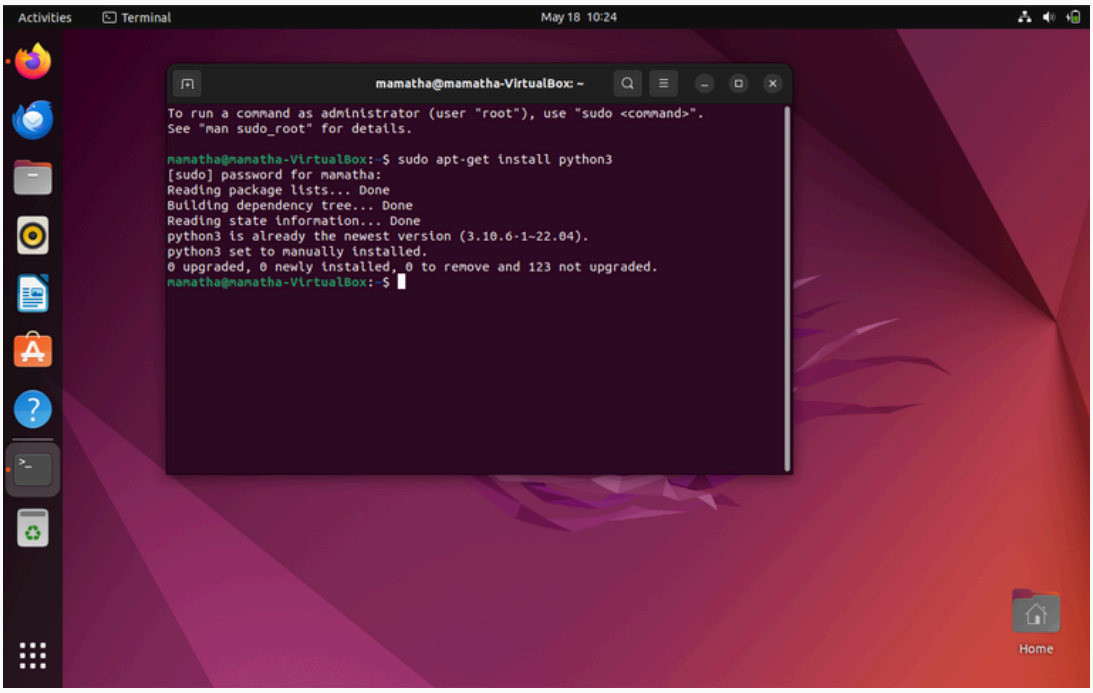

Figure 1.1: installing python3 by typing the command $sudo apt-get install python3.

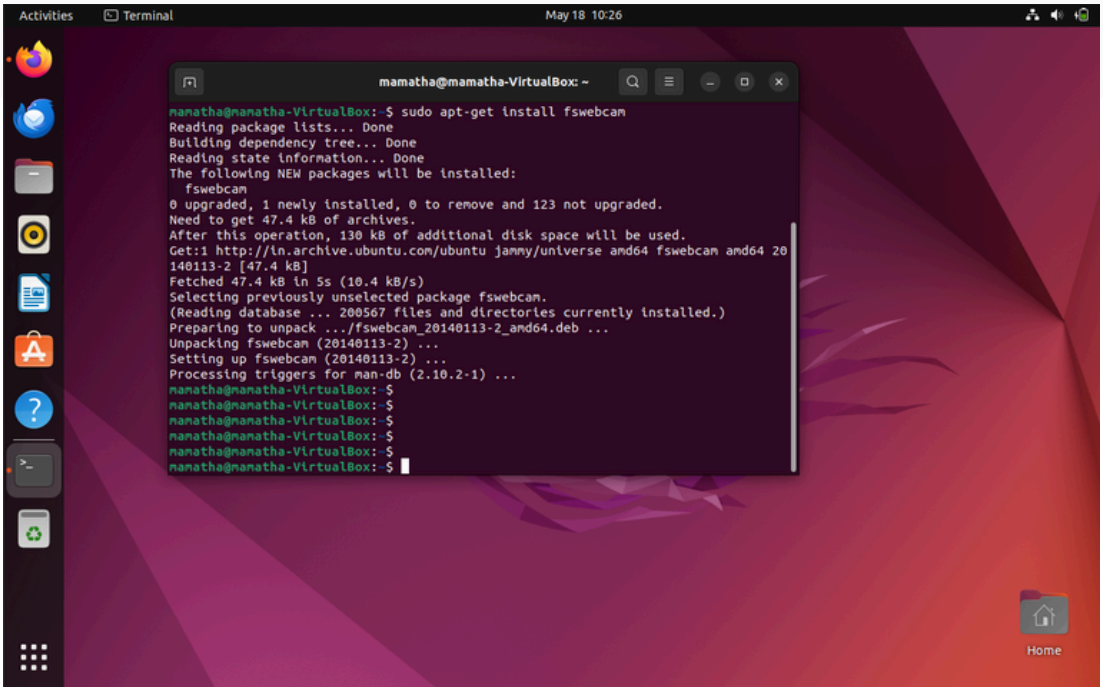

Figure 1.2: installing fswebcam by typing the command $sudo apt-get install fswebcam.

WRITING SCRIPT AND SAVING

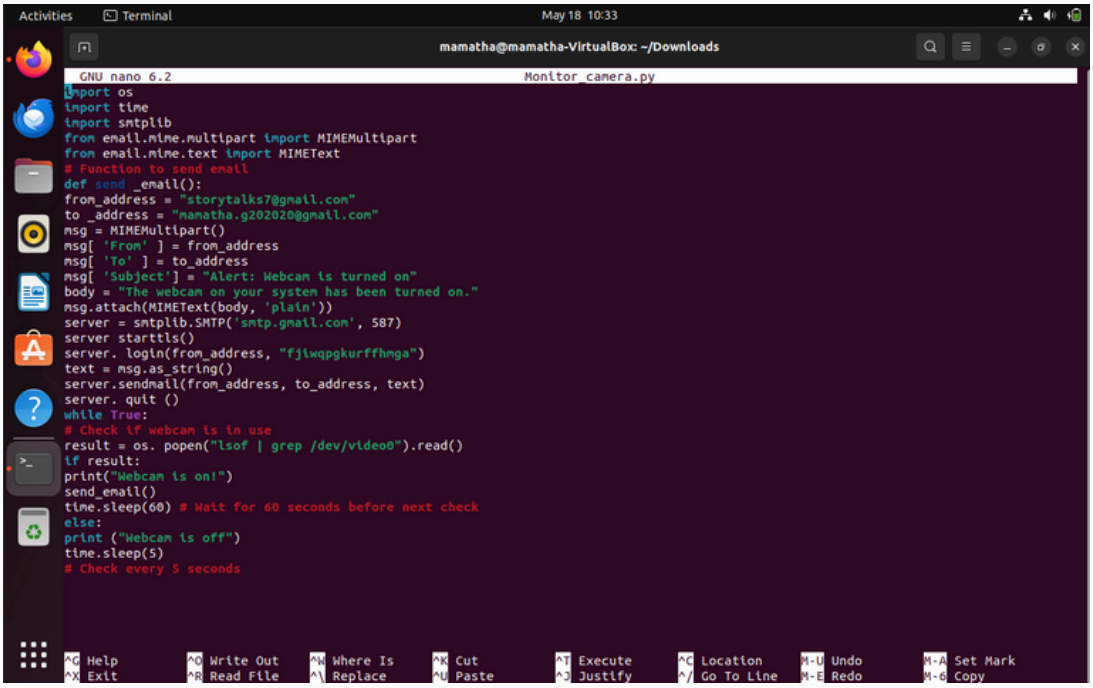

Figure 2.1: opening and writing script by typing $nano Monitor_camera.py.

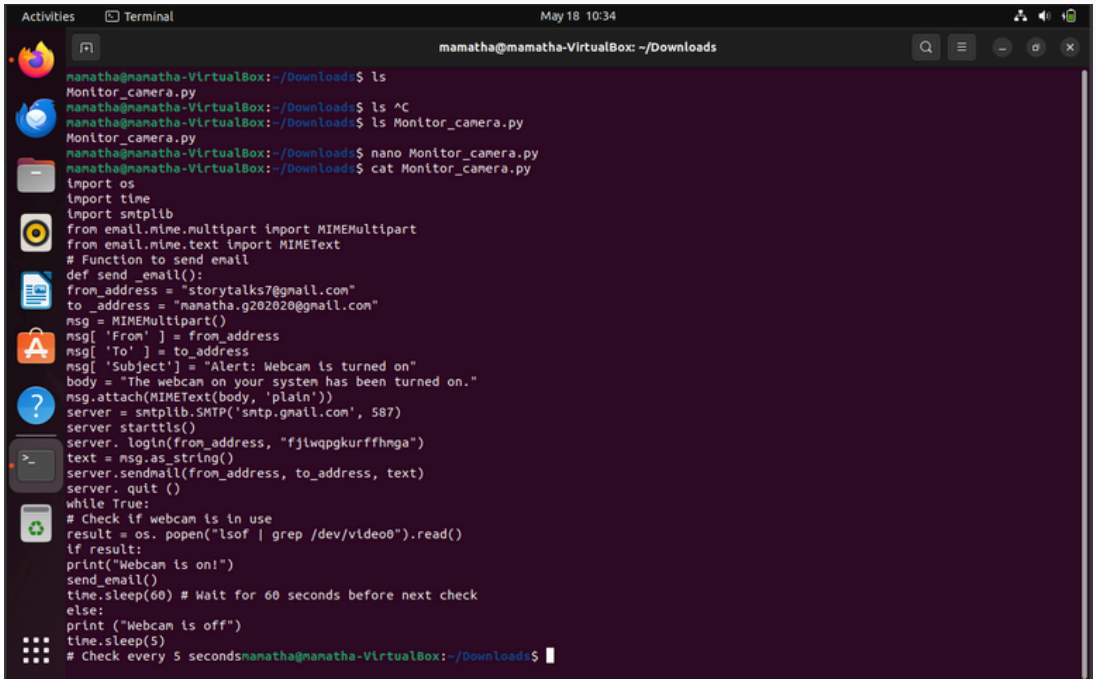

Figure 2.2: Save and exit, press Ctrl + X, then Y, and finally “Enter”.

CHANGING PERMISSIONS OF FILE

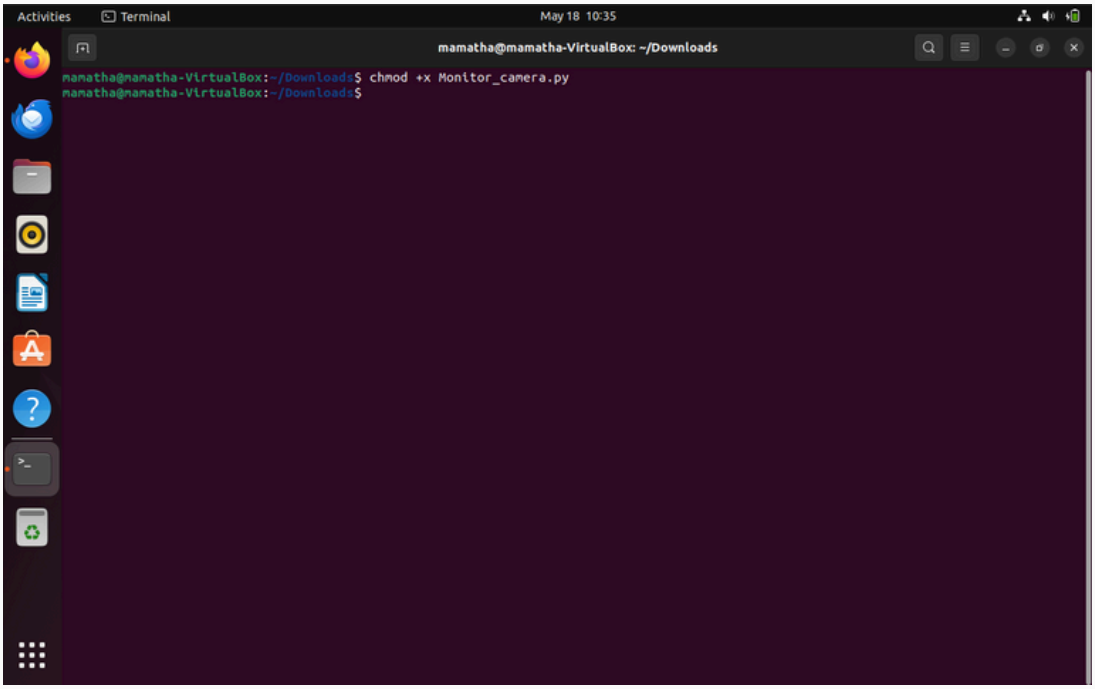

Figure 3.1: changing the permission of the file to execute.

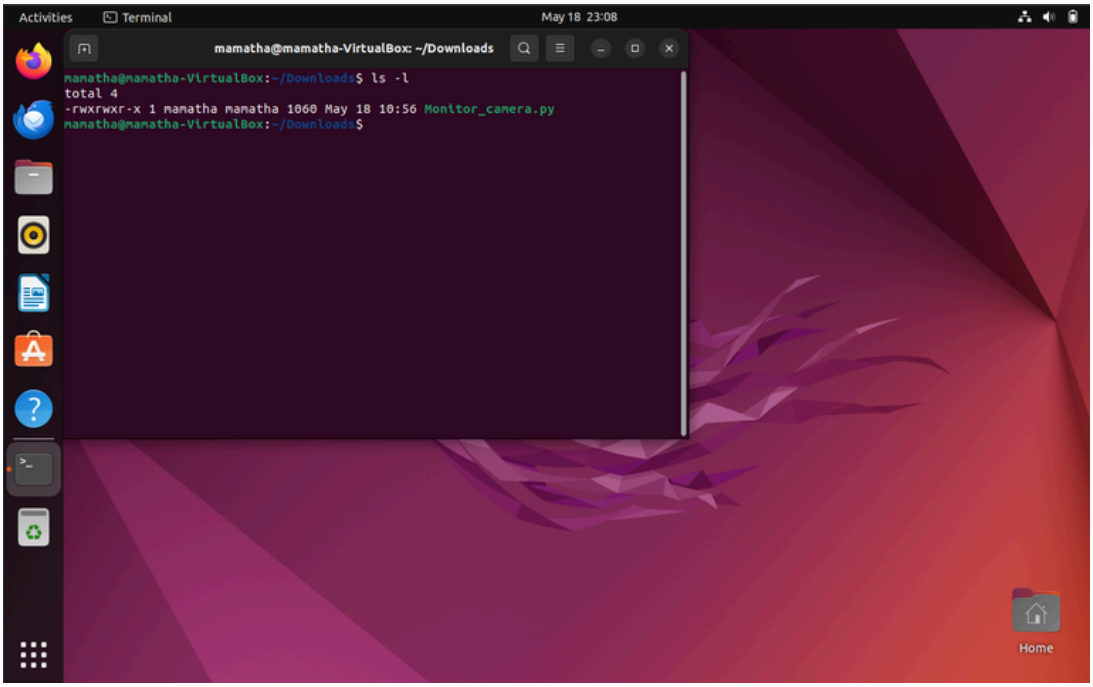

Figure 3.2: permissions has changed t the python file.

ROUBLESHOOT OF WEBCAM

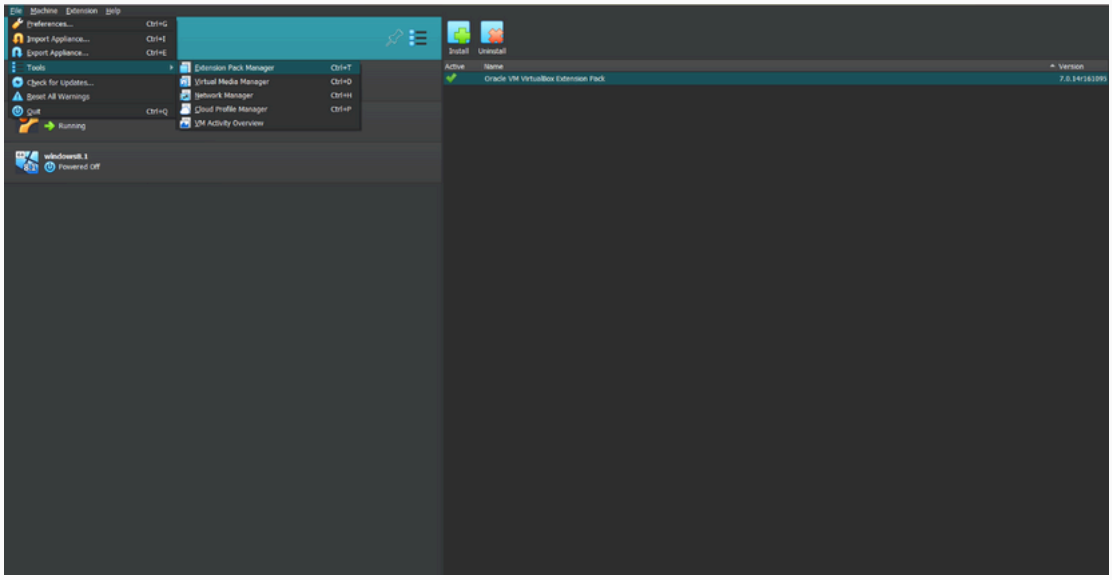

Figure 4.1: attaching extension file in virtual box.

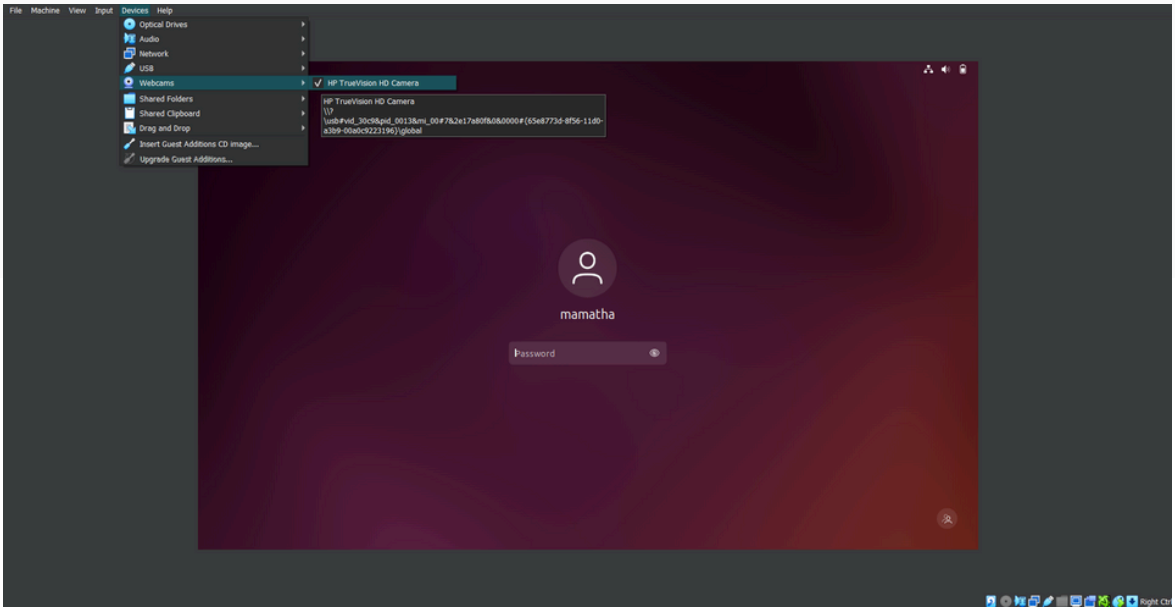

Figure 4.2: Enabling webcam in virtual machine.

EXECUTING THE Monitor_camera.py FILE

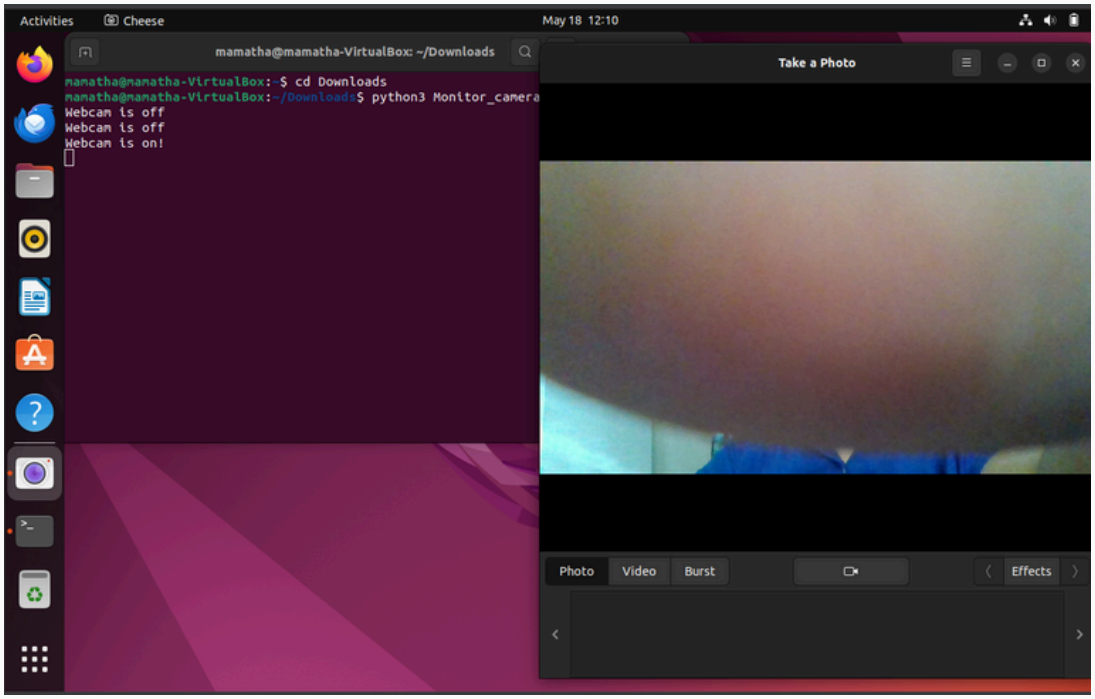

Figure 5.1: executing the python file by typing $python Monitor_camera.py

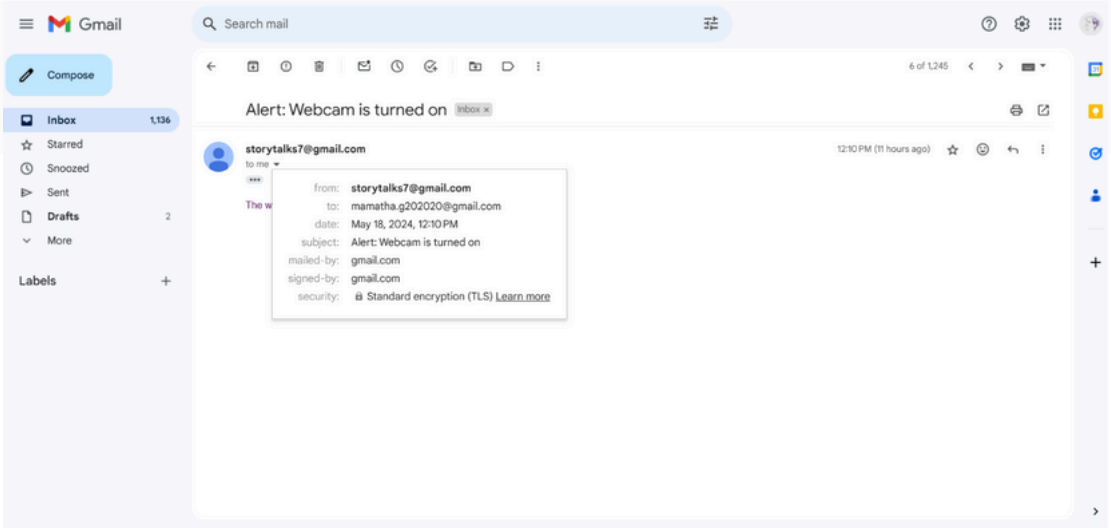

Figure 5.2: mail sent when the camera is triggered.

CONCLUSION

The implementation of a Webcam Activation Alert System with Email Notifications serves as a vital tool in enhancing security, privacy, and monitoring in various settings. This system addresses several critical needs and offers multiple benefits, making it a valuable addition to both personal and professional environments.

Enhanced Security and Privacy

In an era where digital security threats are pervasive, ensuring the privacy of individuals is paramount. Webcams, while essential for communication and surveillance, also pose a potential privacy risk if accessed without authorization. The Webcam Activation Alert System mitigates this risk by providing real-time notifications whenever the webcam is activated or deactivated. This immediate awareness allows users to quickly respond to unauthorized access, ensuring that their privacy is maintained. By sending alert emails, the system provides a documented trail of webcam usage, which can be crucial for auditing and investigating any security breaches.

Proactive Monitoring

For organizations and individuals concerned with security, the ability to monitor webcam activity proactively is invaluable. The system enables continuous monitoring without the need for manual checks. Whether used in a home office, a corporate environment, or a public setting like a school or library, the system ensures that any unusual or unauthorized webcam activity is promptly flagged. This proactive approach helps in preventing potential security incidents before they escalate.

User Awareness and Control

Empowering users with the knowledge of when their webcam is in use fosters a greater sense of control and awareness over their digital environment. In workplaces, it ensures that employees are aware of surveillance measures, thereby promoting transparency. For individuals, it provides peace of mind knowing that they will be immediately informed of any webcam activity, allowing them to take swift action if needed.

Automation and Efficiency

The integration of automated email notifications streamlines the process of monitoring and alerting. Automation reduces the need for manual intervention, allowing users to focus on other tasks while remaining confident that any significant activity will be reported. This efficiency is particularly beneficial in environments where multiple webcams are in use, as it centralizes the monitoring process and ensures consistent, timely alerts.

Versatility and Adaptability

The Webcam Activation Alert System is versatile and can be adapted to various operating systems and environments. Whether implemented on a personal computer, a shared workstation, or a server, the system can be customized to meet specific needs. The use of Python and tools like `fswebcam` on Ubuntu provides a flexible and powerful framework that can be easily modified and expanded to include additional features such as logging, detailed reporting, and integration with other security systems.