Security Assessment of Surveillance Systems

A B. Tech Project Report Submitted
in Partial Fulfillment of the Requirements
for the Degree of

Bachelor of Technology

by

Devansh Gupta
(170101022)

under the guidance of

Dr. Sukanta Bhattacharjee

to the

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY GUWAHATI
GUWAHATI - 781039, ASSAM

CERTIFICATE

*This is to certify that the work contained in this thesis entitled "**Security Assessment of Surveillance Systems**" is a bona fide work of **Devansh Gupta** (**Roll No. 170101022**), carried out in the Department of Computer Science and Engineering, Indian Institute of Technology Guwahati under my supervision and that it has not been submitted elsewhere for a degree.*

supervisor: **Dr. Sukanta Bhattacharjee**

Assistant/Associate Professor,

April, 2021 Department of Computer Science & Engineering,

. Indian Institute of Technology Guwahati, Assam.

Acknowledgements

This prototype is built after studying various researches done by different scientists and authors. I want to thank all those researchers for their valuable contribution in this field. I would like to thank my supervisor Dr. Sukanta Bhattacharjee for continuously guiding me and helping me where help was required. I would also like to thank my mentor Dippojwal Ray for regularly motivating me. I am thankful to all my friends and colleagues for enriching me with the knowledge they have in these topics.

Abstract

Wireless Sensor Networks are becoming popular and are being deployed in various Internet-of-Things (IoT) devices these days. Raspberri Pi is the small and cost-effective device used for home and personal security, data collection, smart home projects and some times hacking applications. In this project we are building low cost, technically advanced and secure surveillance system using Raspberry Pi. Most of the surveillance systems available at present assumes motion as sufficient condition for threat, but this system records the activities based on object-detected and in the night it records the activities based on motion. This model process footage capture by Raspberry Pi Camera module in real time, and this model detects any weapon such as gun, the model notifies its owner. In the night it uses the PIR motion sensor to detect motion and if PIR sensor detects motion it switches on the lights and alerts owner simultaneously. Also we are going to do security assessment on this model, this will ensure the security of the surveillance system. Raspberry Pi is the most frequent used device used for IoT applications. Thus Security of Raspberry pi would be a major concern. Different software and hardware vulnerabilities that can be found in Raspberry Pi is analysed in this paper. Also Wireless networks have various security threats due to the open channel communication model. Due to several limitations (low power, small, less memory, space for storage, time complexity, less processing power) posed by Wireless Sensor Networks, the implementation of security channel is a very complex process in the internet of things. Thus, Wireless Sensor Networks require light weight cryptography algorithms such as ECC, in the implementation of security channel.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Today security systems had become an one of the basic necessity in all sectors. Crime by voluntary causing hurt by dangerous weapons or means had increased significantly i.e crime rate 11.1 in 2017 increased to 11.9 in 2018. Percentage share in IPC (Indian Panel Code; Official criminal code in India) crimes is 5%. Incidence in 2018 1,57,899. With the rise in crimes such as burglary and gun violence the use of surveillance systems is also drastically increasing. Advancement of technology can provide people a way to fulfill their security purposes. Surveillance Systems have proved themselves very efficient in preventing crimes and are playing a very vital role in today's world.

Although most of the CCTV systems are not very efficient as they record and monitor all the time, this makes them consume more power and memory wastage. Also, most of the CCTV systems are not that much advance to give alert explicitly on detecting some kind of threat and also they are not portable and are not easy to deploy. Some smart security systems are available in market such as Retina scanner, fingerprint scanner, IR lasers and RFID systems but they are not affordable by common-man as a result such systems are used for serving the security purposes for small scale applications. There can be variety of approaches for developing the surveillance systems but in this project we tried to make cost-efficient surveillance that can prevent major crimes happening according to

the National Crime Records Bureau Report 2018.

Crimes	Crime Rates (in years)		%age in IPC	Total Cases
	2017	2018		
Theft	45.7	47.3	20%	6,25,441
Burglary	8.6	7.6	3.2%	98,629
Robbery	2.4	2.3	1%	30,822

Table 1.1 National Crime Records Bureau Report 2018

Raspberry Pi had gained world attention due to its cost efficiency, small size, portability,potential and flexibility making it a perfect IoT device. Several researchers and scholars have tried to build surveillance system using Raspberry Pi as you can see in previous works section but our model is unique in its own as it uses latest technologies and large number of features by using the limited number of resources available in that cost.

Raspberry Pi used in this prototype is configured with Ubuntu Server 20.04.2 LTS OS and wifi network for communicating with internet. This Prototype will process footage captured by Raspberry Pi Camera Module in real time by using Tensorflow models and will simultaneously host that footage on the web portal using flask technology, user is required to make his/her own account on that portal and while signing up user requires server ip and server password to see the footage captured by that server. Usee can see the live footage of the camera any time and anywhere on the web and can also take the snap of footage if he wants and can save it. During day time if prototype detect weapon then it will send alert mail to its owner. At night (here night refers to the time when no activity is expected at the place under surveillance), due to the absence of a light source it is hard to process image and it will be cost inefficient to use flood lights all time so, when motion is detected by PIR Motion Sensor then only it will switch on the lights and Raspberry Pi Camera Module and will host the live footage on that portal.

Also, in this technically advanced world where there are millions of videos available free on the internet on how to hack the security systems, the security of the security systems is becoming the important topic to think upon. Also some researchers say that enough

security assessments are not done on Raspberry Pi to be used on such large scale. Several researchers had tried to do security assessments on Raspberry Pi and tried to make wireless sensor networks more secure.

Raspberry Pi's are the small computers and have been used for various applications and also can be integrated within the networks due to which it is important to assess it's weaknesses. There are various standard assessment procedures available for accessing Raspberry Pi's security, one of the famous is NIST (The National Institute of Standards and Technology) Special Publication (SP) series. NIST risk standard is widely applied and accepted in various applications and hardware devices. As Raspberry Pi is used for IoT applications, digital forensics procedures are being developed in case of cyber attacks. OS need to be installed on Raspberry Pi, using default installation of OS can create several hardware and software vulnerabilities. There can be physical attack along with the cyber attack example in some cases analysis of power consumption or noise produced can used for finding secret keys which can be further exploited to attempt attack on the system, such types of attacks come under side-channels attacks.In case of hardware analysis, USBs Poer and Backfeeding, overclocking, GPIO logic levels and serial access, real time clock absence and xenon flash shyness features can be included during assessment. In case of software analysis, security achieved by using different OS can be assessed. Securing a Wireless Sensor Networks poses unique challenges due to high-volume data transfer, the limitations of the motes (small, low power sensor nodes that combine to form a WSN), the accessibility of the locations in which they are deployed, in IoT non-standard communication protocol causes security threats due to various problems of data authentication and authorization and these networks are one, in which the dynamic number of nodes can join and withdraw during the data transfer mechanisms. Because of these obstacles traditional security mechanisms are not suited to wireless networks as these techniques have specific limitations in terms of complexity and strength, such as time complexity, memory, energy, and space for storage. So, lightweight cryptography algorithms such as ECC are required for the security of these

network systems. In this project we tried to different aspects of security. The systems requires authentication from user in case any user tries to access live streaming of footage and various steps are taken to make our system secure. Also we had not hard coded any data/information in the code, all data/information are taken dynamically while running the application. We had used Ubuntu Server 20.04.2 LTS as user, as this OS is way more secure than the OS (Raspbian) which is commonly used in Raspberry devices. Also Ubuntu is open source project and we can configure it or change its code according to our requirements for providing security to the system. In this project we are the building low cost,portable, secure and advanced security surveillance system.

Chapter 2

Review of Prior Works

There are many surveillance security systems with raspberry pi available on the web but but most of them uses only one either Raspberry Pi Camera Module or PIR Motion Sensor. In project only using PIR Motion Sensor they assume motion as a sufficient condition for predicting danger/threat but motion in the day-time can be because of various reasons with no danger/threat involve and it will be annoying to repeatedly getting alert without the presence of any threat. In project only using Raspberry Pi Camera Module it is not possible to detect danger/threat in dark/night, this can be overcome by using night-vision camera but it will again add cost to our system and one solution can be use high intensity flood lights all time during the dark/night and continuously monitor using by Raspberry Pi Camera Module but this makes our system more resources consuming which will lead to increase in cost of using it and it is also leading to wastage of resources.

Some of the projects used face recognition technology for detecting threat but this limits there usage because they can't be used in the public places or near the public places as there will be lots of new faces for the system. Some projects used motion detection technology but motion during the working/day time can't be considered as threat.

Some of the projects used weapons detection techniques which are not that accurate like using cascade classifier. Some of the projects had used tensorflow technology which is

better and recent technology used for weapon detection but they didn't provided live footage streaming on the internet. Almost all the projects used Raspbian OS which made their system slower because lot of softwares and libraries are pre-installed on it even though they didn't use those softwares. We used latest Ubuntu Server 20.04.2 LTS as OS and installed only those libraries which we are using making it fast. So our model is unique in its own providing variety of features by using the limited number of resources available in that cost.

Various researchers and scholars had tried to assess security of Raspberry Pi and introduced procedure/steps to make Raspberry Pi more secure. Digital Forensics Procedures are developed by scientists to trace the digital forensics evidence in case cyber attacks happens on Raspberry Pi. Researchers also made some changes in traditional cryptography method used for encrypting data to make the system resistant against some side channel attacks. Some researchers had done security assessment of the Raspberry Pi on the basis of standard provided by NIFT. Hardware and Software features of Raspberry Pi were also assessed by some researchers. Since, due to various limitations imposed by Raspberry Pi or WSN's traditional cryptography algorithm can't be used over data transmission, so the researchers had also tried to develop the new cryptography algorithms which can be used in small devices like Raspberry Pi.

- M.Surya Gupta,Virginia ,Vamsikrishna Patchava had implemented a system which continuously monitors motion, if the system detects motion it turns on the light and captures the screenshots and sends those screenshots to the owner of the system.[MSDG15].

- Chinmaya Kaundanya,Omkar Pathak,Akash Nalawade,Sanket Parode have implemented system on raspberry pi which recognizes faces and if it founds new face then it gives notification to owner[CK17].

- Huu-Quoc Nguyen, Ton Thi Kim Loan, Bui Dinh,Eui-Nam Huh have implemented surveillance system in that user can real time monitor camera and also the model was able to detect motion[HQN15].

- S Sruthy and Sudish N George implemented system whose prime features were Remote user alerts, live video streaming and portability[SS17].
- Onafeso Babatunde and Liu E had developed the procedure to find digital forensic evidence in case cyber security attack happen on the system.[BE]
- Akihiro SANADA, Yasuyuki NOGAMI, Kengo IOKIBE and Md. Al-Amin Khandaker had made some changes in RSA cryptography algorithm to resist Side Channel Attacks. [ASK19]
- Michael Williams had done a risk assessment on Raspberry Pi using NIST standards. [Wil18]
- Jorge Sainz-Raso, Sergio Martin, Gabriel Diaz, and Manuel Castro had analysed the security vulnerabilities (both harware and software) in Raspberry Pi. [JSRC19]
- B. Vanathy and M. Ramakrishnan had introduced dynmic key distribution management using key escrow based ECC algorithms in Mobile Ad-hoc networks. [BV20]
- Rakesh Sharma, Brahmjit Singh and Poonam Jindal had done Study and analysis of various different key generation techniques in internet of things.[RSS20]
- J. Louw, G. Niezen, A.M. Abu-Mahfouz and T.D. Ramotsoela had introduced key distribution scheme using elliptic curve cryptography in Wireless Sensor Networks. [JLTR16]
- Sarmad Ullah Khana, Claudio Pastroneb, Luciano Lavagnoa, Maurizio A. Spiritob had introduced authentication and key establishment scheme for the IP-based wireless sensor networks.[SUK11]
- eliu01001 github used Tensorflow to detect the weapons on webcam and in videos[eli19].
- seraj94ai github user implemented live streaming of processed video using flask[ser19].

- namdothanh github user implemented home security system in which user gets mail on Gmail and gets message on the messenger when motion is detected and also user have facility to control the system through messenger[nam17].

Chapter 3

Motivation and Features

3.1 Motivation

Over the years there is significant increase in crimes such as theft, burglary and robbery. Surveillance systems using Raspberry Pi can provide great security by alerting as they can detect the possibility of threat beforehand. These models are easy to deploy, small, portable and most importantly affordable as a result can be used majority of people anywhere they want. Also Raspberry Pi has arisen as the most common IoT device for educational purposes as well as for industrial purposes now-a-days, which increases the importance of security of Raspberry Pi and till now not enough security assessments are performed on Raspberry Pi devices.

3.2 Features of this Prototype

- This model detects weapons such as guns and knives in real time.
- On detection of the weapon the model will notify the owner through mail.
- Owner has facility to see camera footage.The Owner can also take the snapshots of the camera footage.

- Only authenticated users can see the footage.
- At night, when motion is detected by the model, the model switches on the LED bulb and will notify the owner.
- OS used in Raspberry Pi is Ubuntu Server 20.04.2 LTS which is open source and can be configured as required very easily.
- This model is cost efficient, fast and portable.

Chapter 4

Technologies Required

4.1 Sensors/Actuators Required

Raspberry Pi 3B+, PIR Motion Sensor, Raspberry Pi Camera Module, Relay Circuit for Switching Bulb, Bulb, Wifi Network and 16/32 GB Memory Card.

4.2 Open Source OS/Packages/Libraries/Application Installed

Ubuntu Server 20.04.2 LTS, Python3, OpenCV, Rpi.GPIO, flask, flask-sqlalchemy, flask-login, tensorFlow, smtplib, pip3, imutils, net-tools, raspi-config, rpi-update, binutils and imutils. Various other packages are also used which are by default installed in the system while installing the above packages/softwares.

Chapter 5

Working

5.1 Streaming on Web

Flask is used to hosting the python application for streaming on web [Ron20]. Flask is a lightweight WSGI web application framework. It is designed to make getting started quick and easy, with the ability to scale up to complex applications. We can buy IP and can host our application on that IP but for this prototype we are hosting this application on local wifi network i.e. any device connected with that Wifi network can access that application. We have to pass the IP where we want to host application at the time of running application. In while loop we are continuously yielding the processed image and that yielded value is then forwarded to html page for live streaming of the footage.

```
92  @main.route('/profile')
93  @login_required
94  def profile():
95      # return render_template('profile.html', name=current_user.name)
96      return render_template('js.html', name=current_user.name)
```

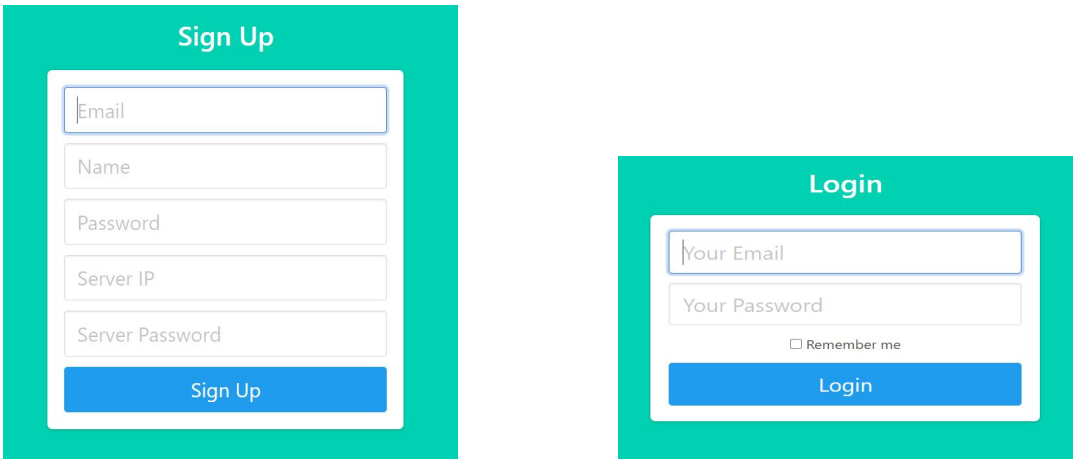
Fig. 5.1 Login Authentication

Flask-login (as shown in above figure) is used for checking whether user is authenticated user when user try to see live footage or else everybody will gain access to that live footage.

Flask-sqlalchemy database is used for storing the data. It contain two tables (as shown in figure), one table is used for storing user information when new user will signup and another table is used for storing server IP and sever Password on which application is running, which will be required by the new user while signup for security purposes.

```python
# models.py

from flask_login import UserMixin
from . import db

class User(UserMixin, db.Model):
    id = db.Column(db.Integer, primary_key=True) # primary keys are required by SQLAlchemy
    email = db.Column(db.String(100), unique=True)
    password = db.Column(db.String(100))
    name = db.Column(db.String(1000))

    # def __rep__(self):
    #     return f"User('{self.id}','{self.email}')"
class Post(db.Model):
    ip = db.Column(db.String(15), primary_key=True) # primary keys are required by SQLAlchemy
    password = db.Column(db.String(100))

    # def __rep__(self):
    #     return f"Post('{self.ip}','{self.password}')"
```

Fig. 5.2 Flask-Sqlalchemy Database Tables



(a) Signup Page (b) Login Page

Fig. 5.3 Web Pages for Authenticating User

5.2 Weapon Detection

5.2.1 Haar Cascade Classifier

For weapon detection we had previously used Haar Cascade Algorithm. Haar Cascade is a machine learning object detection algorithm used to identify objects in an image or video proposed by Paul Viola and Michael Jones in "Rapid Object Detection using a Boosted Cascade of Simple Features" in 2001 [VJ01]. In this machine learning approach cascade function is trained from a lot of positive and negative images. Later that cascade function is used to detect objects. This machine learning approach can be used for detecting any object. Though is commonly known for detecting body parts. For training we need a lots of positive and negative images of object that is to be detected. Then we are required to extract features from it.

- Haar Feature Collection: A Haar feature considers adjacent rectangular regions at a specific location in a detection window, sums up the pixel intensities in each region and calculates the difference between these sums.

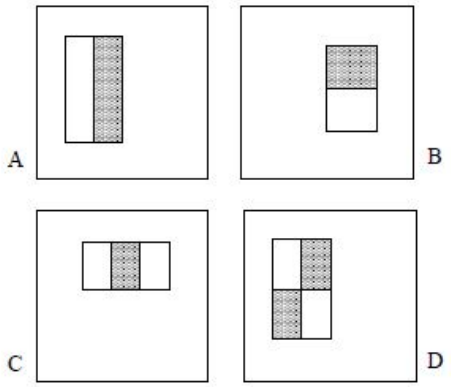


Fig. 5.4 Rectangle Feature

- Creating Integral Images: Rectangle features can be computed very rapidly using an intermediate representation for the image which is called as integral image.

- Adaboost Training: As most of the features are irrelevant, for selecting best feature

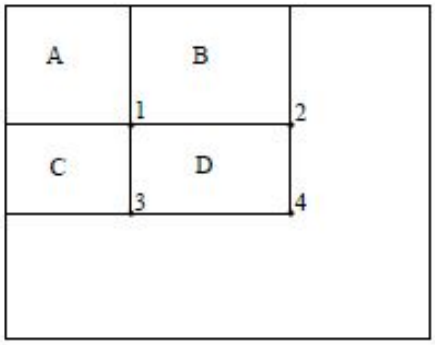


Fig. 5.5 The sum of the pixels within rectangle can be computed with four array references. The value of the integral image at location 1 is the sum of the pixels in rectangle A. The value at location 2 is A+B; at location 3 is A+C and at location 4 is A+B+C+D. The sum within D can be computed as (D+A) - (B+C).

among all features Adaboost concept is used . For example, as shown in fig:4.4 These features are selected by Adaboost. The first feature selected specifies the property that the region of the eyes is more darker than the region of the nose and cheeks. The second feature selected specifies the property that the eyes are more darker than the bridge of the nose. But these features will be irrelevant if we try to apply same windows on cheeks.

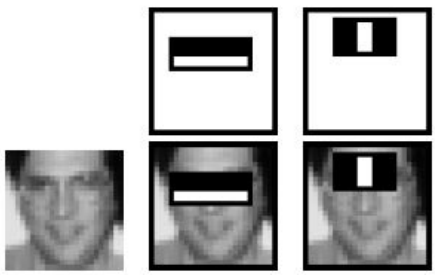


Fig. 5.6 Features Selected by Adaboost

Adaboost trains the classifiers that use them. This algorithm constructs a "strong" classifier as a linear combination of weighted simple "weak" classifiers. Haar feature is a weak classifier, a large number of Haar features are organized into cascade classifiers to form a strong classifier which then gives sufficient accuracy for detecting objects.

- Cascading Classifiers: The cascade classifier consists of a collection of stages, where each stage is an ensemble of weak learners. Each stage of the classifier labels the region defined by the current location of the sliding window as either positive or negative. Positive indicates that an object was found and negative indicates no objects were found. If the label is negative, the classification of this region is complete, and the detector slides the window to the next location. If the label is positive, the classifier passes the region to the next stage. The detector reports an object found at the current window location when the final stage classifies the region as positive. The stages are designed to reject negative samples as fast as possible. The assumption is that the vast majority of windows do not contain the object of interest. Conversely, true positives are rare and worth taking the time to verify.

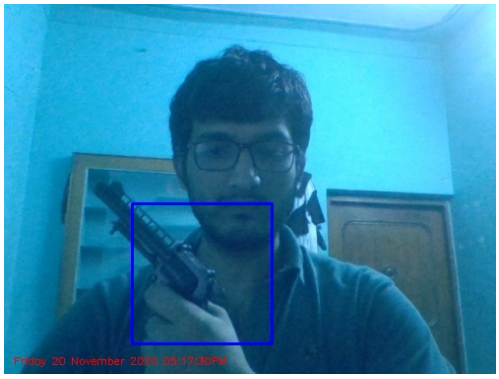


Fig. 5.7 Gun Detected by Cascade Classifier

Cascade Trainer GUI was used to train cascade classifier model. It directly outputs cascade.xml file which was later used for detecting weapons. Data sets of 3000 positive and 3000 negative images were taken from kaggle [kag] But accuracy of that cascade classifier was not not upto the mark. It was observed that several times it was sending wrong alerts that may be annoying for the users. This model is not that suitable for our prototype.

5.2.2 Tensorflow

TensorFlow is an open-source library which is used in Machine and Deep Learning fields like Computer Vision and Natural Language Processing and was developed by Google and released for public use in 2015. The advantage of using TensorFlow is that one does not need to do the back propagation manually while building a neural network. This back propagation step, often the trickiest step of building a neural network, has been generally implemented in TensorFlow and simply doing the forward propagation completes the job. Some advantages of using TensorFlow over any other framework include easy model building with the help of keras APIs, availability in many languages and a simple and fast architecture for powerful experimentation.

Transfer Learning

It is a method used in the field of machine learning where the output of a trained model is used as an input for some other model. Fixing the weights of the previously trained model and training on the weights of the new model produces accurate results and predictions. Some of the most accurate models like EfficientNet, ResNet, MobileNet, VGG are already in-built in TensorFlow for direct use.

Model Building

To localize a gun in an image, we have first trained a model to check whether a gun is present in the image or not. If a gun is present in the image, then the second model tries to draw a rectangle in the probable area where the weapon might be present.

To build the classifier on top of pre-built model, a set of 6000 images, half of which contained a weapon, was taken from Kaggle and weights were trained on this dataset. The model was run for 13(5+8) epochs and the cross categorical entropy loss was found to be less than 0.5.

In order to identify the object, a dataset of 3000 labelled images was taken from Kaggle,

all of which contained a weapon and the co-ordinates of the weapon in the image were also given. The model was trained on this dataset for 10(5+5) epochs and was trained until the model started producing satisfactory results.

For the purpose of these tasks, we have used ResNetV2 model from TensorFlow which has been trained on the ImageNet data, attached a few fully connected dense layers to it and trained it with a learning rate of 0.00001. ResNet is convolutional neural network which produces state-of-the-art results over ImageNet dataset which consists of more than 1 million images organised in 1000 categories.

Testing the Model

```
#loading the model.
model = keras.models.load_model('pre-model2.h5')
```

```
#loading the model.
modelt = keras.models.load_model('guns.h5')
```

```
#function to plot rectangle.
def plot_pred(img,z):
    fig, ax = plt.subplots(1)
    ax.imshow(img)
    rect =Rectangle((z[0][0],z[0][1]),width = z[0][3],height = z[0][2], linewidth=1,edgecolor='g',facecolor='none')
    ax.add_patch(rect)
    plt.show()
```

```
#loading and preprocessing an image for testing.
img1 = cv2.imread('7.jpg')
img1 = cv2.cvtColor(img1, cv2.COLOR_BGR2RGB)
img1 = cv2.resize(img1, (416,416))
```

```
y_hat = model.predict(img1.reshape(-1,416,416,3))[0]
print(y_hat)
if y_hat[1]>0.40:
    img1 = cv2.cvtColor(img1, cv2.COLOR_BGR2RGB)
    z = modelt.predict(img1.reshape(-1,416,416,3))
    plot_pred(img1,z)
```

```
[0.36104006 0.63896   ]
```

Fig. 5.8 Code for Loading and Testing Model

Above is the code associated with identifying a gun in an image. The classifier model outputs 2 numbers, the second of which is the probability of a gun being present in the image. If the gun is present, the second model draws a rectangle around the gun, else the algorithm moves onto the next image.

Fig. 5.9 Weapon Detected by Tensorflow Model

5.3 Alerting

```
56
57    EMAIL_ADDRESS = os.environ.get('EMAIL_USER')
58    EMAIL_PASSWORD = os.environ.get('EMAIL_PASS')
59
```

Fig. 5.10 Extracting Gmail Credentials from System Variables

smtplib is used to send mail when weapon gets detected.The smtplib module defines an SMTP client session object that can be used to send mail to any Internet machine with an SMTP or ESMTP listener daemon. Gmail account is used for sending mail. This model requires its own Gmail account to send mails. For security purposed we had not directly hard-coded credentials for that account in program instead we had set the environment variables for credentials.

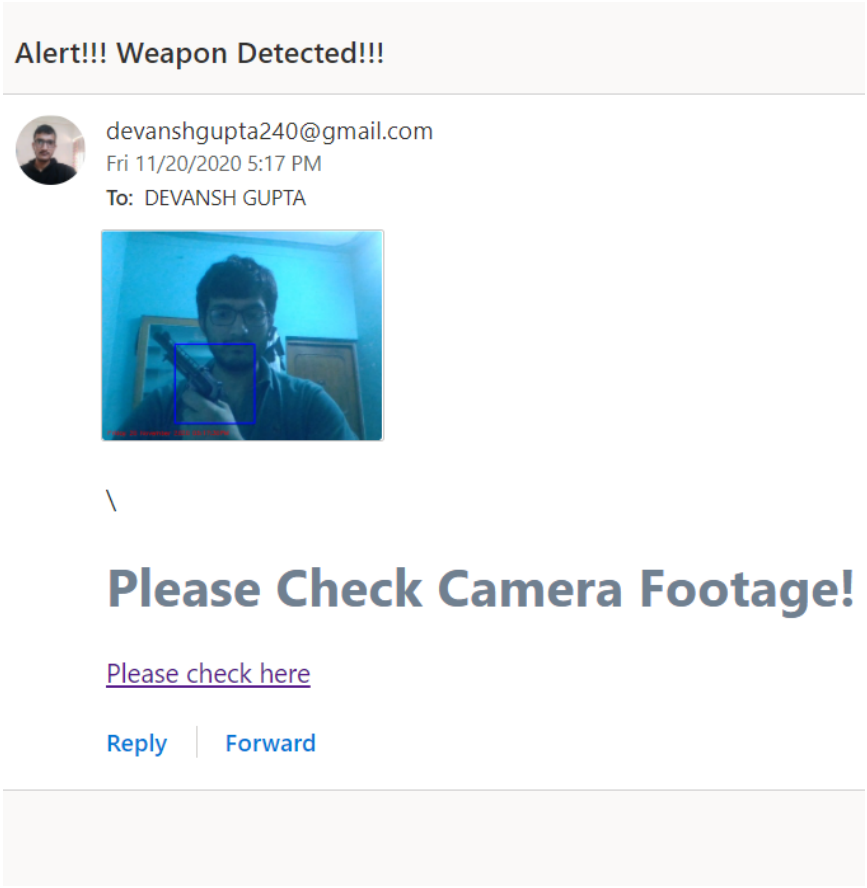
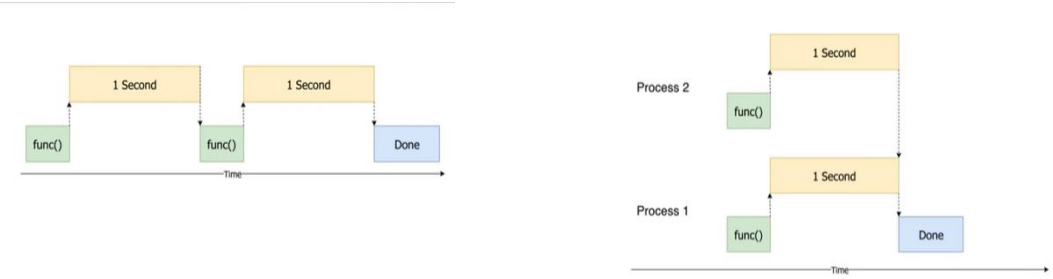


Fig. 5.11 Mail Received on Gun Detection

Mail will contain the picture of the instance when the system detected weapons and also mail will contain direct link to live streaming of footage.

We observed that the live footage was lagging when we send mail because it takes time to send mail and time taken for sending mail may be large depending on multiple factors such as processor speed, internet speed etc. As Raspberry Pi 3b+ model has quad core 64-bit processor clocked at 1.4GHz, so we used multiprocessing (as shown in figure) to prevent delay during mailing. Multiprocessing will run two process at a time which will prevent delay/lagging.



(a) Without Multiprocessing (b) With Multiprocessing

Fig. 5.12 Multiprocessing

```
295
296             p1 = multiprocessing.Process(target=sendMail)
297             p1.start()
298
```

Fig. 5.13 Multiprocessing Code

5.4 PIR Motion Sensor Working

During the dark/night time, it will be not possible for camera to see objects clearly and also we don't want to on lights as well as camera whole night as this will result in wastage of resources. So, PIR Motion sensor is deployed on the system, when it senses motion in region under surveillance it switches the light and Raspberry Pi camera module on and also sends a alert mail to owner. In this case also we had used multiprocessing for sending mail to prevent lag/delay in the live streaming of camera footage. The mail contains the picture of the instance when motion is sensed by the PIR Motion sensor and also contain quick link to the live streaming of the footage.

PIR uses the fact that human and animals emits Infrared Waves. The PIR sensor has two slots in it, each slot is sensitive to infrared rays. The lens is used to converge the infrared rays to those two slots. When there is no motion, both slots detect the same amount of infrared, the ambient amount radiated from the room or walls or outdoors. When human or animal who emits large amount of infrared passes by, it first intercepts one half of the PIR sensor, which causes a positive differential change between the two halves. When the warm body leaves the sensing area, the reverse happens, whereby the sensor generates a

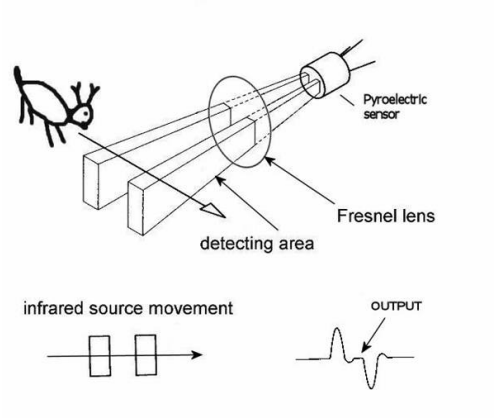


Fig. 5.14 PIR Sensor Working

negative differential change. These change in pulses is the indicator of motion.[VJ01] It outputs high voltage when motion is detected and this output is further used to light led bulb. Since raspberry pi can't provide enough power to light 5W led bulb we are using relay circuit as a control switch.

Chapter 6

Configuration System and Running Application

6.1 Configuration

- **Installing Ubuntu Server**
  - Download Ubuntu Server 20.04.2 LTS .img file from url: `https://ubuntu.com/download/raspberry-pi`.
  - Flash the downloaded .img file into the memory card using Rufus.
  - Put the card in the Raspberry Pi 3B+ Model and boot/power on the system.
  - Login using default credentials i.e. username:ubuntu and password:ubuntu. After that it will ask you to change password, change the password of the system.
- **Wifi Configuration**
  - Open the internet configuration file using the command *sudo nano /etc/netplan/50-cloud-init.yaml* and add your wifi network and its password (as shown in figure) to the file and save it.
  - Then run the following command to apply change in the system:

* *sudo netplan generate*

* *sudo netplan apply*

```
# This file is generated from information provided by the datasource.  Changes
# to it will not persist across an instance reboot.  To disable cloud-init's
# network configuration capabilities, write a file
# /etc/cloud/cloud.cfg.d/99-disable-network-config.cfg with the following:
# network: {config: disabled}
network:
    ethernets:
        eth0:
            dhcp4: true
            optional: true
    version: 2
    wifis:
        wlan0:
            access-points:
               "RAM":
                    password: "qwertyui"
            dhcp4: true
            optional: true
```

Fig. 6.1 Wifi Settings

- **Install ifconfig package:** ifconfig is used for checking IP of the system which we will required at the time of ssh for controlling our system remotely. Run command *sudo apt install net-tools* for installing ifconfig package.

- **Running Raspberry Pi Camera Module:** Since Raspberry Pi modules are still unknown to OS therefore we need to install some Raspberry Pi packages to the system and to run camera module.

  – We need raspi-config package for enabling camera module. Run the following commands to install raspi-config package:

    * *sudo apt-get update*

    * *sudo apt-get upgrade*

    * *wget https://archive.raspberrypi.org/debian/pool/main/r/raspi-config/raspi-config_20160527_all.deb -P /tmp*

    * *apt-get install libnewt0.52 whiptail parted triggerhappy lua5.1 alsa-utils -y*

* *apt-get install -fy*
* *dpkg -i /tmp/raspi-config_20160527_all.deb*
* *sudo mount /dev/mmcblk0p1 /boot*

– Run *sudo raspi-config* command and then enable camera module using keyboard controls.

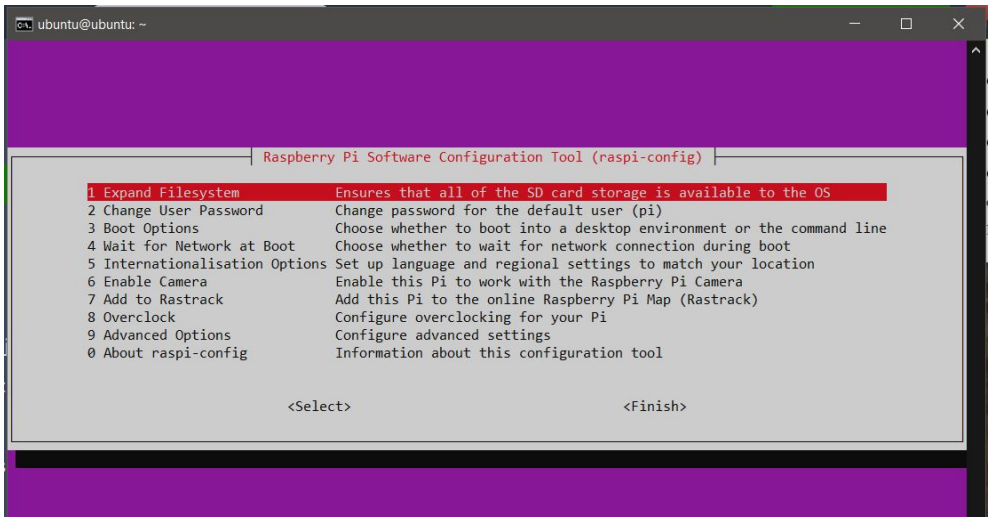


Fig. 6.2 Enable Raspberry Pi Camera

– We need raspi-update package for running camera module. Run the following commands to install rpi-update package:

* *sudo curl -L --output /usr/bin/rpi-update https://raw.githubusercontent.com/Hexxeh/r update/master/rpi-update sudo chmod +x /usr/bin/rpi-update*
* *sudo apt-get install binutils*

– Now run *sudo rpi-update* command and reboot the system.

– Now run *raspistill -o cam.jpg* command to ensure camera module is working properly.

- **Install pip3 package:** This package will be required for install further packages in future. To install pip3 package run *sudo apt install python3-pip* command.

- **Install opencv package:** This package is required for processing image while detecting weapon. To install opencv package run *sudo apt install python3-opencv* command.
- **Install imutils package:** This package is also required for processing image while detecting weapon. To install imutils package run *sudo pip3 install imutils* command.
- **Install flask packages:** 3 packages are required to run our application. Those are:
  - flask: It is the package required for hosting application on web.
  - flask-sqlalchemy: This is the database package required for string data required in application.
  - flask-login: This package is required for authenticating user when he/she tries to access live footage.

  All these 3 packages can be installed by running *pip3 install flask flask-sqlalchemy flask-login* command.
- **Installing Tensorflow Package:** Tensorflow packages are required during gun detection. Install tensorflow by running *pip3 install –upgrade tensorflow* command.
- **Installing RPi.GPIO:** This package allows us to GPIO pins available on the Raspberry Pi. Run the following commands for installing RPi.GPIO package:
  - *sudo apt-get update*
  - *sudo apt-get upgrade*
  - *sudo pip3 install RPi.GPIO*
- **GPIO pins setup:** Pin number used in steps are referencing to the pin numbers shown inside the circle in the given diagram. Follow the below steps to connect the sensors and actuators to the GPIO pins properly:
  - PIR Motion Sensor:

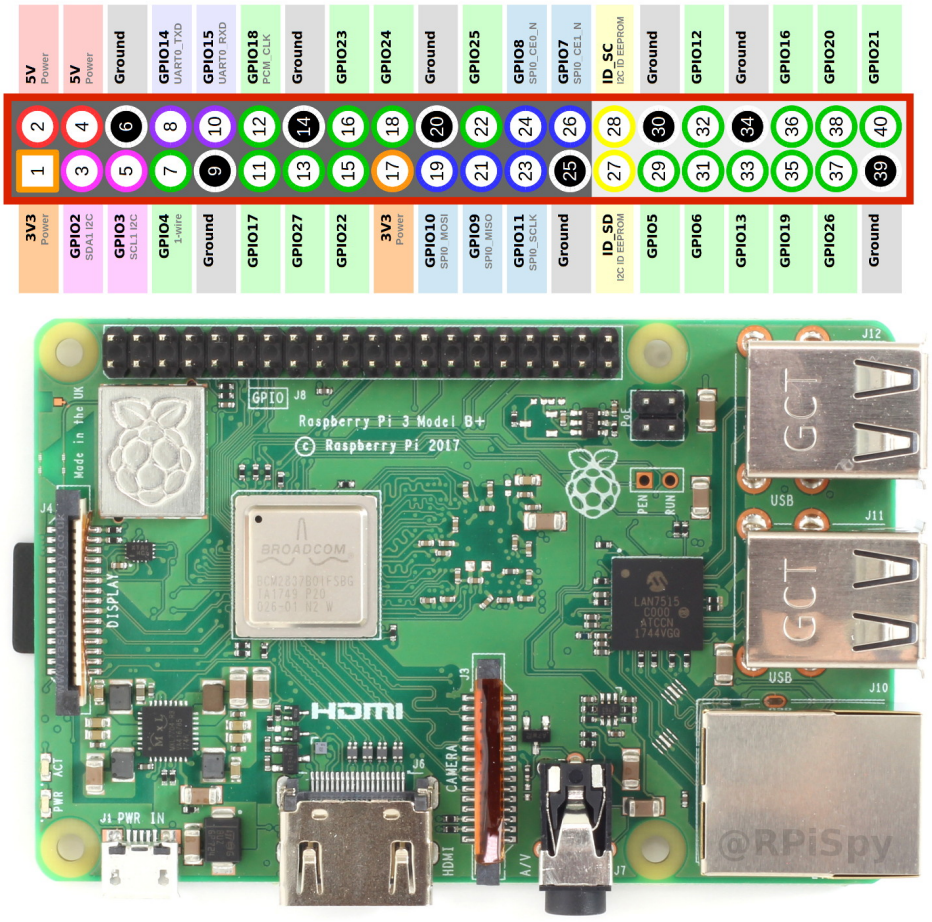


Fig. 6.3 Pin Diagram of Raspberrpy Pi 3B+ Model

* Connect Vcc to pin number 1.
* Connect Gnd to pin number 6.
* Connect output pin to pin number pin number 7.

– Relay Circuit:

* Connect Vcc to pin number 17.
* Connect Gnd to pin number 9.
* Connect input pin to pin number pin number 11.
* Cut any one of the two wire powering bulb and connect the one of those ends to common pin and another to normally closed pins of relay circuit.

Now relay circuit will act as the switch for bulb.

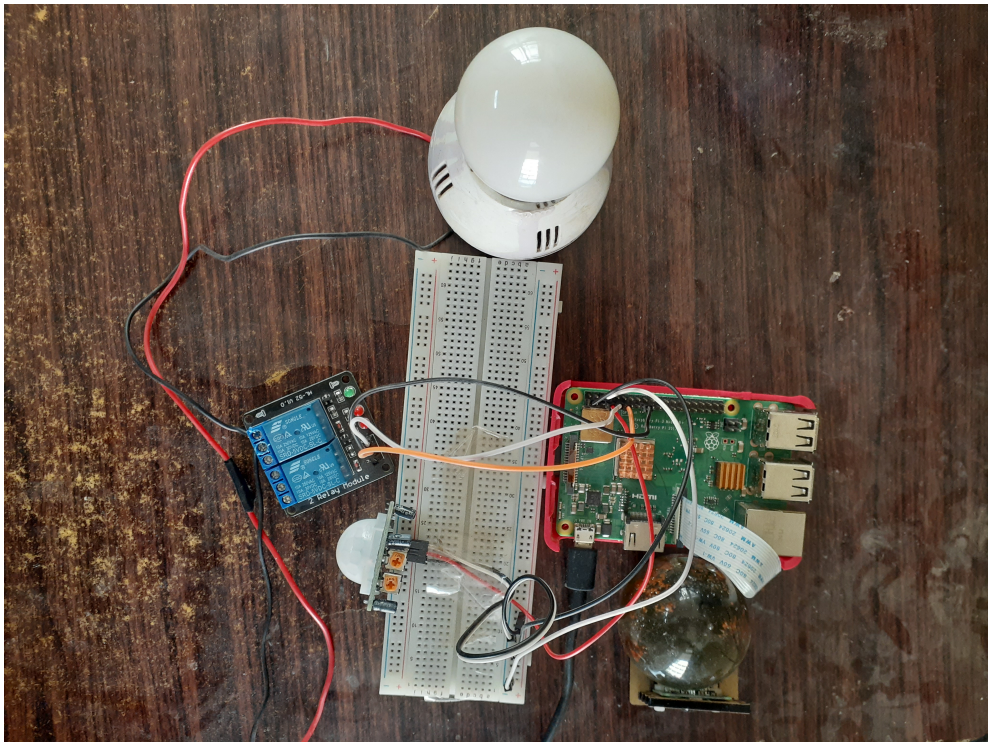


Fig. 6.4 Configured Prototpye

6.2 Running Application

- **Set System Variables:** Account Details which is used by system for sending mails is not hard-coded rather they are stored as system variables. Run the command shown in figure in the terminal of your system to set variables.

```
ubuntu@ubuntu:~/btp/till14thpm$ export USER_NAME="alhpa123@gmail.com"
ubuntu@ubuntu:~/btp/till14thpm$ export USER_PASS="alpha@123"
```

Fig. 6.5 Setting Variables

- **Create Database Tables:** Go inside the main application directory which contain project folder. Run python and use the commands given in figure to create the database. After creating the database new file nameed db.sqlite will be created inside project folder.

```
>>> from project import db, create_app
>>> db.create_all(app = create_app())
```

Fig. 6.6 Creating Database

- **Setting Server Password:** Add the sever IP and server Password as a entry to the database table Post, which will be required by users to authenticate themselves when they sign up. In the main application directory, open python interface and run the commands shown in figure.

```
>>> app = create_app()
C:\Users\devansh\AppData\Local\Packages\PythonSoftwareFoundation
2: FSADeprecationWarning: SQLALCHEMY_TRACK_MODIFICATIONS adds si
s warning.
  warnings.warn(FSADeprecationWarning(
>>> app.app_context().push()
>>> from project.models import User, Post
>>> post1 = Post(ip = '192.168.137.206', password = 'qwertyui')
```

(a) Creating Entry containing IP and Password

```
>>> db.session.add(post1)
>>> db.session.commit()
>>> Post.query.all()
[<Post 192.168.137.206>]
```

(b) Adding Entry

Fig. 6.7 Setting System IP and Password in Database

- **Starting Application:** Set the FLASK_APP variable as project and run the *flask run –host=192.168.137.65* command on terminal to run the application (where 192.168.137.65 is the system IP).

```
ubuntu@ubuntu:~/btp/till14thpm$ export FLASK_APP=project
ubuntu@ubuntu:~/btp/till14thpm$ flask run --host=192.168.137.65
 * Serving Flask app "project"
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployment.
   Use a production WSGI server instead.
 * Debug mode: off
/home/ubuntu/.local/lib/python3.8/site-packages/flask_sqlalchemy/__init__.py:872: FSA
K_MODIFICATIONS adds significant overhead and will be disabled by default in the futu
press this warning.
  warnings.warn(FSADeprecationWarning(
 * Running on http://192.168.137.65:5000/ (Press CTRL+C to quit)
```

Fig. 6.8 Starting Application

Chapter 7

Security Assessment of the Model

7.1 Hardware Analysis

Vulnerabilities can also be in hardware if hardware is not properly designed. Hardware design should be robust enough to secure system in the case when non-authorized people have access to system. Also many side channel attacks can be prevented by using proper hardware design. Hardware analysis includes the processor, internal design and the external connectors.

7.1.1 USBs Power and Backfeeding

The output current on the USBs of Raspberry Pi can go from 500mA to 1.2A so it is not always enough to power device. Since our Raspberry Pi would be in open environment or in public place which make or prototype vulnerable to physical attack i.e. attacker can plug in the device consuming more power through USB which may eventually lead to failure of power system in Raspberry Pi.In that case powered USB hubs are used to back feed Raspberry Pi or one simple way to prevent any type of physical attack is to tightly pack raspberry pi in the metal or carbon fiber box so it would be not possible for attacker to insert any device in Raspberry Pi through USB or damage Raspberry Pi physically.

7.1.2 Overclocking

When there are so many processes to run on CPU, system automatically overclock itself that is it increases the multiplier on the CPU, allowing it to run faster. The higher the CPU is clocked, the higher the voltage is required as a result is produces more heat. Above 85^0C, system gets into danger zone, it would not kill CPU immediately but lead to a shorter lifespan for the chip. However, if the temperature exceeds 105^0C, CPU could be damaged immediately. Also high voltage directly could also fry our chip and kill it. Overclocking is always possible in Raspberry Pi due to its smaller processor. It opens a way for attacker to attempt attack on Raspberry Pi and damage it physically. If attacker could gain the access to the terminal of raspberry pi and if he is able to execute even a single command on it, he/she can run that command or programs frequently to put load on its processor which will lead to overclocking and will eventually lead to heating and over-voltage causing physical damage to Raspberry Pi. To prevent over heating or over voltage due to overclocking, force_turbo has been set to 1 so in case if Raspberry Pi temperature reaches above 85^0C, overclocking will automatically will be disabled. force_turbo option is available in raspi-config application.

7.1.3 GPIO Logic Levels and Serial Access

The GPIO pins of Raspberry Pi use a voltage of 3.3V and do not have any protection against over voltage which means that any voltage higher that 3.3 can destroy Raspberry Pi. This limitation also make Raspberry Pi venerable to physical attack which can damage Raspberry Pi permanently. As our model will in public place, intruder/attacker can break the connection of GPIO pins from PIR Sensor or Relay Circuit and can connect the wires from GPIO pins to high voltage device which can damage Raspberry Pi. So, instead of direct access, access of GPIO pins can be made from serial port. This part is currently no included in our system. Our system is currently vulnerable to over-voltage issue.

7.1.4 Real Time Clock Absence

Raspberry Pi does not have its own real time clock which means date and time are not stored internally which required for proper functioning of cryptographic actions as well as for some other important actions like certificate validation. Due to absence of real time clock Raspberry Pi will depend on internet for the time and date and will make Raspberry Pi vulnerable to malfunction during cryptography putting security of data at risk, also at the time of certificate validation our system could validate expired certificate leading unauthorized user/attacker to interact with the system putting system at risk. Our system does not have real time clock which means our system is vulnerable to malfunctioning of these actions.

7.2 Software Analysis

Software analysis is important to prevent remote attacks on the system. Software analysis involve Operating System analysis, Networks analysis, Cryptography analysis and analysis of applications used by the system.

7.2.1 Operating System Analysis

The operating system is the life of a computer system. It's the main software component that which allows the system to become controllable and operational. It manages all the programs and applications on the computer. Being the control center of the computer, its role in the overall security of the system is paramount. So securing operating system should be first priority of any system designer. Any attacker/intruder who wants to attack system should be able to exploit bugs/vulnerabilities in the Operating System and make his attack successful. Ubuntu Server 20.04.2 LTS has been installed on our system. This the latest and updates version of Ubuntu available for raspberry pi, which means that OS is not vulnerable for general bugs/attacks. Updation of OS can be done at regular intervals to keep it free

from common attacks. Further some changes can be done in default configuration of kernel to make it more robust.

7.2.2 Source Code

Hard-coding information required to run application is considered as bad practice as it can lead to breach in privacy/security of system. If the attacker/intruder gets access to view the code file which is much easier than extracting information from system variables, he/she can extract those information from the code file and can use that information to interrupt/distract the functioning of the application. In our case gmail id and password are required to run the code for sending alert mails to the the owner, if attacker could extract those gmail credentials, he/she can disable that gmail account so the system will not be able to alert mails to the owner. So, the information required for running our application is not hard-coded in code instead they are taken dynamically from the system so in case if some outsider gets access to source code, he/she can't breach information or disrupt functioning of application.

7.2.3 Application

Hosting the live footage on the web with no authentication process can lead to breach in the privacy, any user can who has access to internet can see the live footage of other's (system owners) place and can use that information for his own benefits. So, system password has been configured which will be required by the user while accessing the live footage. So,only authenticated users (who have system password) will be able to access live footage. The system password would be set up by the system owner while running the system for the first time.

7.2.4 Cryptographic Analysis

Wireless Sensor Networks poses unique security challenges due to the accessibility of the location of deploy and due to open communication channel (wifi network). Attacker can easily capture the data packets communicated by our system and can extract the information the contain, this can put privacy as well as security of the system in danger. So it is always considered better practice to have good encryption and key distribution schemes so that the data packets can be communicated over the open channel safely, at the receiver end, receiver can decrypt that data and use it accordingly. But due to small processor of the Raspberry Pi we can't use complex encryption algorithm which may take so much time for encryption. ECC (Elliptic Curve Cryptography) can achieve same level of security as other traditional algorithms like RSA with much small key size and less time. So, a lightweight cyptographic algorithm such as ECC with robust key distribution scheme can be used to enhance the security of our data.

Chapter 8

Future Work

- CentOs can be installed on raspberry pi instead of Raspbian Os to make system more secure. CentOS has multiple security features built-in.CentOS helps to protect cyber-attacks by utilizing Security-Enhanced Linux (SELinux). SELinux is an access control mechanism that can enforce rules on processes and files, based on policies that are defined by owner. One of the most beneficial it reduces vulnerabilities on privilege escalation attacks and if a process is compromised, the attacker would only have access to the normal functions of the process. I had tried to build this model upon centos, but I was not able to find some configuration like interface to enable camera module in centos.

- Additionally, Message can be sent to owner on messenger as people are more active on messenger rather than on mail these days. I had tried it but was not successful in implementing.

- We can make this system more interactive by taking inputs such as adding owner, recording footage, control working etc from html page but for now, owner details is hard-coded in this model.

- Live streamed data can be encrypted by using lightweight cryptography algorithms

such as ECC during communication over networks.

- As Ubuntu is open source we can modify its source code or configure it according to our system requirements to make it more secure.
- Serial post can be used to access GPIO pins to prevent over voltage.
- Real time clock can be integrated inside the system.

Chapter 9

Conclusion

- All the sensors and actuators except Raspberry pi cost less than Rs. 600. There are lot of functionality of raspberry pi which we haven't use so if we fabricate board only for surveillance security system, it will cost much lower than Rs.3000. This makes our system cost effective.
- This model is successful in detecting threats and alerting owner and provides variety of functionalities of smart surveillance security system.
- Security Assessment have been performed on this prototype.
- Ubuntu OS is used, which is open to any modification we need in order to make our system efficient and secure.
- Several steps had been taken to make the system secure.
- This model is environmental friendly as it uses far less electricity as compare to any traditional CCTV system.
- This system is user friendly as it is portable and easy to deploy.

Chapter 10

Major Contributions in Phase II

- **Web Portal for Authentication:** In the phase 1, we just streamed footage on the web which required one html page (only front-end), no back-end work was involved in that. But for checking authenticity of the user we have to use back-end also which required the use of flask database package i.e. flask-sqlalchemy for storing data of the user and flask-login package for checking authenticity (user has account) of the user when he tries to see the live footage. This time we had made whole portal which has 4 html pages linked with each other. Also we included security aspect in it i.e. user had to know server's password if he/she wants to see live footage captured through that server.

- **Robust Weapon Detection Model:** In the phase 1, we had used cascade classifier model for detecting weapon which was not accurate and due to which user gets so many false alerts which would had annoying for user and also cascade classifier model was old model based on machine learning but now various new technologies have evolved. This time, for weapon detection we had used tensorflow which is the latest technology and it is developed by google based on deep learning and it is still under development. For weapon detection we had build two models using transfer learning concept on pre-trained model Resnet50, one for identifying whether weapon is present

in the image and second for encapsulating weapon in a rectangle if it is present in the image. After combining these models, results were much satisfactory than the cascade classifier model. To test the accuracy of these models we had collected randomly shuffled 250 images some containing weapons and some do not. We found that accuracy of tensorflow model was approx 82.8% while the accuracy of cascade classifier was only 61.2%.

```
                count1 = count1 + 1
#             print("guns NOT detected")
        gun_exist = False

    print(count1, count2)
    return count1,count2
        # show the frame and record if the user presses a key
```

```
In [87]: count1,count2 = findAccuracy(images)
armas (322).jpg 1 True
pexels2373.jpg 0 False
armas (2854).jpg 1 False
armas (1888).jpg 1 True
pexels784.jpg 0 True
pexels229.jpg 0 False
armas (2113).jpg 1 True
armas (860).jpg 1 True
armas (2951).jpg 1 True
armas (1203).jpg 1 True
pexels1314.jpg 0 False
pexels2159.jpg 0 True
pexels1089.jpg 0 True
armas (803).jpg 1 True
pexels1845.jpg 0 True
armas (1163).jpg 1 True
pexels211.jpg 0 True
pexels94.jpg 0 True
pexels2719.jpg 0 False
153 97

In [88]: count1/(count1+count2)
Out[88]: 0.612
```

(a) Accuracy of Cascade Classifier

```
In [83]: def tensorModel(images):
    right,wrong=0,0
    counter = 0
    for file in images[:250]:
        img = cv2.imread(os.path.join(r'C:\Users\Pranav\OneDrive\Desktop\New folder\gun-data\combine',file[0]))
        img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
        img = cv2.resize(img, (416,416) )
        img = np.array(img)
        y_hat = model.predict(img.reshape(-1,416,416,3))[0]
        if y_hat[1]>0.40:
            if file[1]==1:
                right+=1
            else:
                wrong+=1
        else:
            if file[1]==1:
                wrong+=1
            else:
                right+=1
        counter+=1

    return right,wrong

In [84]: right,wrong = tensorModel(images)
print(right, wrong)
207 43

In [85]: print(right/(wrong+right))
0.828
```

(b) Accuracy of Tensorflow Model

Fig. 10.1 Accuracy Test on Weapon Detection Models

- **Ubuntu Sever as OS:** In the phase 1, first we had tried to use CentOs as OS which was more secure and safer but due to lack of information regarding its configuration procedure to sync it with raspberry pi on the internet we had to use Raspbian OS

which is pre-configured and ready to use with Raspberry Pi but it is slower and also not much explored to be configured according to our use. This this time we had used Ubuntu Server 20.04.2 LTS as OS which is more secure, faster and much explored OS. We can configure its kernel or its setting according to our needs to make it more faster and secure. In this we had to find and do various manual configuration to sync it with Raspberry Pi, Raspberry Pi Camera Module and Raspberry Pi GPIO pins.

- **Literature Survey of Raspberry Pi Security:** In this phase, I had read several papers related to security of Raspberry Pi which includes: -
  - Digital Forensic Procedure to find digital evidences after any attack attempt.
  - Changes in traditional cryptographic methods to prevent some side channel attacks.
  - Security assessment of Raspberry Pi based NIFT standard.
  - Hardware and software feature assessment of Raspberry Pi.
  - Lightweight crpytographic algorithms and key distribution schemes like ECC that can be successfully implemented on Raspberry Pi.

References

[ASK19] Kengo IOKIBE Akihiro SANADA, Yasuyuki NOGAMI and Md. Al-Amin Khandaker. Security analysis of raspberry pi against side-channel attack with rsa cryptography. *Consumer Electronics Security*, 2019.

[BE] Onafeso Babatunde and Liu E. Cyber security investigation for raspberry pi devices.

[BV20] M. Ramakrishnan B. Vanathy. Dynamic key distribution management using key escrow based ecc algorithm in manets. *International Journal of Advanced Research in Engineering and Technology (IJARET)*, 2020.

[CK17] Akash Nalawade Sanket Parode Chinmaya Kaundanya, Omkar Pathak. Smart surveillance system using raspberry pi and face recognition. *International Journal of Advanced Research in Computer and Communication Engineering*, 6:622–623, 2017.

[eli19] eliu01011. Weapon-detection-using-neural-networks-in-real-time-surveillance-video, 2019. `https://github.com/eliu01011/Weapon-Detection-Using-Neural-Networks-in-Real-Time-Surveillance-Video`.

[HQN15] Bui Dinh Eui-Nam Huh Huu-Quoc Nguyen, Ton Thi Kim Loan. Low cost real-time system monitoring using raspberry pi. *2015 Seventh International Conference on Ubiquitous and Future Networks*, 2015.

[JLTR16] G. Niezen J. Louw and A.M. Abu-Mahfouz T.D. Ramotsoela. A key distribution scheme using elliptic curve cryptography in wireless sensor networks. 2016.

[JSRC19] Gabriel Diaz Jorge Sainz-Raso, Sergio Martin and Manual Castro. Security vulnerabilities in raspberry pi- analysis of the system weaknesses. *Consumer Electronics Security*, 2019.

[kag] kaggle. weapons datasets. `https://www.kaggle.com/search?q=weapons+in%3Adatasets`.

[MSDG15] Vamsikrishna Patchava M. Surya Deekshith Gupta, Virginia Menezes. Surveillance and monitoring system using raspberry pi and simplecv. *2015 International Conference on Green Computing and Internet of Things (ICGCIoT)*, 2015.

[nam17] namdothanh. Homesecuritysystem, 2017. `https://github.com/namdothanh/HomeSecuritySystem`.

[Ron20] Armin Ronacher. Flask, 2020. `https://flask.palletsprojects.com/en/1.1.x`.

[RSS20] Poonam Jindal Rakesh Sharma and Brahmjit Singh. Study and analysis of key generation techniques in internet of things. *Journal of Discrete Mathematical Sciences and Cryptography*, 2020.

[ser19] seraj94ai. Flask-streaming-pedestrians-detection-using-python-opencv-, 2019. `https://github.com/seraj94ai/Flask-streaming-Pedestrians-detection-using-python-opencv-`.

[SS17] Sudish N George S Sruthy. Wifi enabled home security surveillance system using raspberry pi and iot module. *2017 IEEE International Conference on Signal Processing, Informatics, Communication and Energy Systems (SPICES)*, 2017.

[SUK11] Luciano Lavagno Maurizio A. Spirito Sarmad Ullah Khan, Claudio Pastrone. An authentication and key establishment scheme for the ip-based wireless sensor networks. *The 7th International Symposium on Intelligent Systems Techniques for Ad hoc and Wireless Sensor Networks (IST-AWSN)*, 2011.

[VJ01] Paul Viola and Michael Jones. Rapid object detection using a boosted cascade of simple features. *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, 2001.

[Wil18] Michael Grant Williams. A risk assessment on raspberry pi using nist standards. 2018.