INTRUSION DETECTION SYSTEM USING RASPBERRY PI

TERM PROJECT

Submitted by

CB.EN.U4CCE21019	**G. RAMA PHANI VARMA**
CB.EN.U4CCE21021	**G.JATIN VARMA**
CB.EN.U4CCE21022	**G.ANIRUDH**
CB.EN.U4CCE21035	**B.MOUNI PRAKASH REDDY**

B.TECH COMPUTER AND COMMUNICATION ENGINEERING
2021 BATCH

ABSTRACT:

- An intrusion detection system (IDS) is a security system that monitors a network or system for suspicious activity and provides alerts when it is detected.
- An IDS can be used to detect a wide range of attacks, including unauthorized access, network intrusions, and malware infections
- An intrusion detection system using a Raspberry Pi camera can be a simple and effective way to monitor a home or office for security breaches.

CODE:

```
import RPi.GPIO as GPIO
import smtplib
from email import encoders
from email.mime.text import MIMEText
from email.mime.base import MIMEBase
from email.mime.multipart import
MIMEMultipart
from picamera import PiCamera
import time
import io
import logging
import socketserver
from threading import Condition
from http import server



PAGE="""\
<html>
<head>
<title>Web Streaming</title>
</head>
<body>
<center><h1>Web Streaming</h1></center>
<center><img src="stream.mjpg"
width="640" height="480"></center>
</body>
</html>
"""

class StreamingOutput(object):
    def __init__(self):
        self.frame = None
        self.buffer = io.BytesIO()
        self.condition = Condition()
    def write(self, buf):
        if buf.startswith(b'\xff\xd8'):
            # New frame, copy the existing buffer's
content and notify all
            # clients it's available
            self.buffer.truncate()
            with self.condition:
                self.frame = self.buffer.getvalue()
                self.condition.notify_all()
            self.buffer.seek(0)
        return self.buffer.write(buf)

class
StreamingHandler(server.BaseHTTPRequestH
andler):
    def do_GET(self):
        if self.path == '/':
            self.send_response(301)
            self.send_header('Location',
'/index.html')
            self.end_headers()
        elif self.path == '/index.html':
            content = PAGE.encode('utf-8')
            self.send_response(200)
            self.send_header('Content-Type',
'text/html')
            self.send_header('Content-Length',
len(content))
            self.end_headers()
            self.wfile.write(content)
        elif self.path == '/stream.mjpg':
            self.send_response(200)
            self.send_header('Age', 0)
            self.send_header('Cache-Control', 'no-
cache, private')
            self.send_header('Pragma', 'no-cache')
            self.send_header('Content-Type',
'multipart/x-mixed-replace;
boundary=FRAME')
            self.end_headers()
            try:
                while True:
                    with output.condition:
                        output.condition.wait()
                        frame = output.frame
                    self.wfile.write(b'--FRAME\r\n')
                    self.send_header('Content-Type',
'image/jpeg')
                    self.send_header('Content-
Length', len(frame))
                    self.end_headers()
                    self.wfile.write(frame)
                    self.wfile.write(b'\r\n')
            except Exception as e:
                logging.warning(
                'Removed streaming client %s:
%s',
                self.client_address, str(e))
        else:
            self.send_error(404)
            self.end_headers()

class
StreamingServer(socketserver.ThreadingMixI
n, server.HTTPServer):
    allow_reuse_address = True
    daemon_threads = True




GPIO.setwarnings(False)
GPIO.setmode(GPIO.BOARD)
buzzer=37
pir = 12
```

```
GPIO.setup(pir,GPIO.IN)
GPIO.setup(buzzer,GPIO.OUT)
print("Sensor is ready!")
while True:
    dectmotion = GPIO.input(pir)

    if dectmotion == 1:
        print("Motion Detected")
        GPIO.output(buzzer,GPIO.HIGH)
        picam = PiCamera()
        picam.rotation = 180
        picam.start_preview()
        picam.resolution = (960, 480)
        time.sleep(1)
        picam.capture("test.png")
        picam.stop_preview()
        picam.close()
        server =
smtplib.SMTP('smtp.gmail.com',587)

        server.ehlo()
        server.starttls()
        server.ehlo()


server.login('revanth.damisetty@gmail.com','l
oxzniiucosnspby')

        text = MIMEMultipart()
        text['From']='Revanth'

text['To']='juturuloknath2003@gmail.com'
        text['Subject']='SMTP in action'

        text.attach(MIMEText("Motion
Detected"))
```

```
        filename = 'test.png'
        attachment = open(filename, 'rb') #Here
rb denotes read bytes becz we are dealing with
image not text
        p = MIMEBase('application', 'octet-
stream') #for processing image data
        p.set_payload(attachment.read())
        encoders.encode_base64(p)
        p.add_header('Content-Disposition',
f'attachment; filename = {filename}')
        text.attach(p)


server.sendmail("revanth.damisetty@gmail.co
m","juturuloknath2003@gmail.com",text.as_s
tring())
        print("Mail Sent")
```

```
    with PiCamera(resolution='640x480',
framerate=24) as camera:

        #Uncomment the next line to change
your Pi's Camera rotation (in degrees)
        camera.rotation = 180
        output = StreamingOutput()
        camera.start_recording(output,
format='mjpeg')
        GPIO.output(buzzer,GPIO.LOW)
        try:
            address = ('192.168.173.69', 5000)
            server = StreamingServer(address,
StreamingHandler)
            server.serve_forever()
            time.sleep(5)



        finally:
            camera.stop_recording()

      else:
         GPIO.output(buzzer,GPIO.LOW)


GPIO.cleanup()
```

OUTPUT

WORKING-PRINCIPLE:

- The camera can be used to monitor a room or area for movement.
- When movement is detected, the Raspberry Pi can take a picture of the intruder and send an alert to the user.
- **Hardware Setup**: The system consists of a Raspberry Pi, a camera, and a motion sensor. The camera is connected to the Raspberry Pi, and the motion sensor is connected to both the Raspberry Pi and the camera.
- **Motion Detection**: The motion sensor detects movement in the area being monitored. When movement is detected, the sensor sends a signal to the Raspberry Pi.
- **Image Capture:** The Raspberry Pi receives the signal from the motion sensor and triggers the camera to take a picture of the area being monitored.

- **Image Processing:** The Raspberry Pi then processes the captured image to identify any potential intruders. This may involve using face recognition software or simply looking for features that are indicative of a human presence.
- **Alert Generation:** If the system determines that an intruder is present, it will generate an alert. This alert may be sent to the user's smartphone, email, or another designated notification system.

OUTPUT: