Intruder Detection and Alarm System Using IoT

Introduction:

In the world of Internet of Things (IoT) when we have all the technologies to revolutionize our life. There are many types of good security systems and cameras out there for home security but they are much expensive so today we will build a low cost simple Raspberry Pi based Intruder Alert System, which not only alert you through an email but also sends the picture of Intruder when it detects any.

In this IoT based Project, we built a Home Security System using PIR Sensor and PI Camera. This system will detect the presence of Intruder and quickly alert the user by sending him a alert mail. This mail will also contain the Picture of the Intruder, captured by Pi camera. Raspberry Pi is used to control the whole system. This system can be installed at the main door of your home or office and you can monitor it from anywhere in the world using your Email over internet.

Components Required:

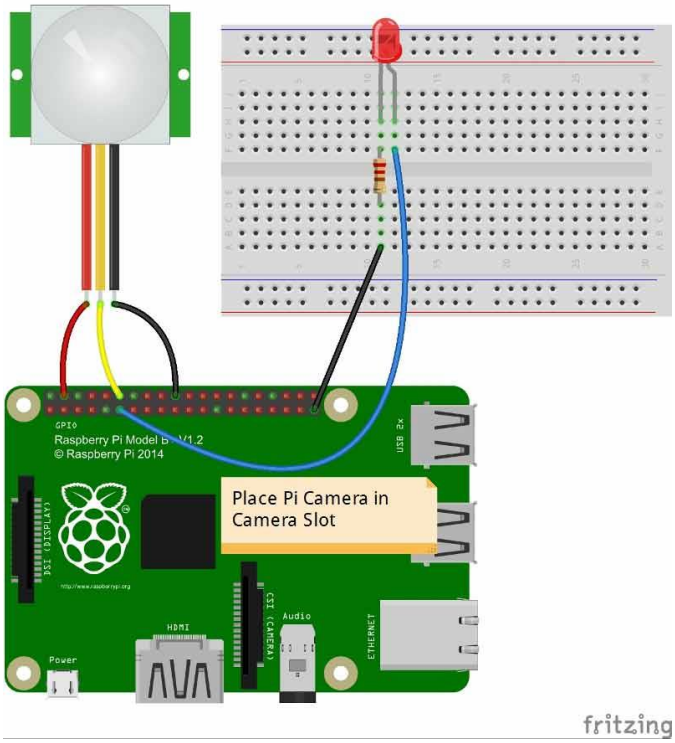
- Raspberry Pi
- Pi Camera
- PIR Sensor
- LED
- Bread Board
- Resistor (1k)
- Connecting wires
- Power supply

Working Explanation:

PIR Sensor**:** It's a 3 pin sensor where there's a ground pin , a signal pin and the power pin. It has a average range of 3-12 meters. It works on 5 volts. PIR are fundamentally made of a pyro electric sensor, which can detect levels of infrared radiation. So it detects the infrared light radiated by a warm object back and reacts when there's some objects in motion in front. And working of this Project is very simple. A PIR sensor is used to detect the presence of any person and a Pi Camera is used to capture the images when the presence it detected. Whenever anyone or intruder comes in range of PIR sensor, PIR Sensor triggers the Pi Camera through Raspberry Pi. Raspberry pi sends commands to Pi camera to click the picture and save it. After it, Raspberry Pi creates a mail and sends it to the defined mail address with recently clicked images. The mail contains a message and picture of intruder as attachment. Here we have used the message "Please find the attachment", you can change it accordingly in the Code given at the end. Here the pictures are saved in Raspberry Pi with the name which itself contains the time and date of entry. So that we can check the time and date of intruder entry by just looking at the Picture name, check the images below. If you are new with Pi Camera then check our previous tutorial on Visitor Monitoring System with Pi Camera.

Circuit Diagram:

In this system, we only need to connect Pi Camera module and PIR sensor to Raspberry Pi **3**. Pi Camera is connected at the camera slot of the Raspberry Pi and PIR is connected to GPIO pin 18. A LED is also connected to GPIO pin 17 through a 1k resistor.



Installing the Libraries/set up:

First we have to install the libraries required for the Raspberry Pi to run this program.

- **<u>Installing pi camera library</u>**: Commands used

  Sudo apt-get install python-pi camera
  Sudo apt-get installpython3-picamera

  After it, user needs to enable Raspberry Pi Camera by using Raspberry Pi Software Configuration Tool (raspi-config).

  Sudo raspi-config

  From here we have to enable the camera.

- **Setting up ssmtp libraries:** Now after setting up the Pi Camera, we will install software for sending the mail. Here we are using *ssmtp* which is an easy and good solution for sending mail using command line or using Python Script**.** We need to install two Libraries for sending mails using SMTP:

  Sudo apt-get install ssmtp

  Sudo apt-get install mailutils

  After this we have to config the ssmtp library by the command ssmtp.config and save it. The commands are

  Sudo nano /etc/ssmtp/ssmtp.config

  And edit the following commands:

  Root= krishnanew99@gmail.com

  Mailhub= ssmtp.gmail.com:587

  Hostname= raspberrypi

  AuthUser= krishnanew99@gmail.com

  AuthPass= ********

  FromLineOverride= Yes

  userSTARTTLS= YES

  userTLS= YES

- **Installing MIME library:** MIME(Multipurpose Internet Mail Extensions) is used to send mails in a more simple method as the smtplib is enough to send massages.

  ```
  Import RPi.GPIO as gpio
  Import picamera
  import time
  import smtplib
  from email.mime.multipart import MIMEMultipart
  from email.mime.text import MIMEText
  from email.mime.base import MIMEBase
  from email import encoders
  from email.mime.image import MIMEImage
  ```

  This snippet can also be added in the main code also thus is not needed to install the libraries deparately. Here is the initial code of the program below:

Main code:

```
import RPi.GPIO as gpio
import picamera
import time
import smtplib
from email.mime.multipart import MIMEMultipart
from email.mime.text import MIMEText
from email.mime.base import MIMEBase
from email import encoders
from email.mime.image import MIMEImage

fromaddr = "recievet@gmail.com"
toaddr = "aj.mahera007@gmail.com"

mail = MIMEMultipart()

mail['From'] = fromaddr
mail['To'] = toaddr
mail['Subject'] = "Attachment"
body = "Please find the attachment"
led=17
pir=18
HIGH=1
LOW=0
gpio.setwarnings(False)
gpio.setmode(gpio.BCM)
gpio.setup(led, gpio.OUT)           # initializing GPIO Pin as outputs
gpio.setup(pir, gpio.IN)           # initializing GPIO Pin as input
data=""
def sendMail(data):
    mail.attach(MIMEText(body, 'plain'))
    print (data)
    dat='%s.jpg'%data
    print (dat)
    attachment = open(dat, 'rb')
    image=MIMEImage(attachment.read())
    attachment.close()
    mail.attach(image)
    server = smtplib.SMTP('smtp.gmail.com', 587)
    server.starttls()
    server.login(fromaddr, "testr1234@")
    text = mail.as_string()
    server.sendmail(fromaddr, toaddr, text)
    server.quit()
def capture_image():
    data= time.strftime("%d_%b_%Y|%H:%M:%S")
    camera.start_preview()
    time.sleep(5)
    print (data)
```

```
    camera.capture('%s.jpg'%data)
    camera.stop_preview()
    time.sleep(1)
    sendMail(data)
gpio.output(led , 0)
camera = picamera.PiCamera()
camera.rotation=180
camera.awb_mode= 'auto'
camera.brightness=55
while 1:
    if gpio.input(pir)==1:
        gpio.output(led, HIGH)
        capture_image()
        while(gpio.input(pir)==1):
            time.sleep(1)

    else:
        gpio.output(led, LOW)
        time.sleep(0.01)
```

Conclusion:

This program is efficient in terms of price, power consumption and as well as set up. It’s also effective for the usage. Though the range may be handy but its worth a try to use it in real life.so, it's a great idea to develop a system which can be controlled and monitored from anywhere.