IOT-TRAFFIC MANAGEMENT

Name:R.Bavithra

Id:aut962921104704

Email id:pavi2004gopal@gmail.com

TABLE CONTENT

- Introduction.......................
- Project objectives................
- IOT-device.........................
- Sensor setup......................
- Mobile app development.........
- Code implementation.............
- Script of raspberry pi.............
- Working of app.....................
- Benefits of traffic management...........
- Conclusions..................

Introduction:

Traffic management is the organisation, arrangement, guidance and control of both stationary and moving traffic, including pedestrians, bicyclists and all types of vehicles. Its aim is to provide for the safe, orderly and efficient movement of persons and goods, and to protect and, where possible, enhance the quality of the local environment on and adjacent to traffic facilities. This book is an introduction to traffic management, written in laypersons' language, and assuming no background knowledge of the subject. Various basic traffic characteristics relating to road users, vehicles and roads, and traffic regulation and control, are discussed, including some traffic volume and traffic flow considerations relevant to traffic management. For effective traffic management, it is essential that the practitioner works from factual information. Road inventory and statistical methods, and the more common types of traffic studies, including traffic volume and composition, origin and destination, speed, travel time and delay, accidents and parking are described. "Before and after" studies, and estimation of future traffic are also covered. As a basis for logically applying traffic management techniques it is necessary to develop a classification or hierarchy of all roads to ensure that the primary purpose of each of them is defined, agreed and understood. A functional classification of roads suitable for traffic management purposes, and a process for developing such a system is described. Several chapters go on to discuss various aspects of traffic management, including signing and delineation, pedestrian facilities, bicycle facilities, intersections, traffic signals, road capacity, parking, roadside safety and roadway lighting. The objectives of local area traffic management schemes, and a systematic process for developing them are described, and the various techniques that may be used and the principles of design of traffic management devices are summarised. The application of traffic management techniques to rural and urban arterial roads respectively is discussed, emphasising the desirability of treating routes or networks as a whole rather than simply focussing on isolated problem spots. Past and likely future trends in road travel, and various techniques for travel demand management are described. While these sorts of techniques are well known, and their use should be encouraged, they are unlikely to have much effect on travel in Australia at least for the foreseeable future. The important area of traffic enforcement and the associated aspects of education and encouragement are considered. Unless traffic management is logically applied and consistently enforced, it will not be effective. Enforcement must be considered an integral part of traffic management. (TRRL)

Project objectives:

Road Network Operations is a strategic approach to maximising efficiency on existing and future road infrastructure. At the tactical level this strategy translates into improving operations with the objective of reducing traffic delays and operating day-by-day in a more efficient way. At the strategic level, it means integrating the operations concept early on in the development of all road infrastructure projects, beginning with the planning and design process and ensuring adequate resources, both fiscal and personnel.

The road network operator has to consider the methods, organisational structures and resources needed to support strategies for road network operations, maintenance and incident response.

Objectives of the network operator include:

- improving safety on the road network
- optimising traffic flow on arterial and freeway networks
- reducing congestion within and between cities
- co-ordinating agency traffic/transit operations
- managing incidents, reducing delays and adverse effects of incidents and congestion, weather, roadwork, special events, emergencies and disaster situations
- effectively managing maintenance and construction work to minimise the impact on safety and congestion
- informing travellers with timely and accurate information
- improving the interfaces between modes of transport for passengers and freight
- eliminating bottlenecks due to inadequate road geometry
- providing reliable and convenient public transport services

Congestion:

Relieving congestion is achieved by optimizing the management of traffic signals; detecting and managing incidents on the highway network, access control systems; High-Occupancy Vehicle (HOV) lanes; journey time information; speed management.

Safety:

Improving safety requires measures such as adaptive speed control, collision detection and avoidance; enhanced vehicle safety systems; weather and road condition information.

Security:

Maintaining security is done through evacuation route signing and priority; homeland security initiatives such as deployed in the USA, hazardous load monitoring and assistance for vulnerable road users.

Environmental protection:

Ensuring environmental monitoring and protection requires a reduction in traffic congestion, creation of low-emission zones and promotion of public transport alternatives.

Support for business and commerce:

Increasing productivity and operational efficiencies can be achieved by fleet management; computer aided dispatch; automatic vehicle location; automatic cargo tracking; electronic pre-clearance; vehicle compliance checking and driver monitoring.

Road user services:

Providing comfort to users of transportation system who need to feel confident and secure is the motivation for applications such as route confirmation, journey time estimates and clear advice on approaching interchanges and connections. Relevant ITS services include real-time traffic and public transport information; dynamic route guidance; automotive vehicle location (AVL); smart card payment systems for toll highway and public transport use.

IOT-device:

This intelligent system comprises several components, including wireless sensors, RFID tags, and BLE beacons installed at the traffic signals to monitor the movement of vehicles. A real-time data analytics tool connects the Geographic Information System (GIS-enabled) digital roadmap with control rooms for real-time traffic monitoring.

The smart traffic management system captures the images of vehicles at the signals using the digital image processing technique. This data is then transferred to the control room via wireless sensors. The system also leverages BLE beacons or RFID tags to track the movement of vehicles and keep traffic congestion in control, track down stolen vehicles and even clear the road for emergency vehicles that are installed with RFID readers.

Sensor setup:

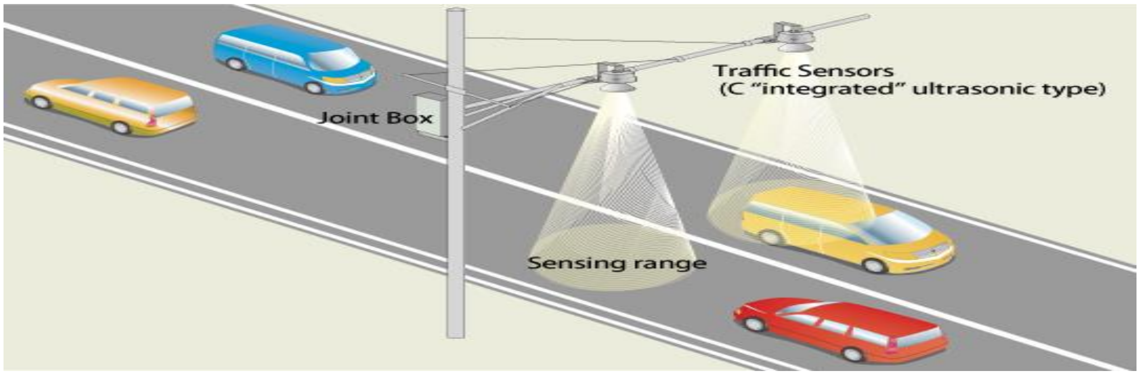
Information Supply in Real Time

Traffic sensors are used to collect road traffic information for signal control, and the data collected is also supplied to the Vehicle Information and Communication System Center (VICS Center)

Pedestrian sensors provide security for pedestrians by replacing pushbuttons and sensing pedestrians crossing the street to extend the time of green light.

Traffic Sensors (ultrasonic type)

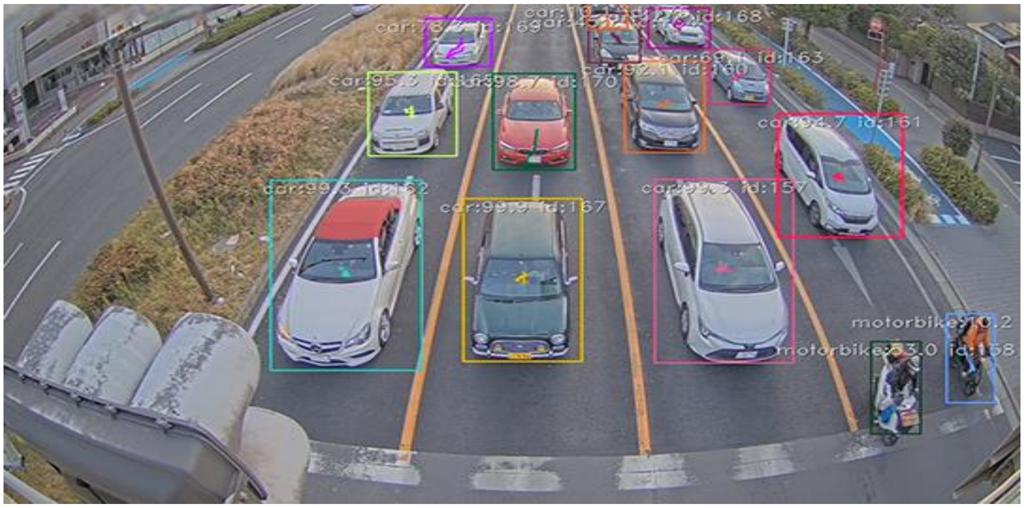
These sensors detect the presence of a vehicle using the reflection of ultrasonic waves transmitted from ultrasonic transmitters/receivers installed directly above the road. They are used for the purpose of measuring traffic volume and occupancy rates by means of a stable sensory function.



Traffic sensors (image system)

The image-based vehicle sensor has a maximum measuring area of 30 meters, and uses AI (deep learning)-based object detection technologies to detect vehicles.

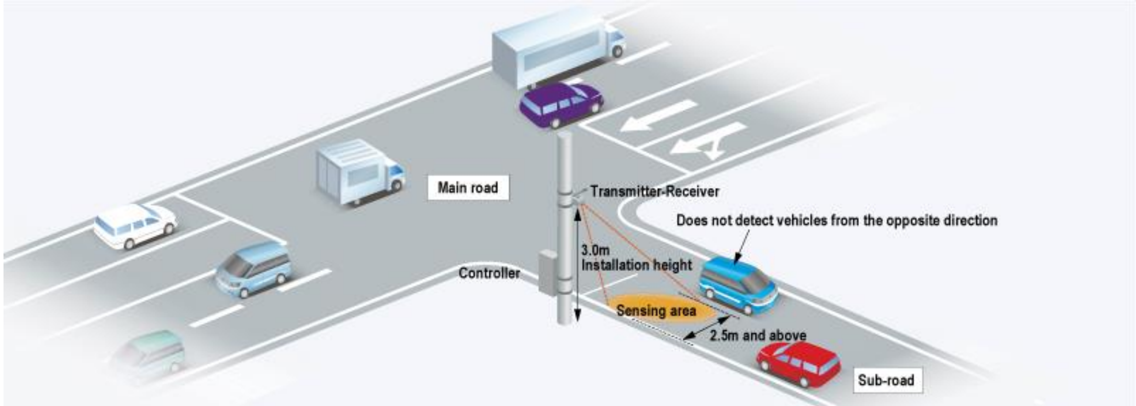
In addition to measuring traffic congestion in the immediate vicinity of the vehicle, this sensor is used for right turn sensing control, which involves detecting vehicles in right-turning lanes and optimally changing the amount of time for green right turn arrows based on whether vehicles are detected, and recall control, which only turns traffic lights on subsidiary roads green when detecting vehicles on subsidiary roads with minimal traffic.



Traffic Sensors (Doppler type)

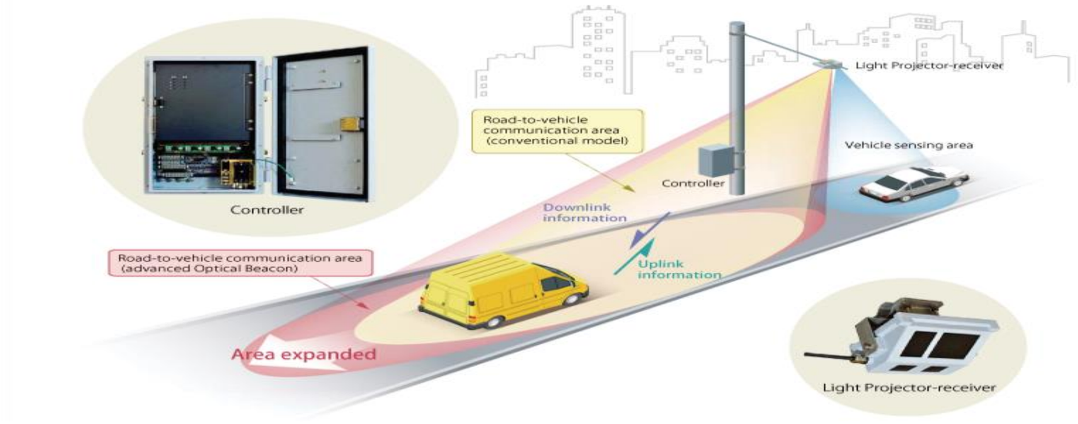
These sensors use the ultrasonic Doppler effect. They detect vehicles travelling in a particular direction using a change in frequency (the Doppler effect) according to vehicle.

They are installed on side roads with low traffic volume, and are used for recall control to change the traffic light on the side road to green only when a vehicle is detected.



Optical beacon

The optical beacon is a key infrastructure for supplying information to road traffic facilities.



It supports the road vehicle cooperation system (a system that provides various support for safe driving and controls traffic flow by means of road-to-vehicle communication between optical beacons and onboard devices), can alleviate traffic congestion and help reduce road traffic accidents and co2.

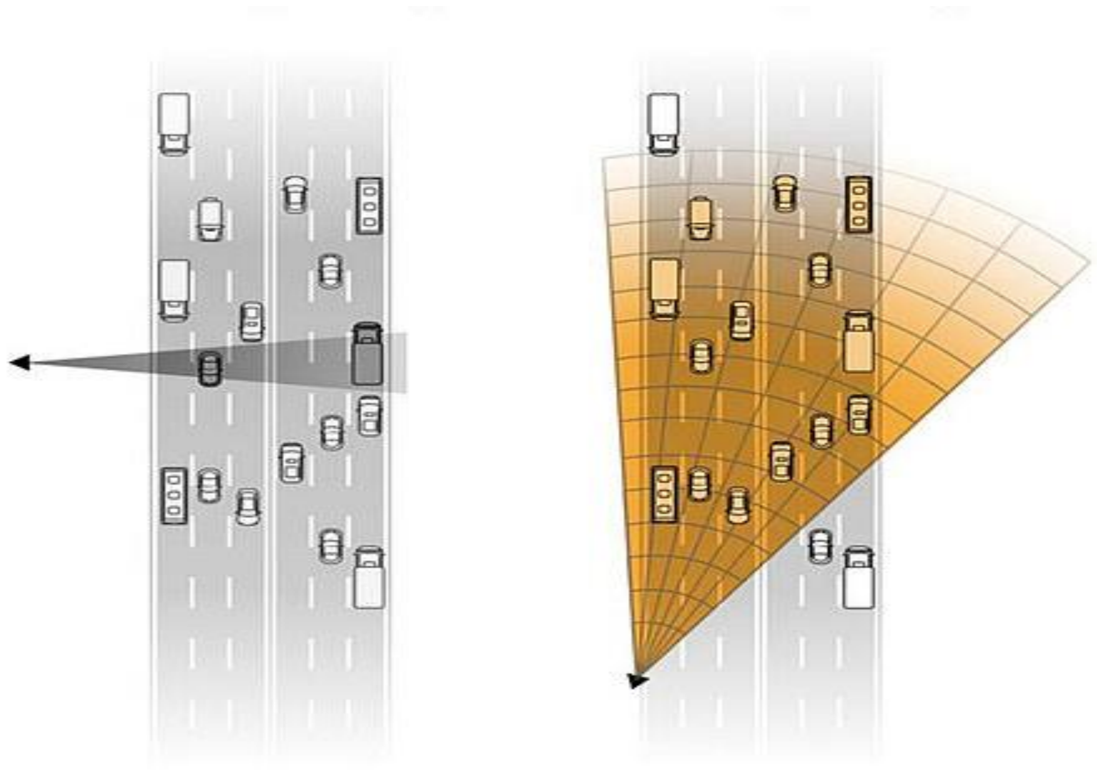
FORWARD FIRING

Unlike older side firing sensors, the modern smartmicro sensors use multiple forward firing beams which has many advantages.

Vehicles remain inside the field of view for much longer, their position and speed vectors are measured with higher accuracy, and a better classification is achieved. Our sensors are the only technology available on the market today that supports precise trajectory tracking of vehicles. Due to their high coverage, vehicle positions can be tracked for up to 300m and over a 100-degree field of view.

Looking along the road, the mounting position is more flexible. On the roadside, at the corner of an intersection, at the median of a highway or on a gantry, whatever sensor position: forward firing provides the best coverage.

Existing infrastructure like traffic light poles or street light poles can be used. Other than for obsolete side-firing technology which usually need a very high mounting position, moderate heights are sufficient. Since vehicles are tracked over a longer period when they drive in the field of view, occlusion rarely happens.



3D MEASUREMENT:

smartmicro sensors featuring high-definition 3D technology provide Cartesian coordinates, velocity vector, range and azimuth angle simultaneously for all traffic objects within the field of view.

Our 3-dimensional radar detection principle measures:

- Direct unambiguous Doppler (speed)
- Direct range
- Direct azimuth (horizontal) angle

4D measurement:

In addition to these three dimensions, our 4D measurement includes the elevation angle simultaneously for all traffic objects within the field of view. The additional measurement of the elevation angle indicates the vehicle height.

Our 4-dimensional radar detection principle measures:

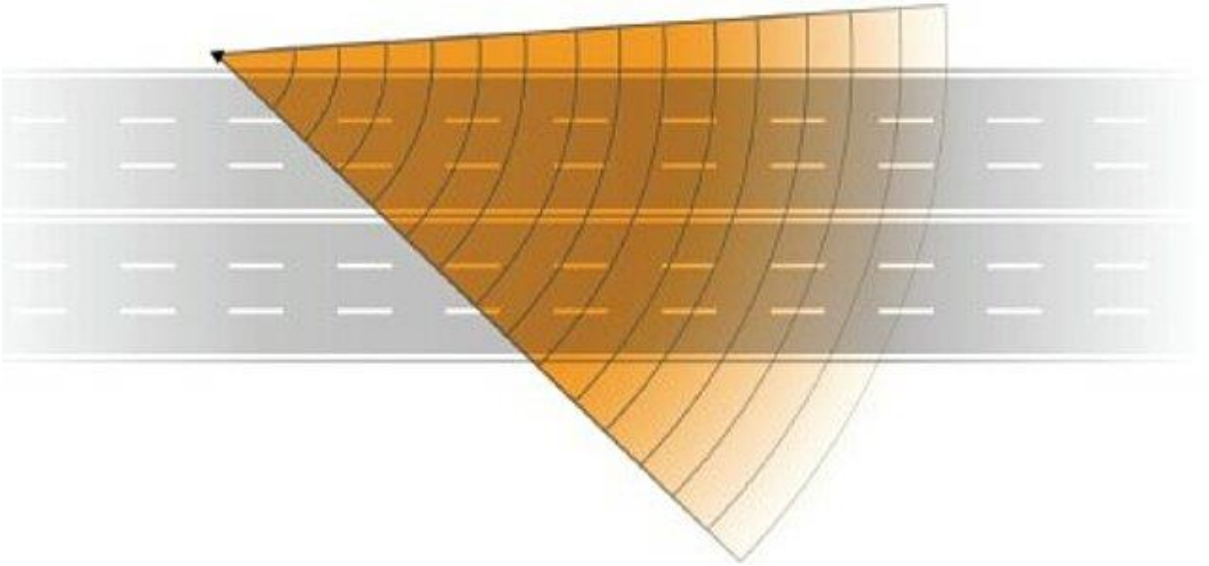
- Direct unambiguous Doppler (speed)

- Direct range
- Direct azimuth (horizontal) angle
- Direct elevation (vertical) angle

HIGH DEFINITION (HD):

Most other detectors on the market separate objects by only one parameter, for example either range or speed. Featuring HD technology smartmicro sensors separate objects by both speed and range. This leads to robust performance especially in dense traffic conditions.

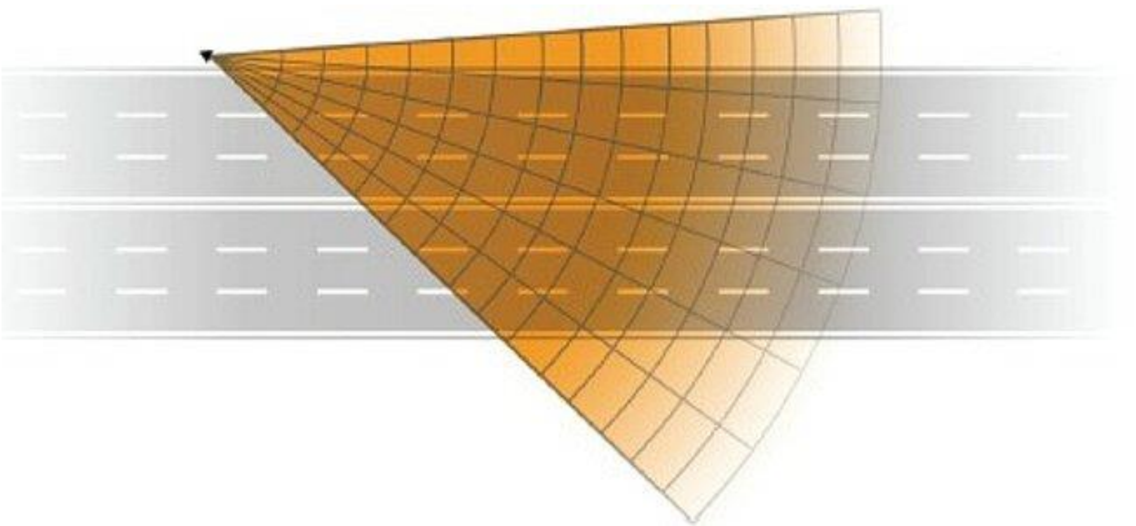
- Separation in speed
- Separation in range



Ultra-high definition (UHD):

smartmicro sensors featuring ultra-high definition (UHD) technology achieve the highest performance available today and outperform other traffic radars. UHD radar technology enables to separate objects by speed, range and azimuth angle and to measure elevation angles.

- Separation in speed
- Separation in range
- Separation in angle



Mobile app development:

Szczecin Traffic Manager application has been developed to support the needs of Drivers traveling around Szczecin, particularly in terms of early display of messages shown on each variable message sign installed within the city as part of the above-described Traffic Management System for Szczecin. Proper operation of application requires constant access to the Internet, both in terms of packet data transmission and Wi-Fi connection. It also requires GPS module. Additionally, the device supporting the application (smartphone or tablet) must be equipped with an accelerometer and magnetometer. Application uses class diagram in UML modelling (Figure 3) in order to illustrate the scope of the types of objects used in applications designed for a user. Its

Consulting

Choose a from-scratch tool or upgrade your existing one? What solutions will enhance your business performance? Need a second opinion? Our team is ready to answer your questions during the consultation.

Custom Software Development

Business needs vary, and so should the software! With our custom traffic management software development, have full control of your solution's design, functionality, performance, scalability, and maintenance.

Third-Party Integration

The market is full of tools that can improve your product usability and efficiency. Our engineers will connect the third-party tools with the existing system and develop risk management strategies to ensure stable functioning.

Support and Maintenance

Whether it’s Relevant-native or developed by other vendors, we offer maintenance services for any solution. We monitor the system updates, legal requirement changes, protocol upgrades, and more to keep the function

```
include <conio.h>
include <iostream.h>
#
```

Code implementation:

Source Code

```
import cv2
import dlib
import time
import threading
import math
import helm
carCascade = cv2.CascadeClassifier('cars.xml')
 bikeCascade = cv2.CascadeClassifier('motor-v4.xml')
video = cv2.VideoCapture('test.mp4')

LAG=7
WIDTH = 1280
 HEIGHT = 720
  OPTIMISE= 7
  def estimateSpeed(location1, location2,fps):
   d_pixels = math.sqrt(math.pow(location2[0] -
  location1[0], 2) +
   math.pow(location2[1] - location1[1], 2))
  # ppm = location2[2] / carWidht
  ppm = 8.8
   d_meters = d_pixels / ppm
  if fps == 0.0:
  fps = 18 speed = d_meters * fps * 3.6
  return speed
  def trackMultipleObjects():
```

```
rectangleColor = (0, 255, 0)
frameCounter = 0
 currentCarID = 0
 currentBikeID=0
fps = 0
carTracker = {}
bikeTracker = {}
bikeNumbers = {}
 carNumbers = {}
 bikeLocation1 = {}
 carLocation1 = {}
bikeLocation2 = {}
carLocation2 = {}
speed = [None] * 1000
go =[False for i in range(1000)]
 identity = [0 for i in range(1000)]
snaps = [False for i in range(1000)]
types = ["cars" for i in range(1000)]
Helmets = ["No Helmet Detected" for i in range(1000)]
 out
=cv2.VideoWriter('outpy.avi',cv2.VideoWriter_fourcc('M','J
','P','G'), 10, (WIDTH,HEIGHT))
while True: start_time = time.time()
rc, image = video.read()
 if type(image) == type(None):
break
image = cv2.resize(image, (WIDTH, HEIGHT))
 resultImage = image.copy()
```

```
frameCounter = frameCounter + 1

 carIDtoDelete = []

for carID in carTracker.keys():

 trackingQuality = carTracker[carID].update(image)

if trackingQuality < 7:

 carIDtoDelete.append(carID)

for carID in carIDtoDelete:


trackers.') print ('Removing carID ' + str(carID) + ' from
list of trackers.')

location.') print ('Removing carID ' + str(carID) + '
previous location.')

location.') print ('Removing carID ' + str(carID) + '
current location.')

carTracker.pop(carID, None)

 carLocation1.pop(carID, None)

 carLocation2.pop(carID, None)

if not (frameCounter % 10):

Gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)

cars = carCascade.detectMultiScale(gray1.1, 13, 18, (24,
24))

bikes = bikeCascade.detectMultiScale(gray, 1.1 , 13, 18,
(24,24))

for (_x, _y, _w, _h) in cars:

x = int(_x)

y = int(_y)

 w = int(_w)

h = int(_h)

 roi = image[y:y+h,x:x+w]
```

```
x_bar = x + 0.5 * w

y_bar = y + 0.5 * h

 matchCarID = None

for carID in carTracker.keys(): trackedPosition = carTracker[carID].get_position()

t_x = int(trackedPosition.left())

 t_y = int(trackedPosition.top())

 t_w = int(trackedPosition.width())

t_h = int(trackedPosition.height())

t_x_bar = t_x + 0.5 * t_w

t_y_bar = t_y + 0.5 * t_h

if ((t_x <= x_bar <= (t_x + t_w)) and (t_y <= y_bar <= (t_y + t_h)) and (x <= t_x_bar <= (x + w)) and (y <= t_y_bar <= (y + h))):

matchCarID = carID

if matchCarID is None: print ('Creating new tracker ' + str(currentCarID))

tracker = dlib.correlation_tracker()
tracker.start_track(image, dlib.rectangle(x, y, x + w, y + h))

carTracker[currentCarID] = tracker
carLocation1[currentCarID] = [x, y, w, h]

types[currentCarID]= "bikes"

 currentCarID = currentCarID + 1

for carID in carTracker.keys():

trackedPosition = carTracker[carID].get_position()

t_x = int(trackedPosition.left())

t_y = int(trackedPosition.top())

t_w = int(trackedPosition.width())

t_h = int(trackedPosition.height())
```

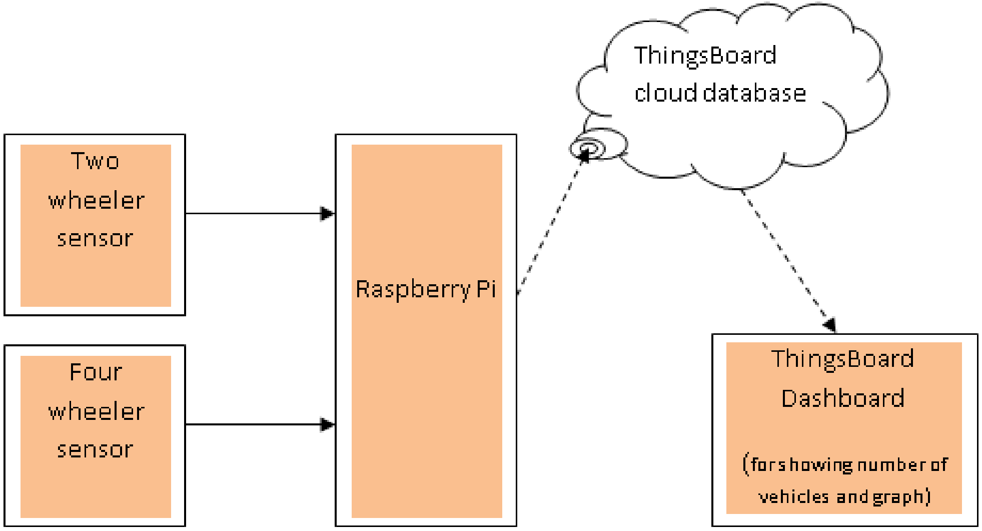
```
cv2.rectangle(resultImage, (t_x, t_y), (t_x + t_w, t_y +
t_h), rectangleColor, 4)

carLocation2[carID] = [t_x, t_y, t_w, t_h]

end_time = time.time()

fps=0.0

for i in carLocation1.keys():

if frameCounter % 1 == 0:

 [x1, y1, w1, h1] = carLocation1[i]

[x2, y2, w2, h2] = carLocation2[i]

carLocation1[i] = [x2, y2, w2, h2]

if [x1, y1, w1, h1] != [x2, y2, w2, h2]:

result = False

roi = resultImage[y1:y1+h1,x1:x1+w1]

if types[i]=="bikes" and Helmets[i] == "No Helmet
Detected" and identity[i]< OPTIMISE:

result = helm.detect(roi)

 if result==True:

if 7==7:

 if not (end_time == start_time):

fps = 1.0/(end_time - start_time)

speed[i] = estimateSpeed([x1, y1, w1, h1], [x2, y2, w2,
h2],fps)

if int(speed[i])>40:

speed[i]= speed[i]%40

 if go[i] == True and int(speed[i])<10:
speed[i]=speed[i]+15

if int(speed[i])==0:

continue

if int(speed[i])>30:
 go[i]=True

cv2.putText(resultImage, "OverSpeeding ALERT", (int(x1 +
w1/2), int(y1-5)),cv2.FONT_HERSHEY_SIMPLEX, 0.75, (0, 0,
255), 2)

elif speed[i] != None and y1 >= 180 and speed[i]!=0:

ans= str(int(speed[i])) + " km/hr "

if types[i]=="bikes":

ans= ans+ Helmets[i] cv2.putText(resultImage, ans, (int(x1
+ w1/2), int(y1-5)),cv2.FONT_HERSHEY_SIMPLEX, 0.75, (0,
255, 0), 2) identity[i]+=1 cv2.imshow('result',
resultImage) if cv2.waitKey(33) == 27:

break

cv2.destroyAllWindows()

 if __name__ == '__main__':

trackMultipleObjects()
```

Script of raspberry pi:

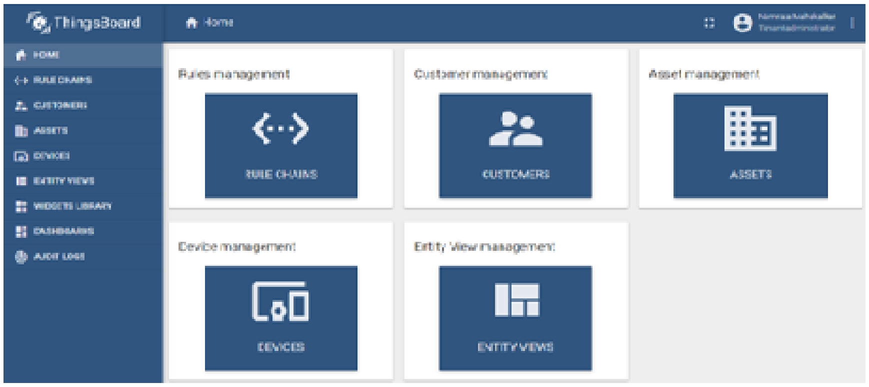
```
#import all necessary libraries

import paho.mqtt.client as mqtt

import time,json

import time

import RPi.GPIO as GPIO

import json

import array

#define raspberry pi GPIO pins as input or output

GPIO.setmode(GPIO.BOARD)

GPIO.setup(3 , GPIO.IN) # set GPIO25 as input (button)

GPIO.setup(7 , GPIO.OUT)

#global num_of_veh

#global num_of_veh_per_min

num_of_veh = 0
```

```
num_of_veh_per_min= 0
def on_log(client, userdata, level, buf):
print(buf)
def on_connect(client, userdata, flags, rc):
if rc==0:
client.connected_flag=True #set flag
print("connected OK")
else:
print("Bad connection Returned code=",rc)
client.loop_stop()
def on_disconnect(client, userdata, rc):
print("client disconnected ok")
def on_publish(client, userdata, mid):
print("In on_pub callback mid= " ,mid)
count=0
mqtt.Client.connected_flag=False#create flag in class
mqtt.Client.suppress_puback_flag=False
client = mqtt.Client("python1") #create new instance
#client.on_log=on_log
client.on_connect = on_connect
client.on_disconnect = on_disconnect
client.on_publish = on_publish
broker="demo.thingsboard.io"
port =1883
topic="v1/devices/me/telemetry"
#need to edit user name
username="xyx" #device house --- Enter your own user name
password=""
if username !="":
pass
client.username_pw_set(username, password)
client.connect(broker,port) #establish connection
while not client.connected_flag: #wait in loop
client.loop()
time.sleep(.1)
time.sleep(.3)
data=dict()
# set GPIO24 as an output (LED)
#for i in range(100):
while True:
num_of_veh_per_min= 0
for i in range(6000): # the counter is incrementing every 10 mSec. the data is updated every 60 Sec i.e. every 1 min. Hence 6000 * 10 mSec = 1 min
channel=GPIO.wait_for_edge(3, GPIO.RISING, timeout= 10)
# wait if a vehicle passes with a time out of 10mSec - debounce time
if channel is None:
if i%100 == 0: # print the timer value after every second beacuse printing timer value every 10mSec increases delay
print('Data will be uploaded after ',60 - i/100,' Seconds')
else:
num_of_veh = num_of_veh +1 # increment the total number of vehicles
num_of_veh_per_min=num_of_veh_per_min+1 # increment the vehicle per min count
print("Number of vehicle",num_of_veh) # print the total numner of vehicle counted till now
time.sleep(0.01)
data["num_of_veh"]=num_of_veh
data["num_of_veh_per_min"]=num_of_veh_per_min
data_out=json.dumps(data) #create JSON object
print("publish topic",topic, "data out= ",data_out)
ret=client.publish(topic,data_out,0) #publish
client.loop()
client.disconnect()
```

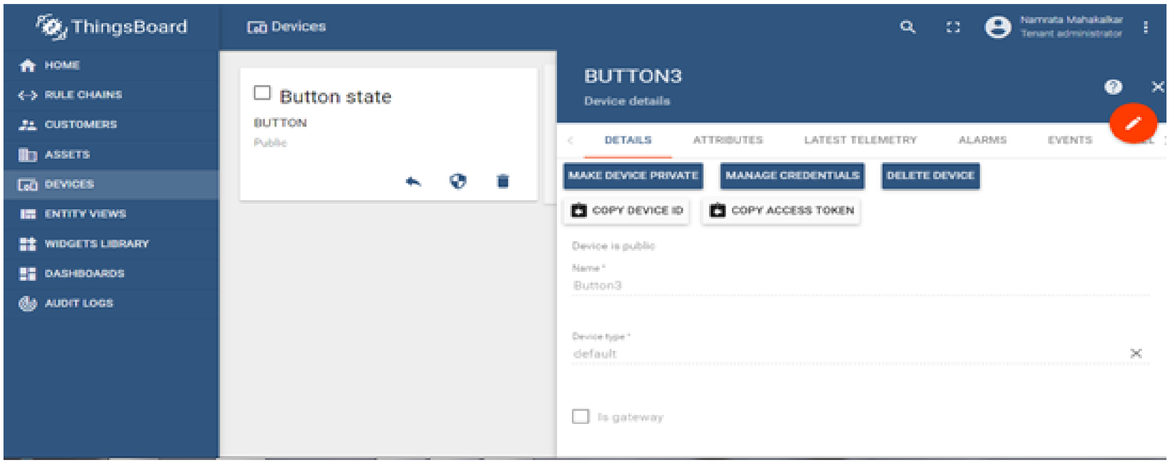
Block diagram :



ThingsBoard: Home page



Things board: Create new device



□ Button state

BUTTON

Public

BUTTON3

Device details

Device is public

Name *

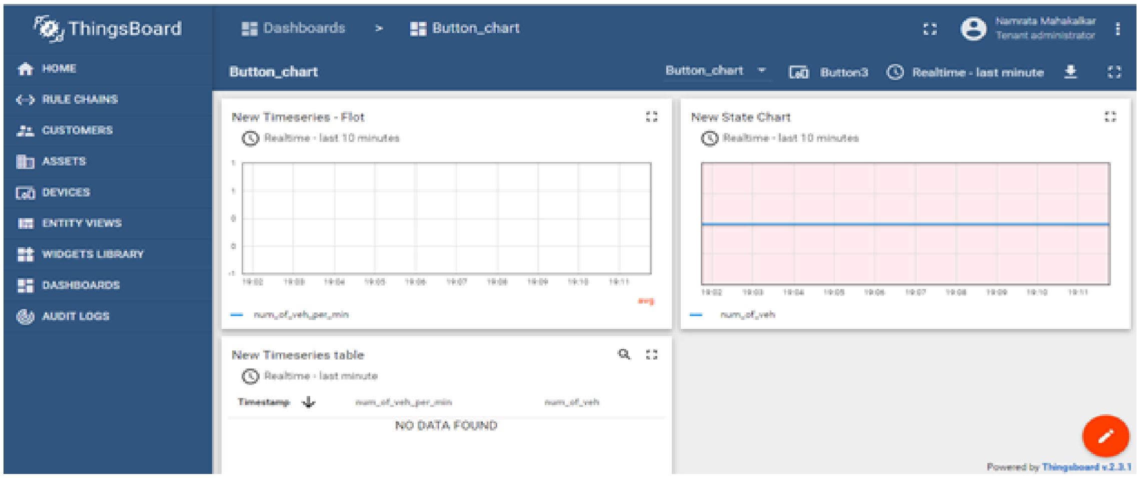
Button3

Device type *

default

□ Is gateway

ThingsBoard: Create new dashboard



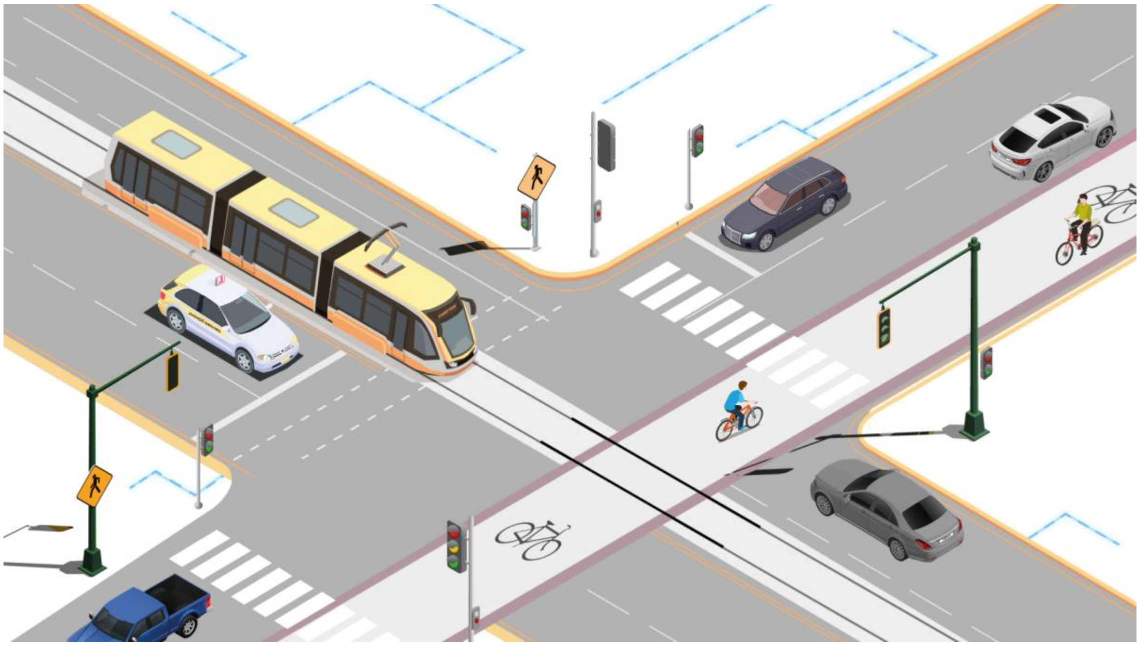
Timestamp ↓	num_of_veh_per_min	num_of_veh

NO DATA FOUND

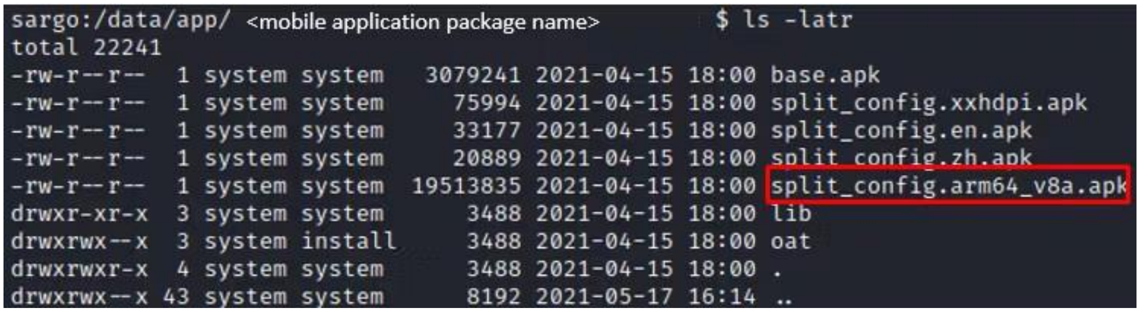
Results on ThingsBoard dashboard:

Here two charts for the number of vehicle per minute and number of total vehicle passed with time are shown respectively.

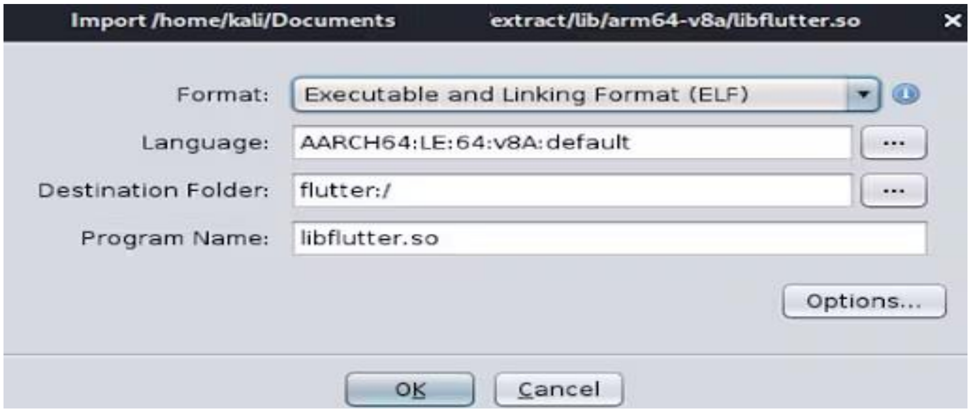
Third chart is a card shows number of vehicle per minute in numerical



Framework in flutter:



```
sargo:/data/app/ <mobile application package name>       $ ls -latr
total 22241
-rw-r--r--   1 system system   3079241 2021-04-15 18:00 base.apk
-rw-r--r--   1 system system     75994 2021-04-15 18:00 split_config.xxhdpi.apk
-rw-r--r--   1 system system     33177 2021-04-15 18:00 split_config.en.apk
-rw-r--r--   1 system system     20889 2021-04-15 18:00 split_config.zh.apk
-rw-r--r--   1 system system  19513835 2021-04-15 18:00 split_config.arm64_v8a.apk
drwxr-xr-x   3 system system      3488 2021-04-15 18:00 lib
drwxrwx--x   3 system install     3488 2021-04-15 18:00 oat
drwxrwxr-x   4 system system      3488 2021-04-15 18:00 .
drwxrwx--x  43 system system      8192 2021-05-17 16:14 ..
```



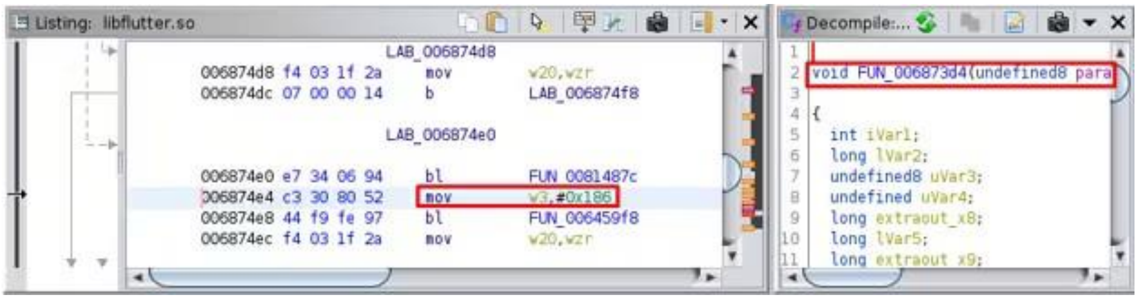
Import /home/kali/Documents extract/lib/arm64-v8a/libflutter.so

Format: Executable and Linking Format (ELF)

Language: AARCH64:LE:64:v8A:default

Destination Folder: flutter:/

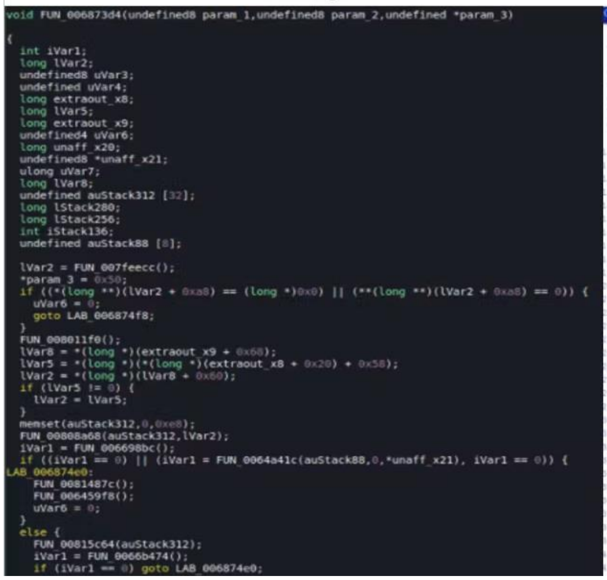
Program Name: libflutter.so



```
                     LAB_006874d8
006874d8 f4 03 1f 2a     mov     w20,wzr
006874dc 07 00 00 14     b       LAB_006874f8

                     LAB_006874e0

006874e0 e7 34 06 94     bl      FUN_0081487c
006874e4 c3 30 80 52     mov     w3,#0x186
006874e8 44 f9 fe 97     bl      FUN_006459f8
006874ec f4 03 1f 2a     mov     w20,wzr
```

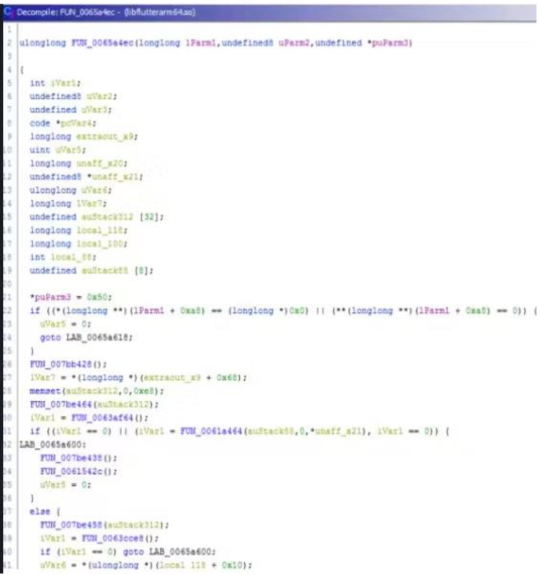
```

void FUN_006873d4(undefined8 para

{
  int iVar1;
  long lVar2;
  undefined8 uVar3;
  undefined uVar4;
  long extraout_x8;
  long lVar5;
  long extraout_x9;
```

Horangi

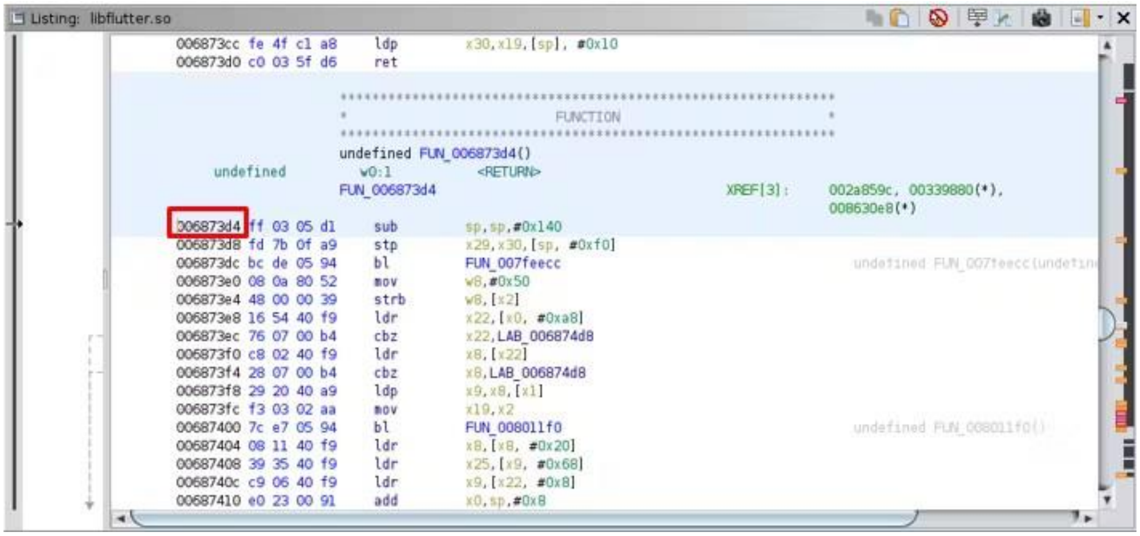


```
void FUN_006873d4(undefined8 param_1,undefined8 param_2,undefined *param_3)

{
  int iVar1;
  long lVar2;
  undefined8 uVar3;
  undefined uVar4;
  long extraout_x8;
  long lVar5;
  long extraout_x9;
  undefined4 uVar6;
  long unaff_x20;
  undefined8 *unaff_x21;
  ulong uVar7;
  long lVar8;
  undefined auStack312 [32];
  long lStack280;
  long lStack256;
  int iStack136;
  undefined auStack88 [8];

  lVar2 = FUN_007feecc();
  *param_3 = 0x50;
  if ((*(long **)(lVar2 + 0xa8) == (long *)0x0) || (**(long **)(lVar2 + 0xa8) == 0)) {
    uVar6 = 0;
    goto LAB_006874f8;
  }
  FUN_008011f0();
  lVar8 = *(long *)(extraout_x9 + 0x68);
  lVar5 = *(long *)(*(long *)(extraout_x8 + 0x20) + 0x58);
  lVar2 = *(long *)(lVar8 + 0x60);
  if (lVar5 != 0) {
    lVar2 = lVar5;
  }
  memset(auStack312,0,0xe8);
  FUN_00808a68(auStack312,lVar2);
  iVar1 = FUN_006698bc();
  if ((iVar1 == 0) || (iVar1 = FUN_0064a41c(auStack88,0,*unaff_x21), iVar1 == 0)) {
LAB_006874e0:
    FUN_0081487c();
    FUN_006459f8();
    uVar6 = 0;
  }
  else {
    FUN_00815c64(auStack312);
    iVar1 = FUN_0066b474();
    if (iVar1 == 0) goto LAB_006874e0;
```

blog.nviso.eu

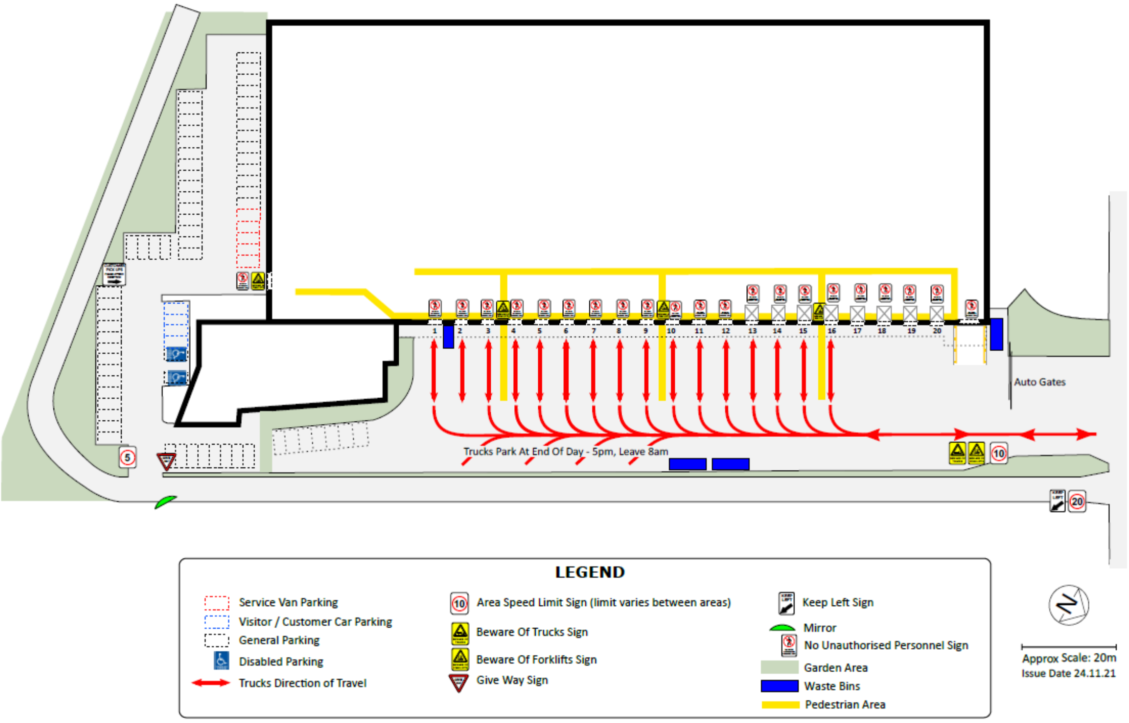


```

ulonglong FUN_0065a4ec(longlong lParm1,undefined8 uParm2,undefined *puParm3)


{
  int iVar1;
  undefined8 uVar2;
  undefined uVar3;
  code *pcVar4;
  longlong extraout_x9;
  uint uVar5;
  longlong unaff_x20;
  undefined8 *unaff_x21;
  ulonglong uVar6;
  longlong lVar7;
  undefined auStack312 [32];
  longlong local_118;
  longlong local_100;
  int local_88;
  undefined auStack88 [8];

  *puParm3 = 0x50;
  if ((*(longlong **)(lParm1 + 0xa8) == (longlong *)0x0) || (**(longlong **)(lParm1 + 0xa8) == 0)) {
    uVar5 = 0;
    goto LAB_0065a618;
  }
  FUN_007bb428();
  lVar7 = *(longlong *)(extraout_x9 + 0x68);
  memset(auStack312,0,0xe8);
  FUN_007be464(auStack312);
  iVar1 = FUN_0063af64();
  if ((iVar1 == 0) || (iVar1 = FUN_0061a464(auStack88,0,*unaff_x21), iVar1 == 0)) {
LAB_0065a600:
    FUN_007be438();
    FUN_0061542c();
    uVar5 = 0;
  }
  else {
    FUN_007be458(auStack312);
    iVar1 = FUN_0063cce8();
    if (iVar1 == 0) goto LAB_0065a600;
    uVar6 = *(ulonglong *)(local_118 + 0x10);
```

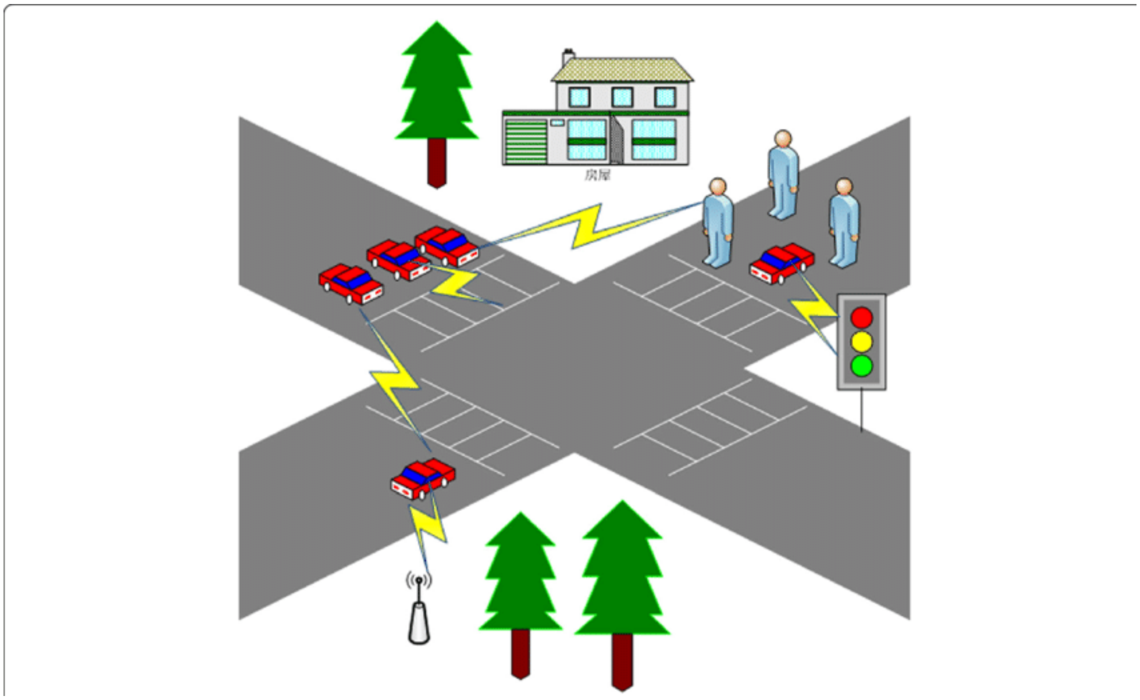


```
006873cc fe 4f c1 a8     ldp     x30,x19,[sp], #0x10
006873d0 c0 03 5f d6     ret

                     **************************************************************
                     *                          FUNCTION                          *
                     **************************************************************
                     undefined FUN_006873d4()
       undefined       w0:1          <RETURN>
                     FUN_006873d4                               XREF[3]:    002a859c, 00339880(*),
                                                                            008630e8(*)
006873d4 ff 03 05 d1     sub     sp,sp,#0x140
006873d8 fd 7b 0f a9     stp     x29,x30,[sp, #0xf0]
006873dc bc de 05 94     bl      FUN_007feecc                    undefined FUN_007feecc(undefine
006873e0 08 0a 80 52     mov     w8,#0x50
006873e4 48 00 00 39     strb    w8,[x2]
006873e8 16 54 40 f9     ldr     x22,[x0, #0xa8]
006873ec 76 07 00 b4     cbz     x22,LAB_006874d8
006873f0 c8 02 40 f9     ldr     x8,[x22]
006873f4 28 07 00 b4     cbz     x8,LAB_006874d8
006873f8 29 20 40 a9     ldp     x9,x8,[x1]
006873fc f3 03 02 aa     mov     x19,x2
00687400 7c e7 05 94     bl      FUN_008011f0                    undefined FUN_008011f0()
00687404 08 11 40 f9     ldr     x8,[x8, #0x20]
00687408 39 35 40 f9     ldr     x25,[x9, #0x68]
0068740c c9 06 40 f9     ldr     x9,[x22, #0x8]
00687410 e0 23 00 91     add     x0,sp,#0x8
```

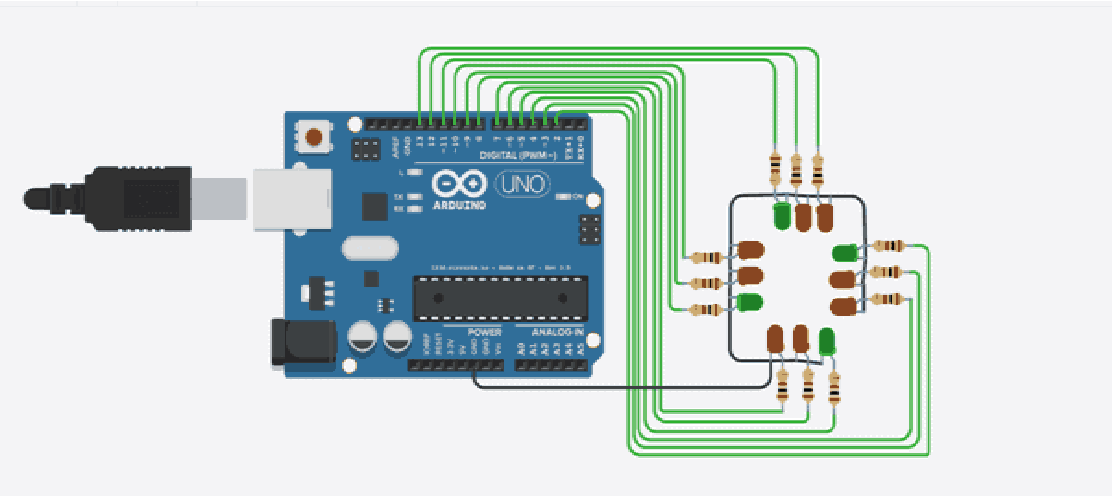
Diagram:



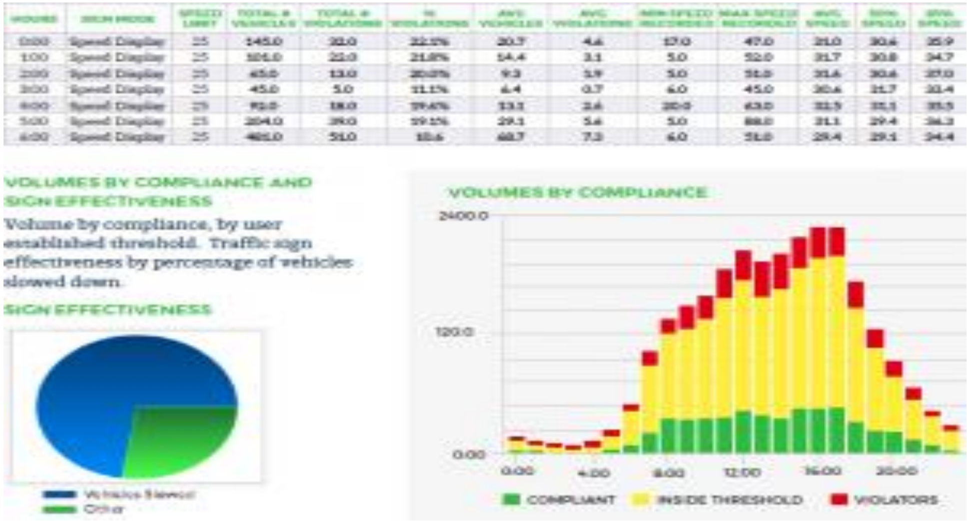
Schematic:



IOT-device:



Data sharing:



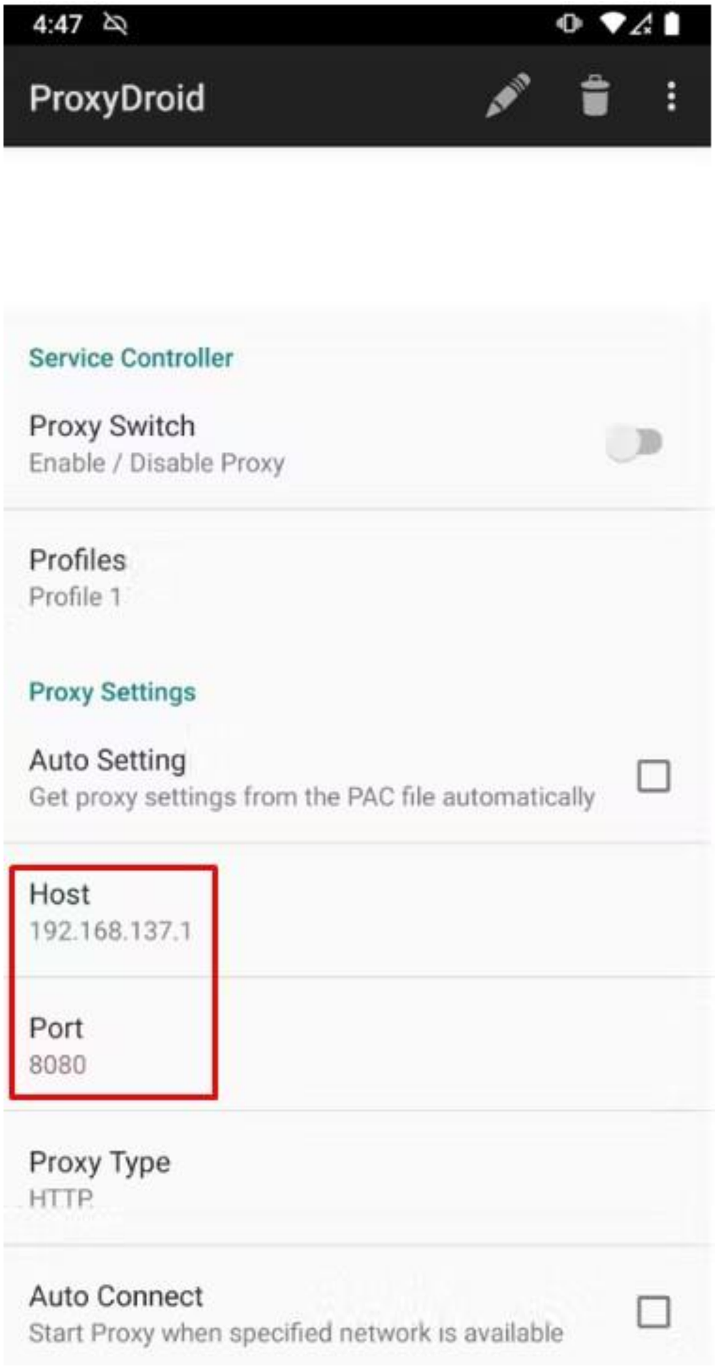
HOURS	SIGN MODE	SPEED LIMIT	TOTAL # VEHICLES	TOTAL # VIOLATIONS	% VIOLATIONS	AVG VEHICLES	AVG VIOLATIONS	MIN SPEED RECORDED	MAX SPEED RECORDED	AVG SPEED	50% SPEED	85% SPEED
0:00	Speed Display	25	145.0	32.0	22.1%	20.7	4.6	17.0	47.0	31.0	30.6	35.9
1:00	Speed Display	25	101.0	22.0	21.8%	14.4	3.1	5.0	52.0	31.7	30.8	34.7
2:00	Speed Display	25	65.0	13.0	20.0%	9.3	1.9	5.0	51.0	31.6	30.6	37.0
3:00	Speed Display	25	45.0	5.0	11.1%	6.4	0.7	6.0	45.0	30.6	31.7	33.4
4:00	Speed Display	25	92.0	18.0	19.6%	13.1	2.6	20.0	63.0	32.5	35.1	35.5
5:00	Speed Display	25	204.0	39.0	19.1%	29.1	5.6	5.0	88.0	31.1	29.4	34.3
6:00	Speed Display	25	481.0	51.0	10.6	68.7	7.3	6.0	51.0	29.4	29.1	34.4

VOLUMES BY COMPLIANCE AND SIGN EFFECTIVENESS

Volume by compliance, by user established threshold. Traffic sign effectiveness by percentage of vehicles slowed down.

SIGN EFFECTIVENESS

Process in mobile app with screenshot:



4:47

ProxyDroid

Service Controller

Proxy Switch
Enable / Disable Proxy

Profiles
Profile 1

Proxy Settings

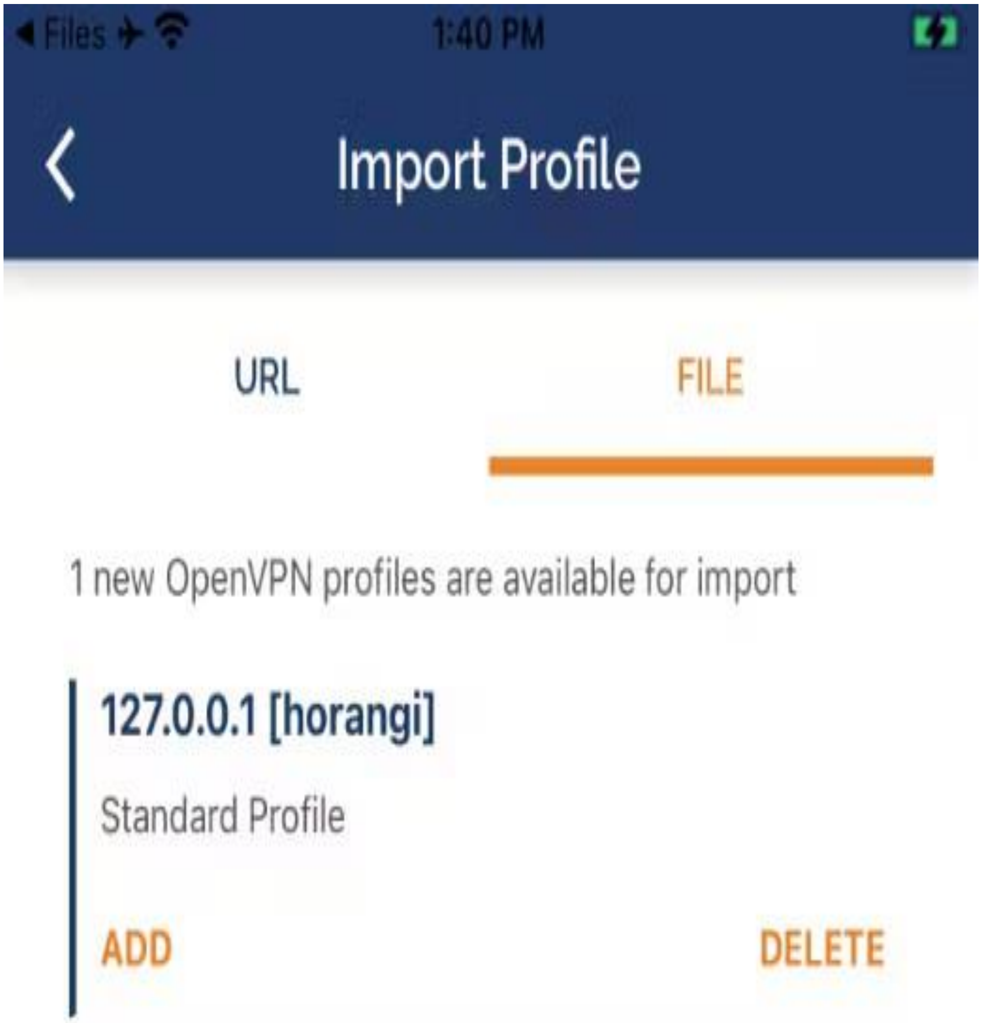
Auto Setting □
Get proxy settings from the PAC file automatically

Host
192.168.137.1

Port
8080

Proxy Type
HTTP

Auto Connect □
Start Proxy when specified network is available

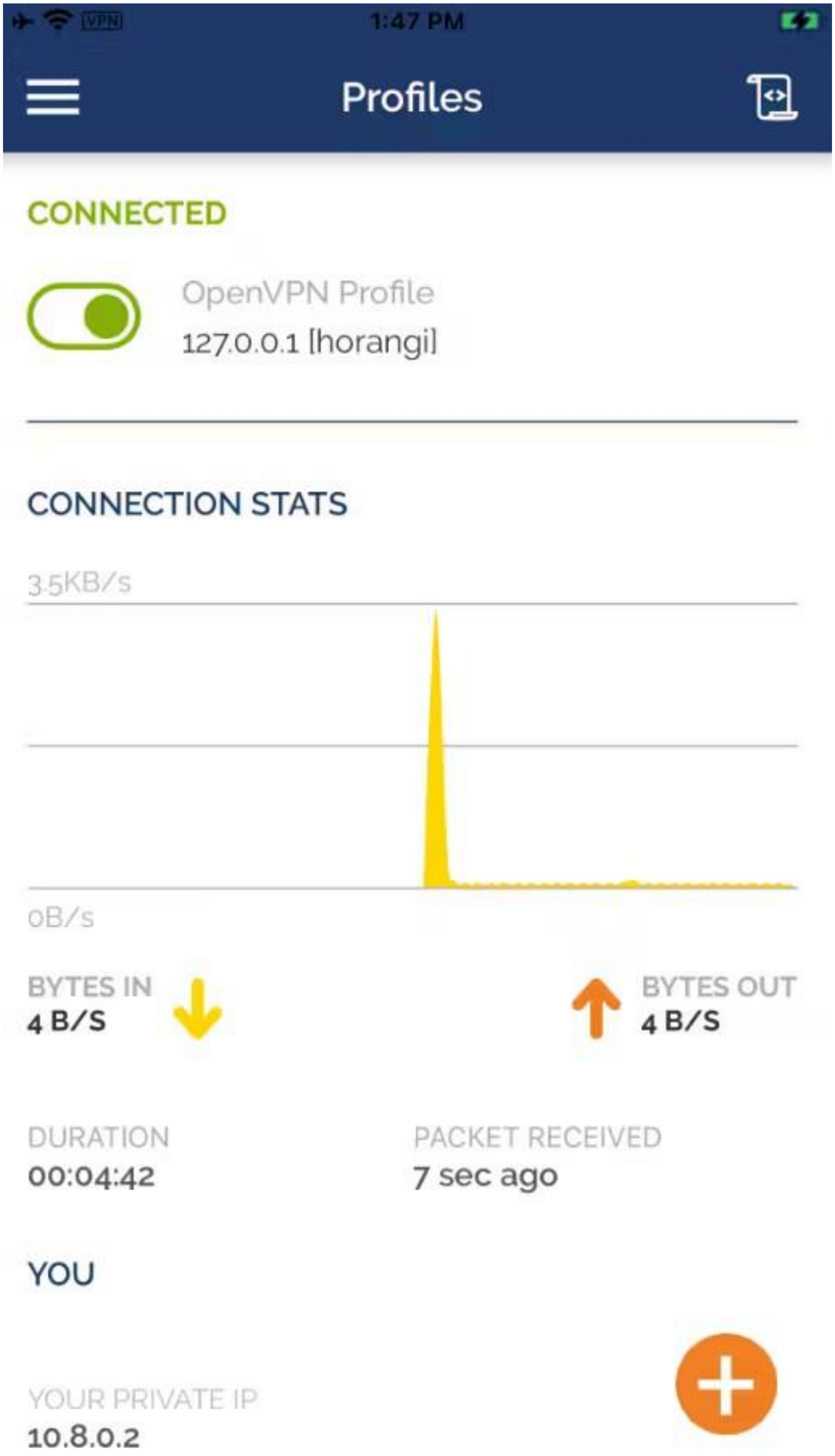


Import Profile

1 new OpenVPN profiles are available for import

127.0.0.1 [horangi]

Standard Profile



CONNECTED

OpenVPN Profile
127.0.0.1 [horangi]

CONNECTION STATS

BYTES IN
4 B/S

BYTES OUT
4 B/S

DURATION
00:04:42

PACKET RECEIVED
7 sec ago

YOU

YOUR PRIVATE IP
10.8.0.2

Benefits of traffic management:

Cleaner, greener, safer, and more accessible roads are a few benefits of implementing IoT and intelligent technology.

It helps with the following:

- Reducing traffic jams and accidents on the streets
- Ensuring immediate clearance for emergency vehicles
- Facilitating safer and shorter commute times
- Reducing congestion & energy consumption at intersections
- Offering significant productivity benefits with real-time monitoring of crucial infrastructures
- Reducing operating costs with efficient traffic management processes
- Ensuring compliance with the regulations for reducing the carbon footprint
- Saving billions of gallons of fuel wasted every year

Conclusion:

Physical infrastructure extension is an available solution for traffic management in many regions of the world. But the task is time-consuming, costly, complex, and disturbing work. Employing technology as a key monitoring and management tool to overcome obstacles is a vice decision over physical infrastructure destruction. The smart traffic management system can provide intelligent insights and solutions for current problems through a few modifications and tech integrations. AI is evolving as a prominent tool in every sector. And the future traffic management system for the smart community would be an intelligent traffic management system. If you resonate with our article please share your thoughts with us.