VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

J- COMPONENT (REVIEW 2)

Information Security Analysis and Audit

TITLE-DATA SECURE HOME AUTOMATION SYSTEM

PRESENTED BY:

Gunjan Kumar – 18BIT0070

SLOT – G1

TO:

SUMAIYA THASEEN I

i) Design of the system and Description (individual for every team member) – 10 marks
5 marks –diagram
5 marks – explanation

Ans: DESIGN:

Entire model :

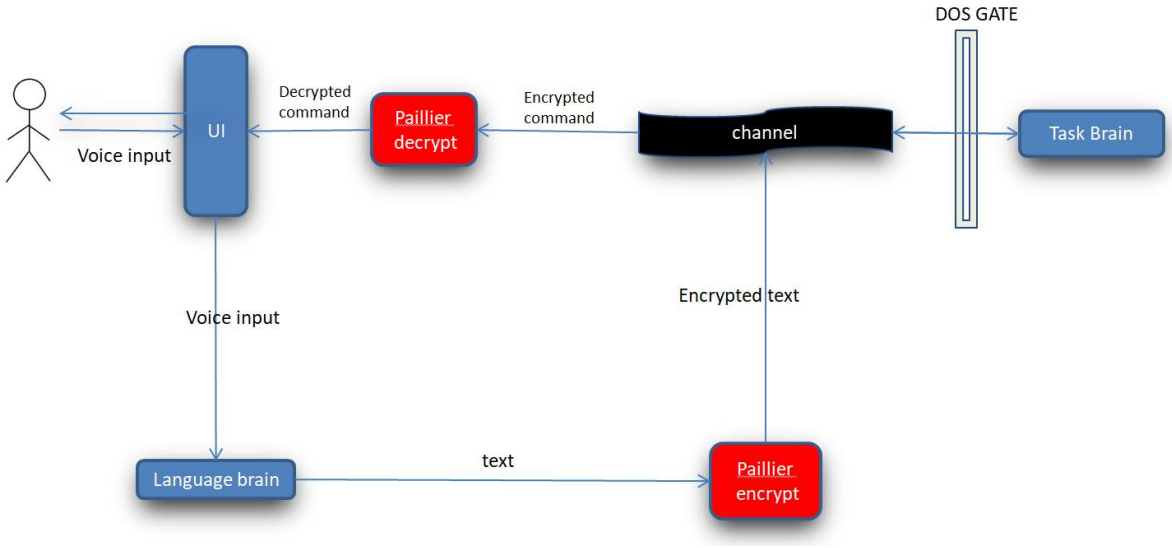


Figure 1: Entire Model

a) Langauge brain:

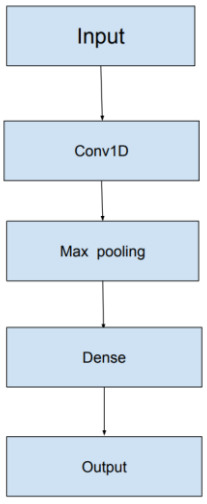


Figure 2: Model

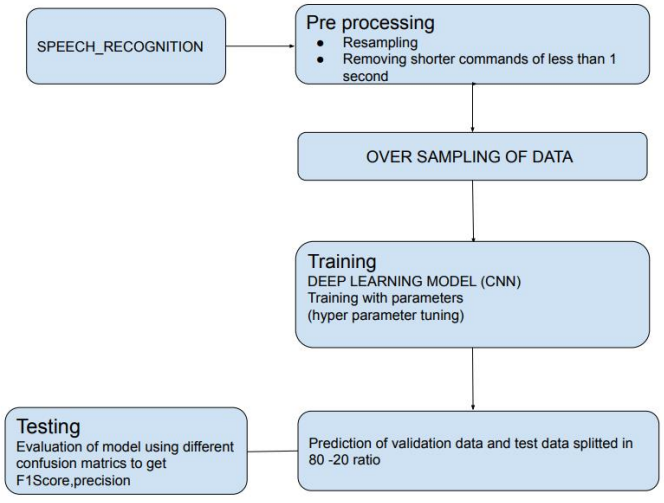


Figure 3: Architecture

b) Encryption

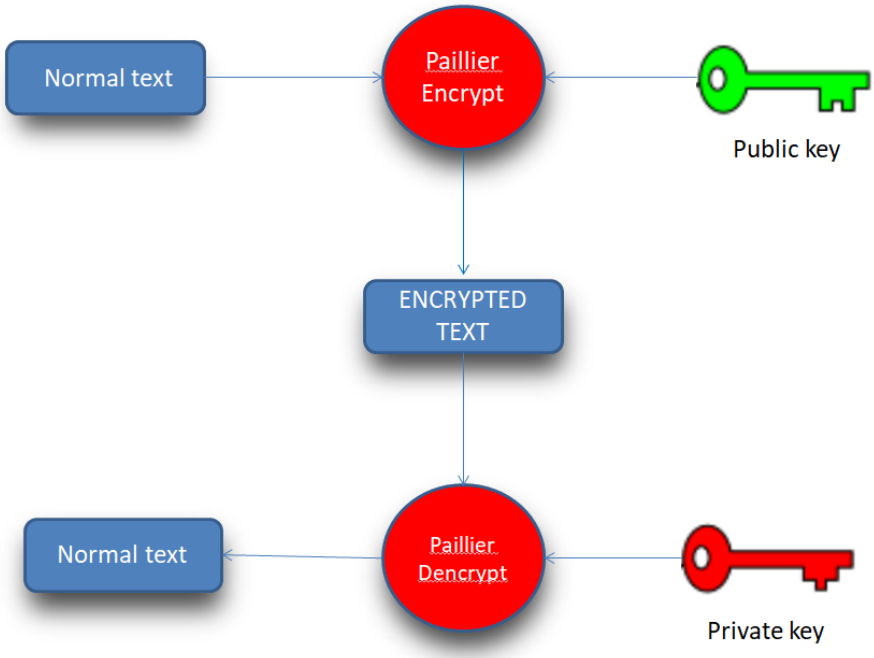


Figure 4: Architecture

c) Task Brain:

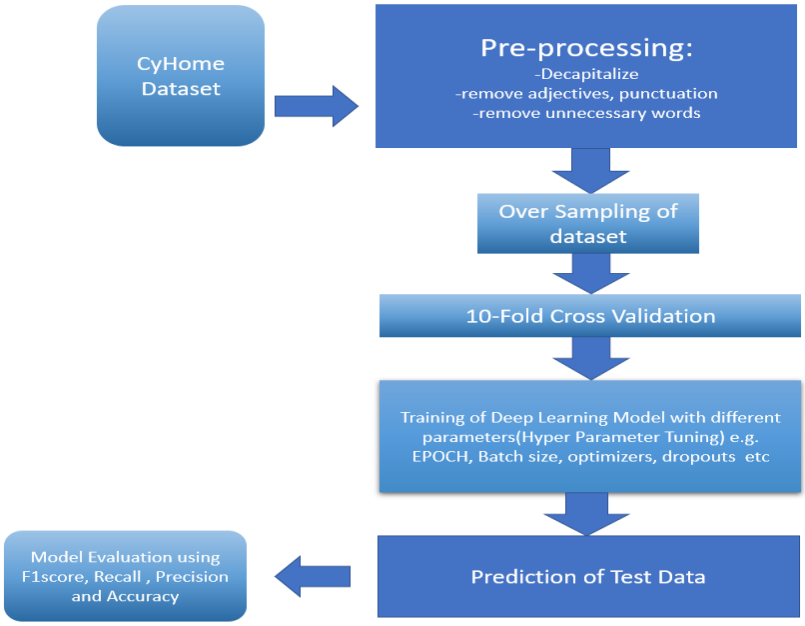


Figure 5: Architecture

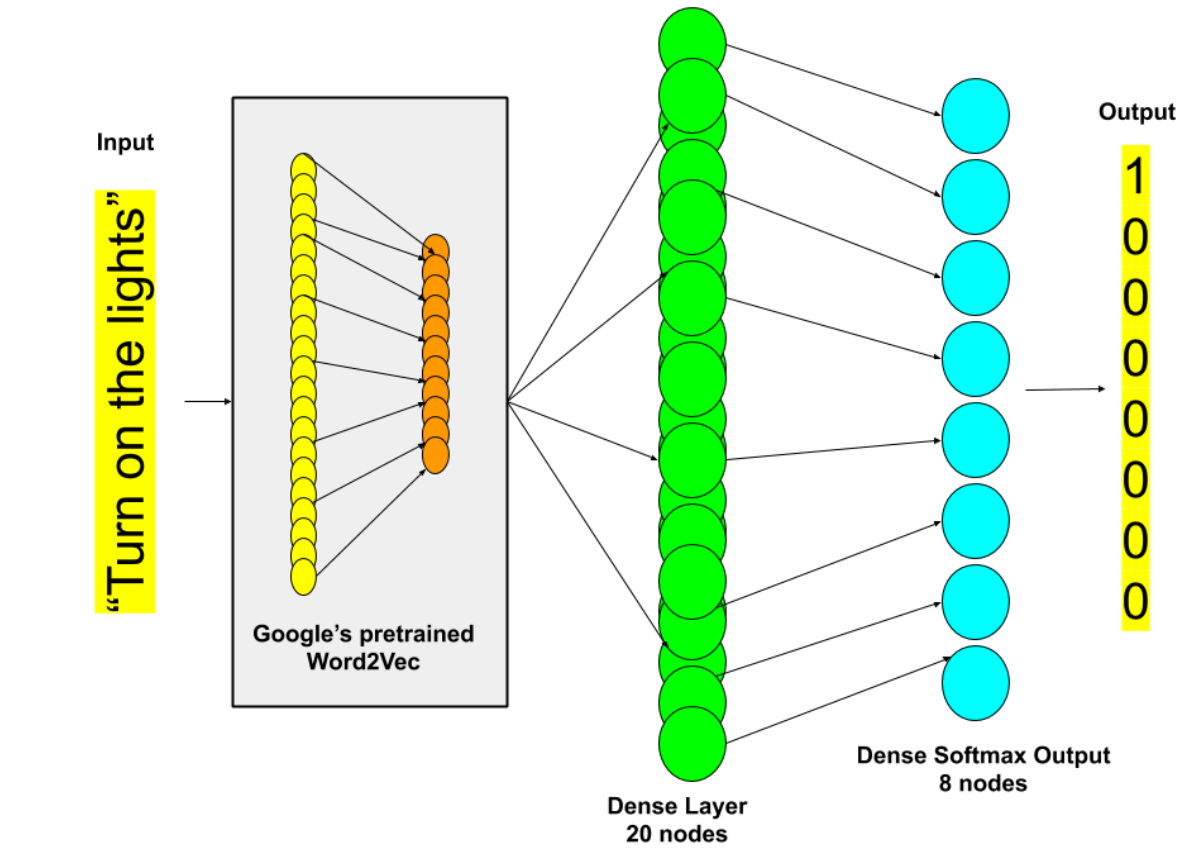


Figure 6: Deep learning model

d) DOS :

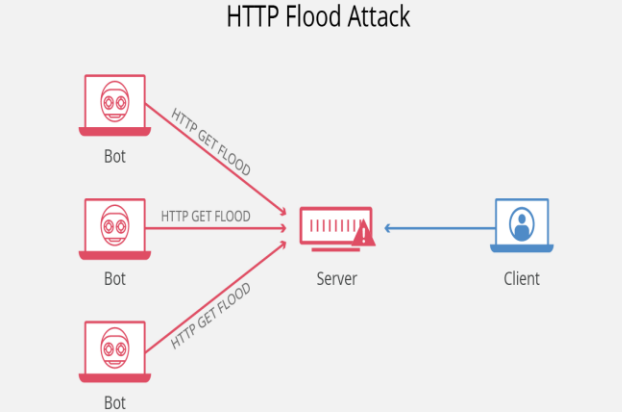


Figure 7: Http Flood Attack

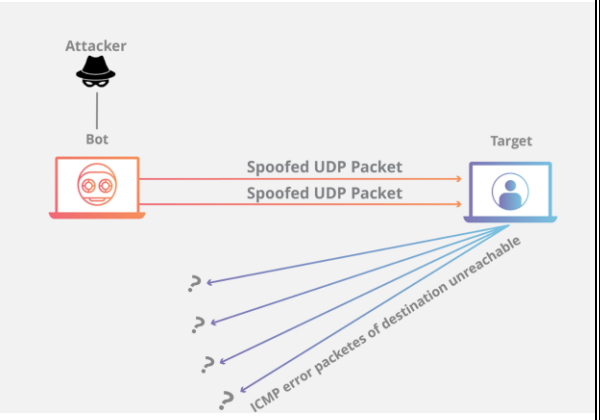


Figure 8: UDP Flood attack

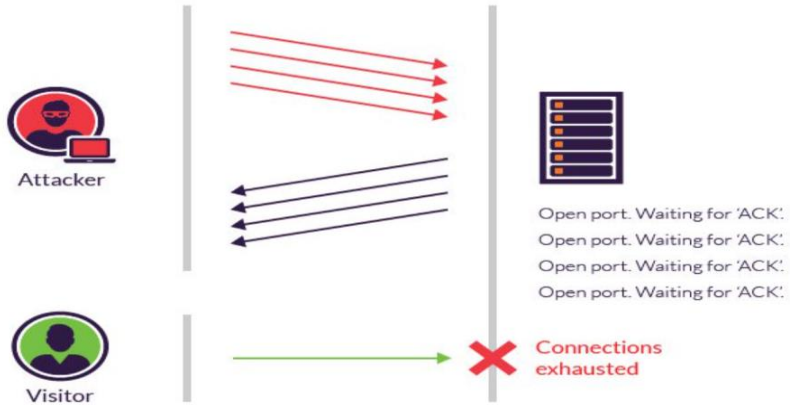


Figure 9: SYN flood attack

DESCRIPTION:

1. Language Brain:

- The user feeds in his voice in the model.
- Preprocessing is done on the voice input in two steps :
    1. Resampling
    2. Removing shorter commands of less than 1 second
- Oversampling of data is done to get 70 percent balanced data
- Training of CNN model is done with different hyper tuning parameters and evaluated many times with different parameters to get better accuracy.
- After that prediction is done with many different parameter and best model is selected
- Last the model is tested and generation of confusion matrics are done to evaluate the model.
- The converted text from the voice input is obtained as the output.

2.Encryption and Task brain:

- The text is then then preprocessed at the client side to convert it to a vector of length 186 by:
    - Decapitalizing each sentence
    - Removing Adjectives and punctuations
    - Removing unnecessary words

- This vector of integers is then encrypted using public key and sent to the server where the task brain resides.
- After this the pre-processed encrypted dataset is oversampled to to get minimum of 70% balanced dataset.
- Then 10-Fold cross validation is used to get max accuracy with a fixed parameter.
- Also parameters are changed (Several times)and 10 fold cross validation is done again.
- At last the best model among all the models is taken and testing is done in that model.
- Finally, to evaluate the model F1-score, Precision, Recall and Accuracy along with confusion matrix is used.
- The trained model processes the input encrypted vector and returns an array of length 8.
- This new array is sent back to the client side.
- It is decrypted using private key and then the values are compared with a predefined dictionary of commands.
- The command with the highest probability is executed in the client side.

3) DOS attack prevention:

- We will be preventing some of the DOS attack on the server where our task brain is located so that the channel between the client and the server is secure.

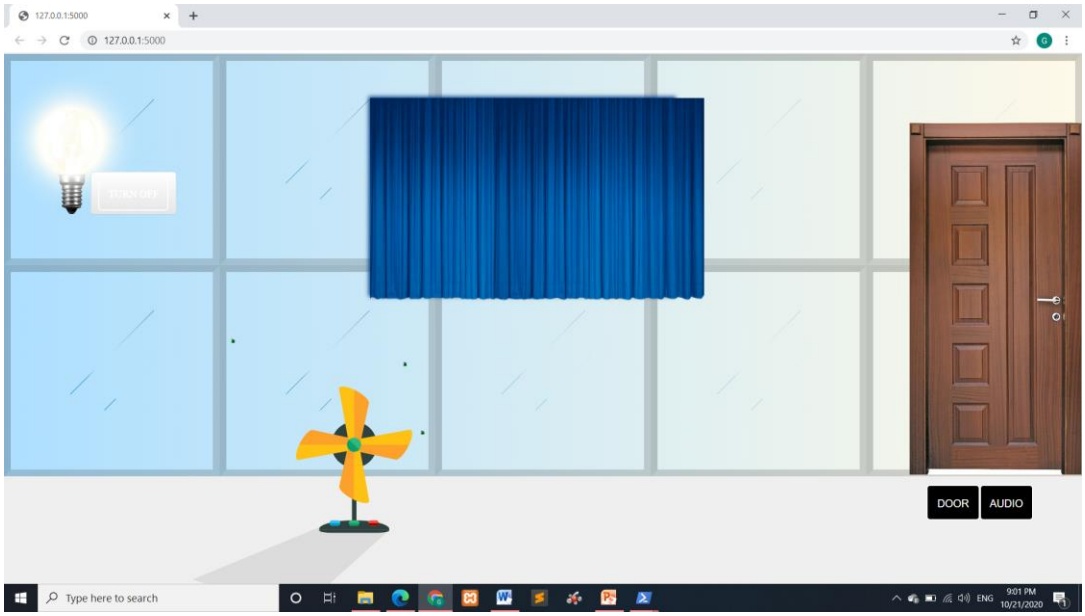
ii) Application developed (localhost- website – username and password- banking/cloud) - 15 marks

Ans:

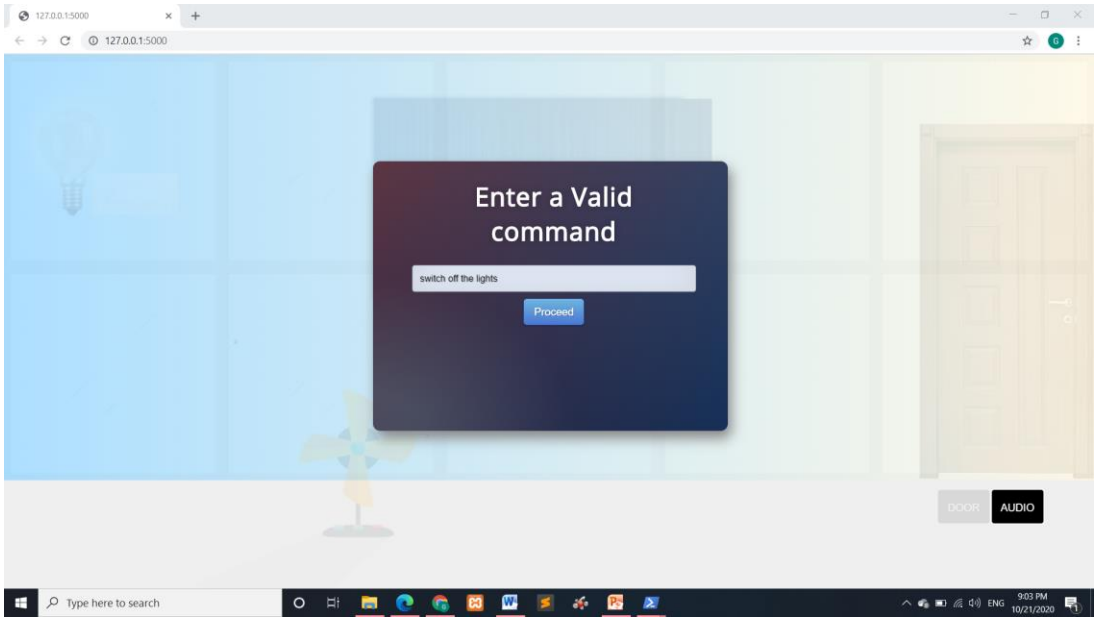
Website is developed on Flask and runs on localhost:5000

FRONTEND :

- Tools used :- HTML, CSS, javascript



- **AUDIO** button allows user to enter text command (in final review it will allow user to give voice input). This text command will be encrypted and send to the server where task brain is located.



BACKEND:

- Tools used : **Flask and python.**
- Runs on **localhost:5000**

```
Windows PowerShell
PS E:\Project_fall_sem\Nascom_v3_review2_final> python client.py              2020-10-21
 21:11:40.779268: W tensorflow/stream_executor/platform/default/dso_loader.cc:59] Could
not load dynamic library 'cudart64_101.dll'; dlerror: cudart64_101.dll not found
2020-10-21 21:11:40.789639: I tensorflow/stream_executor/cuda/cudart_stub.cc:29] Ignore
above cudart dlerror if you do not have a GPU set up on your machine.
P and Q are not independent of each other
CyHome's Client Package successfully imported
 * Serving Flask app "client" (lazy loading)
 * 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
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
127.0.0.1 - - [21/Oct/2020 21:11:46] "[37mGET / HTTP/1.1[0m" 200 -
```

iii) Key generation / Hashing using 'X' algorithm - 10 marks

- Key is generated as well as Input string is converted into vector of length 186 using preprocessing.

```
113
114  def print_keys():
115      print()
116      print("================================================")
117      print("Public key (n,g):\t\t",n,g)
118      print("Private key (lambda,mu):\t",gLambda,gMu)
119      print("================================================")
120
```

```
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
127.0.0.1 - - [21/Oct/2020 21:11:46] "[37mGET / HTTP/1.1[0m" 200 -
================================================
input string:  switch off the lights
================================================
================================================

input_vector:  [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0
, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0,
1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0
, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0]
================================================

================================================
Public key (n,g):                4087 2
Private key (lambda,mu):         660 3831
================================================
```

iv) Encryption using ‘Y’ algorithm - 15 marks

Or

Digital signature generation – 15 marks

Ans:

The vector is encrypted and the new encrypted vector is generated.

CODE:

```
70
71  def Pallier_encrpyt(m):    #THE MESSAGE-NUMBER TO BE ENCRYPTED
72
73      #MESSAGE ENCRYPTION: converting m variable to cipher variable
74      r=randint(1,n)
75      if(r%p==0):
76          r-=1
77      if(r%q==0):
78          r-=1
79      k1 = (pow(g, int(m), n*n))%(n*n)
80      k2 = pow(r, n, n*n)
81      cipher = ((k1% (n*n))*(k2% (n*n)))%(n*n)
82      return cipher
83
```

```
22  #######   step 1   #######
23  p=61 #CONSTANT
24  q=67 #CONSTANT , both constant should be independent of each other
25  n = p*q
26  #function definition
27  def gcd(a,b):
28      while b > 0:
29          a, b = b, a % b
30      return a
31
32  def  lcm(a, b):
33      return a * b // gcd(a, b)
34
35  def L(x,n):
36      return ((x-1)//n)
37
38  #condition checking
39  if (p==q):
40      print("P and Q cannot be the same")
41
42
43  if (gcd(p*q,(p-1)*(q-1))==1):
44      print("P and Q are not independent of each other")
45
46  ########   step 2   #############
47  gLambda = lcm(p-1,q-1)
48  ########   step 3   #############
49  g=2 #select any random integer
50  r=1
51  ########   step 4   #############
52  l = (pow(g, gLambda, n*n)-1)//n
53  gMu = libnum.invmod(l, n)
```

OUTPUT:

The vector of 186 length is encrypted and required output is generated.

```
================================================
Public key (n,g):                4087 2
Private key (lambda,mu):         660 3831
================================================

encrypted vector  [6925897, 10612893, 9597339, 8259617, 4742215, 6127484, 5390211, 5757052, 1
5370335, 12627556, 16440060, 1934255, 9621269, 15517002, 13885934, 14227551, 6323282, 757423,
 15842578, 14683829, 8562519, 4651822, 1717986, 14499974, 11328242, 3319706, 3669507, 5007229
, 6014139, 7823823, 13202709, 2008577, 13708846, 12371827, 1826393, 11238183, 14414489, 55966
8, 10319336, 8570802, 192467, 9707086, 5872312, 1024793, 14018447, 633649, 8132990, 2135236,
10419473, 11033263, 13002012, 14377370, 6622617, 5345524, 8700130, 6239853, 3228712, 16281036
, 1939878, 6153852, 2023461, 7258153, 15662625, 412358, 15731452, 2830565, 2131197, 6384233,
4609082, 6296868, 10214810, 12797635, 10060746, 12318538, 6750559, 635528, 4602303, 10145139,
 10783026, 10860362, 7471167, 2531517, 1586485, 5260829, 14425014, 8554315, 4655403, 15119456
, 15543497, 15540414, 16274621, 4830794, 4432474, 12605812, 7078278, 16394725, 887701, 370189
7, 15674370, 5593742, 15823973, 8972169, 5241953, 99388, 13382269, 12141624, 11844561, 946147
0, 13358389, 2594138, 13371834, 10979646, 754600, 8443952, 3438822, 15744224, 4602303, 414891
4, 13424605, 959345, 4012856, 8846477, 12205312, 3457427, 8202524, 11614121, 404907, 11109827
, 13606859, 4065816, 11151573, 7180929, 2691943, 4510885, 9141472, 2391938, 2635547, 2923459,
 3391349, 117835, 12540435, 210479, 4859008, 7694456, 4082369, 3190558, 13868431, 3500698, 14
056277, 14646529, 29150, 7014965, 559668, 10060746, 6393163, 15904670, 1586485, 8561420, 4120
180, 10434709, 5872312, 8066103, 13377668, 10110529, 13972956, 14161453, 209455, 7495755, 221
8288, 7733076, 8967112, 7261731, 2044489, 11941307, 9582147, 6312256, 12051887, 14018447, 956
4588, 7258434, 4871462, 15649853, 13008044, 14642808, 9530146, 668162, 2088506, 11335004, 749
5755, 15775803, 8963315, 12812166, 412358, 16121856, 8479163, 10532091, 2899274, 8650751, 137
95915, 12693771, 12704490, 12016549, 2309978, 3243679, 16694024, 676621, 1766450, 12141624, 1
0903533, 13873004, 14646529, 15645466, 4219350, 14366018, 8780988, 11110905, 293695, 12270305
, 3940041, 28930, 16160904, 6877848, 9016247, 444859, 10758414, 14338566, 4559105, 15649853,
2308045, 9445416, 4778789, 5046358, 8012866, 1776945, 11832107, 10011771, 7625354, 2765728, 1
5733070, 7617385, 9398748, 13914372, 10423457, 4895283, 9296718, 9840966, 14503960, 10927697,
 10179974, 15716886, 1459204, 9151600, 10222350, 7908743, 14159006, 10863659, 10685019, 86802
55, 13732551, 635528, 15378013, 6204246, 1698836, 723177, 3321550, 6237078, 15026514, 1262741
4, 12347938, 16142196, 15842578, 14487913, 1023940, 1092737, 6579346, 9451339, 12659680, 3673
76, 15365125, 11141699, 4899218, 13919689, 12459936, 10744045, 12410614, 12501722, 9451339, 1
3661484, 6784221, 11024705, 12619027, 4626440, 11739183, 10110529, 12948144, 12336051, 270051
0, 11461616, 14407111, 8870263, 7471167, 10937358, 1663551, 14503455, 1919435, 4325242, 55966
8, 13095135, 13159838, 1351756, 7823823, 1234754, 6172376, 4782086, 3894361, 4924576, 8046038
, 1545294, 9151600, 11427873, 6135264, 4938530, 888821, 14245428, 10406701, 1380755, 13019178
, 13013890, 4032990, 2359686, 633649, 6306597, 7251731, 6867448, 4986404, 11024705, 11569352,
 9264792, 13202871, 16671698, 5893490, 5551996, 1202548, 12663326, 9863032, 1933302, 12540795
, 8492940, 7924427, 1187431, 4865628, 16054999, 10377868, 5509722, 11335004, 16626946, 135707
51, 7042737, 12884130, 12230309, 7156165, 3474796, 1573485, 8486364, 15072534, 6691798, 52344
22, 5114272, 9344104, 534821, 6362909, 4103645, 13758155, 16258710, 10583761, 15785155, 13679
061, 6252898, 13236533, 14823799, 888821, 3042085, 7282877]
```

Digital signature