PRACTICAL NO : 01

AIM : Write Python Program to convert from CSV formats to HORUS format.

CODE:

```
import pandas as pd
sInputFileName='Country_Code.csv'
InputData=pd.read_csv(sInputFileName,encoding="latin-1")
print('Input Data Values ===================================')
print(InputData)
print('=====================================================')
ProcessData=InputData
ProcessData.drop('ISO-2-CODE', axis=1,inplace=True)
ProcessData.drop('ISO-3-Code', axis=1,inplace=True)
ProcessData.rename(columns={'Country': 'CountryName'}, inplace=True)
ProcessData.rename(columns={'ISO-M49': 'CountryNumber'}, inplace=True)
ProcessData.set_index('CountryNumber', inplace=True)
ProcessData.sort_values('CountryName', axis=0, ascending=False, inplace=True)
print('Process Data Values ================================')
print(ProcessData)
print('=====================================================')
OutputData=ProcessData
sOutputFileName='HORUS-CSV-Country.csv'
OutputData.to_csv(sOutputFileName, index = False)
print('CSV to HORUS - Done')
```

OUTPUT :

```
horus.py
Input Data Values ==================================
                      Country ISO-2-CODE ISO-3-Code  ISO-M49
0                 Afghanistan         AF        AFG        4
1               Aland Islands         AX        ALA      248
2                     Albania         AL        ALB        8
3                     Algeria         DZ        DZA       12
4              American Samoa         AS        ASM       16
..                       ...        ...        ...      ...
242  Wallis and Futuna Islands         WF        WLF      876
243            Western Sahara         EH        ESH      732
244                     Yemen         YE        YEM      887
245                    Zambia         ZM        ZMB      894
246                  Zimbabwe         ZW        ZWE      716

[247 rows x 4 columns]
====================================================
Process Data Values ================================
                           CountryName
CountryNumber
716                           Zimbabwe
894                             Zambia
887                              Yemen
732                      Western Sahara
876            Wallis and Futuna Islands
...                                  ...
16                       American Samoa
12                              Algeria
8                               Albania
248                       Aland Islands
4                           Afghanistan

[247 rows x 1 columns]
====================================================
CSV to HORUS - Done
```

PRACTICAL NO : 02

AIM : Write Python Program to convert from XML formats to HORUS format.

CODE:

```
import pandas as pd
import xml.etree.ElementTree as ET
def xml2df(xml_data):
    root = ET.XML(xml_data)
    all_records = []
    for i, child in enumerate(root):
        record = {}
        for subchild in child:
            record[subchild.tag] = subchild.text
        all_records.append(record)
    return pd.DataFrame(all_records)
sInputFileName='Country_Code.xml'
InputData = open(sInputFileName).read()
print('=================================================')
print('Input Data Values ===================================')
print('=================================================')
print(InputData)
print('=================================================')
ProcessDataXML=InputData
ProcessData=xml2df(ProcessDataXML)
ProcessData.drop('ISO-2-CODE', axis=1,inplace=True)
ProcessData.drop('ISO-3-Code', axis=1,inplace=True)
ProcessData.rename(columns={'Country': 'CountryName'}, inplace=True)
ProcessData.rename(columns={'ISO-M49': 'CountryNumber'}, inplace=True)
ProcessData.set_index('CountryNumber', inplace=True)
ProcessData.sort_values('CountryName', axis=0, ascending=False, inplace=True)
print('=================================================')
print('Process Data Values ================================')
```

```
print('=====================================================')
print(ProcessData)
print('=====================================================')
OutputData=ProcessData
sOutputFileName='C:/VKHCG/05-DS/9999-Data/HORUS-XML-Country.csv'
OutputData.to_csv(sOutputFileName, index = False)
print('=====================================================')
print('XML to HORUS - Done')
print('=====================================================')
```

OUTPUT :

```
2 xml to horus.py
=====================================================
Input Data Values ==================================
=====================================================
Squeezed text (707 lines).
=====================================================
=====================================================
Process Data Values ================================
=====================================================
                             CountryName
CountryNumber
716                             Zimbabwe
894                               Zambia
887                                Yemen
732                       Western Sahara
876            Wallis and Futuna Islands
...                                  ...
16                        American Samoa
12                               Algeria
8                                Albania
248                        Aland Islands
4                            Afghanistan

[247 rows x 1 columns]
=====================================================
=====================================================
XML to HORUS - Done
=====================================================
```

PRACTICAL NO : 03

AIM : Write Python Program to convert from Picture (JPEG) formats to HORUS format.

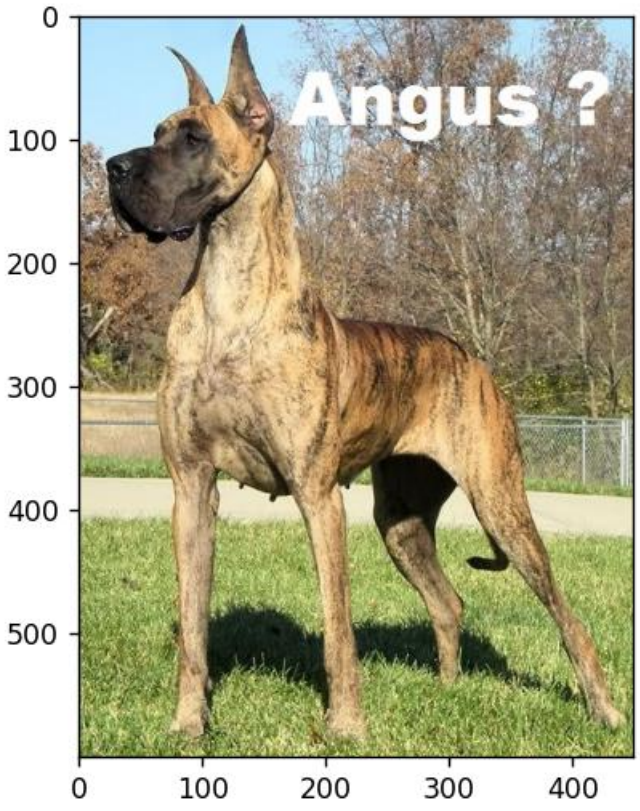
CODE:

```
import imageio
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
sInputFileName='Angus.jpg'
InputData = imageio.imread(sInputFileName)
print('Input Data Values ===================================')
print('X: ',InputData.shape[0])
print('Y: ',InputData.shape[1])
print('RGBA: ', InputData.shape[2])
print('=====================================================')
ProcessRawData=InputData.flatten()
y=InputData.shape[2] + 2
x=int(ProcessRawData.shape[0]/y)
ProcessData=pd.DataFrame(np.reshape(ProcessRawData, (x, y)))
sColumns= ['XAxis','YAxis','Red', 'Green', 'Blue']
ProcessData.columns=sColumns
ProcessData.index.names =['ID']
print('Rows: ',ProcessData.shape[0])
print('Columns :',ProcessData.shape[1])
print('=====================================================')
print('Process Data Values ================================')
print('=====================================================')
plt.imshow(InputData)
plt.show()
print('=====================================================')
OutputData=ProcessData
print('Storing File')
```

```
sOutputFileName='HORUS-Picture.csv'

OutputData.to_csv(sOutputFileName, index = False)

print('=====================================================')

print('Picture to HORUS - Done')

print('=====================================================')
```

OUTPUT:

```
Input Data Values ==================================
X:  600
Y:  450
RGBA:  3
=====================================================
Rows:  162000
Columns : 5
=====================================================
Process Data Values ================================
=====================================================
=====================================================
Storing File
=====================================================
Picture to HORUS - Done
=====================================================
```



PRACTICAL NO : 04

AIM : Write Python Program to convert from MYSQL DATABASE formats to HORUS format.

CODE:

```
import pandas as pd
import sqlite3 as sq
sInputFileName='utility.db'
sInputTable='Country_Code'
conn = sq.connect(sInputFileName)
sSQL='select * FROM ' + sInputTable + ';'
InputData=pd.read_sql_query(sSQL, conn)
print('Input Data Values ===================================')
print(InputData)
print('=====================================================')
ProcessData=InputData
ProcessData.drop('ISO-2-CODE', axis=1,inplace=True)
ProcessData.drop('ISO-3-Code', axis=1,inplace=True)
ProcessData.rename(columns={'Country': 'CountryName'}, inplace=True)
ProcessData.rename(columns={'ISO-M49': 'CountryNumber'}, inplace=True)
ProcessData.set_index('CountryNumber', inplace=True)
ProcessData.sort_values('CountryName', axis=0, ascending=False, inplace=True)
print('Process Data Values ================================')
print(ProcessData)
print('=====================================================')
OutputData=ProcessData
sOutputFileName='C:/VKHCG/05-DS/9999-Data/HORUS-CSV-Country.csv'
OutputData.to_csv(sOutputFileName, index = False)
print('Database to HORUS - Done')
```

OUTPUT:

```
4 mysqlto hr.py
Input Data Values ==================================
     index                    Country ISO-2-CODE ISO-3-Code  ISO-M49
0        0                Afghanistan         AF        AFG        4
1        1              Aland Islands         AX        ALA      248
2        2                    Albania         AL        ALB        8
3        3                    Algeria         DZ        DZA       12
4        4             American Samoa         AS        ASM       16
..     ...                        ...        ...        ...      ...
242    242  Wallis and Futuna Islands         WF        WLF      876
243    243             Western Sahara         EH        ESH      732
244    244                      Yemen         YE        YEM      887
245    245                     Zambia         ZM        ZMB      894
246    246                   Zimbabwe         ZW        ZWE      716

[247 rows x 5 columns]
====================================================
Process Data Values ================================
               index                CountryName
CountryNumber
716              246                   Zimbabwe
894              245                     Zambia
887              244                      Yemen
732              243             Western Sahara
876              242  Wallis and Futuna Islands
...              ...                        ...
16                 4             American Samoa
12                 3                    Algeria
8                  2                    Albania
248                1              Aland Islands
4                  0                Afghanistan

[247 rows x 2 columns]
====================================================
Database to HORUS - Done
```

PRACTICAL NO : 05

AIM : Write a program to demonstrate fixers utilities.

CODE:

```
print('#1 Removing leading or lagging spaces from a data entry');
baddata = "    Data Science with too many spaces is bad!!!    "
print('>',baddata,'<')
cleandata=baddata.strip()
print('>',cleandata,'<')
```

OUTPUT:

```
5 Utilities and Auditing.py
#1 Removing leading or lagging spaces from a data entry
>     Data Science with too many spaces is bad!!!     <
> Data Science with too many spaces is bad!!! <
```

PRACTICAL NO : 06

AIM : Removing nonprintable characters from a data.

CODE:

```
import string
import datetime as dt
#2 Removing nonprintable characters from a data entry
print('#2 Removing nonprintable characters from a data entry')
printable = set(string.printable)
baddata ="Data\x00 Science \x03 with\x02 funny \x41 characters is \x10bad!!!"
cleandata=''.join(filter(lambda x: x in string.printable,baddata))
print('Bad Data : ',baddata);
print('Clean Data : ',cleandata)
```

OUTPUT:

```
- RESTART: C:\Users\Vedika Suresh Wankhe\OneDrive\Desktop\MSC Par
6 Utilities and Auditing.py
#2 Removing nonprintable characters from a data entry
Bad Data :  Data
Clean Data :  Data Science  with funny A characters is bad!!!
```

PRACTICAL NO : 07

AIM : Reformatting data entry to match specific formatting criteria.

CODE:

```
import string
import datetime as dt
# 3 Reformatting data entry to match specific formatting criteria.
# Convert YYYY/MM/DD to DD Month YYYY
print('# 3 Reformatting data entry to match specific formatting criteria.')
baddate = dt.date(2022, 11, 30)
baddata=format(baddate,'%Y-%m-%d')
gooddate = dt.datetime.strptime(baddata,'%Y-%m-%d')
gooddata=format(gooddate,'%d %B %Y')
print('Bad Data : ',baddata)
print('Good Data : ',gooddata)
```

OUTPUT:

```
7.py
# 3 Reformatting data entry to match specific formatting criteria.
Bad Data :  2022-11-30
Good Data :  30 November 2022
```

PRACTICAL NO : 08

AIM : Data bining or bucketing

CODE:

```
import numpy as np
import matplotlib.mlab as mlab
import matplotlib.pyplot as plt
import scipy.stats as stats
np.random.seed(0)
# example data
mu = 90 # mean of distribution
sigma = 10 # standard deviation of distribution
x = mu + sigma * np.random.randn(5000)
num_bins = 25
fig, ax = plt.subplots()
# the histogram of the data
n, bins, patches = ax.hist(x, num_bins, density=1)
# add a 'best fit' line
y = stats.norm.pdf(bins, mu, sigma)
# mlab.normpdf(bins, mu, sigma)
ax.plot(bins, y, '--')
ax.set_xlabel('Example Data')
ax.set_ylabel('Probability density')
sTitle=r'Histogram '
ax.set_title(sTitle)
fig.tight_layout()
sPathFig='Histogram.png'
fig.savefig(sPathFig)
plt.show()
```

OUTPUT:

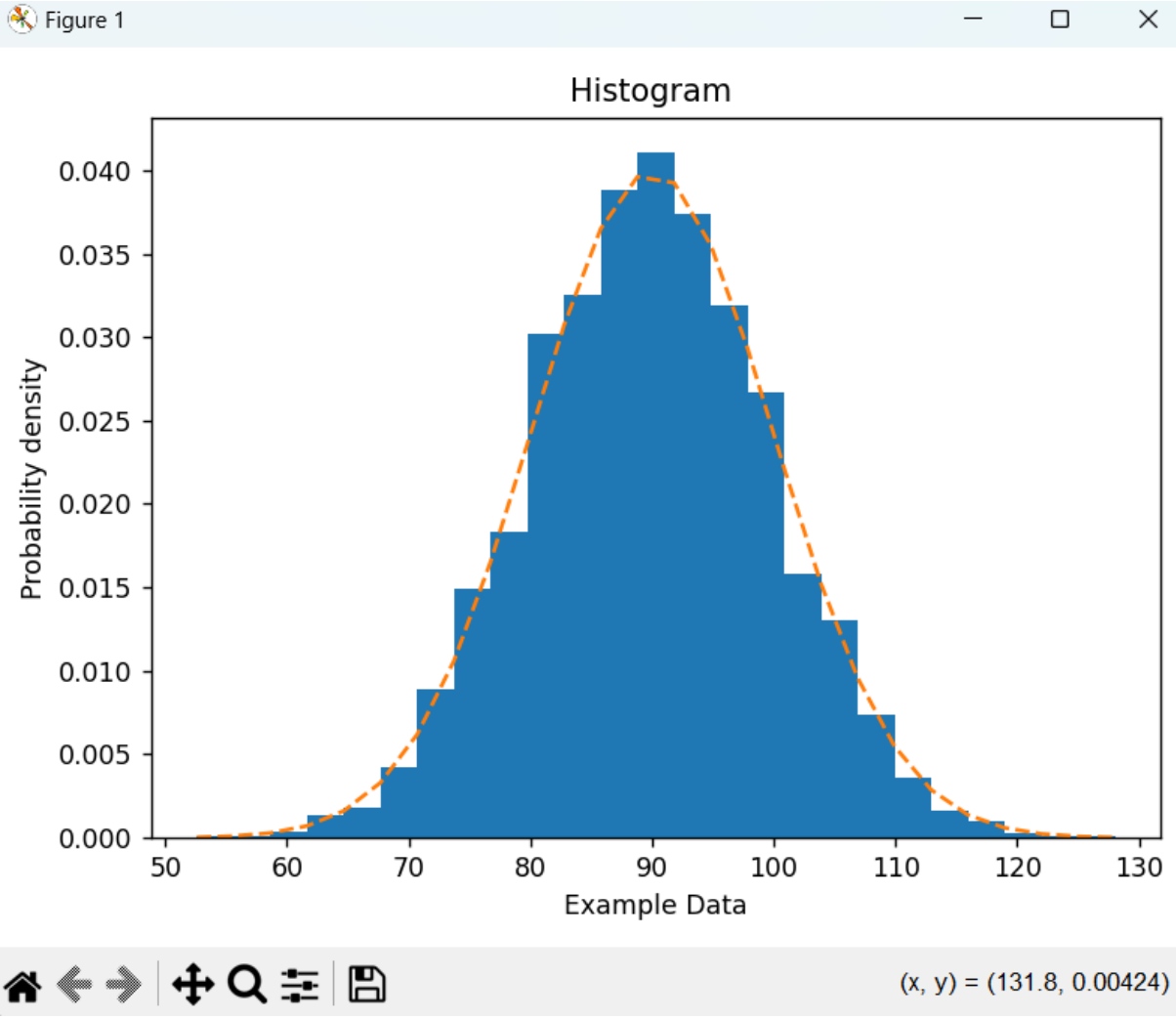


Figure 1
Histogram
Probability density
0.040
0.035
0.030
0.025
0.020
0.015
0.010
0.005
0.000
50
60
70
80
90
100
110
120
130
Example Data
(x, y) = (131.8, 0.00424)

PRACTICAL NO : 9A

AIM : Averaging of data

CODE:

```
import pandas as pd
sFileName='IP_DATA_CORE.csv'
IP_DATA_ALL=pd.read_csv(sFileName,header=0,low_memory=False,
usecols=['Country','Place Name','Latitude','Longitude'], encoding="latin-1")
IP_DATA_ALL.rename(columns={'Place Name': 'Place_Name'}, inplace=True)
AllData=IP_DATA_ALL[['Country', 'Place_Name','Latitude']]
print("Original Data: with Latitude")
print(AllData)
MeanData=AllData.groupby(['Country', 'Place_Name'])['Latitude'].mean()
print("\nMean Data: of Latitude")
print(MeanData)
```

OUTPUT:

```
9A.py
Original Data: with Latitude
     Country Place_Name  Latitude
0         US   New York   40.7528
1         US   New York   40.7528
2         US   New York   40.7528
3         US   New York   40.7528
4         US   New York   40.7528
...      ...        ...       ...
3557      DE     Munich   48.0915
3558      DE     Munich   48.1833
3559      DE     Munich   48.1000
3560      DE     Munich   48.1480
3561      DE     Munich   48.1480

[3562 rows x 3 columns]

Mean Data: of Latitude
Country  Place_Name
DE       Munich        48.143223
GB       London        51.509406
US       New York      40.747044
Name: Latitude, dtype: float64
```

PRACTICAL NO : 9B

AIM : Outlier

CODE:

```
import pandas as pd
sFileName='IP_DATA_CORE.csv'
print('Loading :',sFileName)
IP_DATA_ALL=pd.read_csv(sFileName,header=0,low_memory=False,
  usecols=['Country','Place Name','Latitude','Longitude'], encoding="latin-1")
IP_DATA_ALL.rename(columns={'Place Name': 'Place_Name'}, inplace=True)
LondonData=IP_DATA_ALL.loc[IP_DATA_ALL['Place_Name']=='London']
AllData=LondonData[['Country', 'Place_Name','Latitude']]
print('All Data')
print(AllData)
MeanData=AllData.groupby(['Country', 'Place_Name'])['Latitude'].mean()
StdData=AllData.groupby(['Country', 'Place_Name'])['Latitude'].std()
print('Outliers')
UpperBound=float(MeanData+StdData)
print('Higher than ', UpperBound)
OutliersHigher=AllData[AllData.Latitude>UpperBound]
print(OutliersHigher)
LowerBound=float(MeanData-StdData)
print('Lower than ', LowerBound)
OutliersLower=AllData[AllData.Latitude<LowerBound]
print(OutliersLower)
print('Not Outliers')
OutliersNot=AllData[(AllData.Latitude>=LowerBound) & (AllData.Latitude<=UpperBound)]
print(OutliersNot)
```

OUTPUT:

```
Lower than  51.506176875621264
     Country Place_Name  Latitude
1915      GB     London   51.4739
Not Outliers
     Country Place_Name  Latitude
1917      GB     London   51.5085
1918      GB     London   51.5085
1922      GB     London   51.5085
1928      GB     London   51.5085
1929      GB     London   51.5085
...      ...        ...       ...
3432      GB     London   51.5092
3433      GB     London   51.5092
3434      GB     London   51.5092
3435      GB     London   51.5092
3437      GB     London   51.5085

[1485 rows x 3 columns]
```

PRACTICAL NO : 10

AIM : Retrieve different attributes of data.

CODE:

```
import pandas as pd
sFileName='IP_DATA_ALL.csv'
print('Loading :',sFileName)
IP_DATA_ALL=pd.read_csv(sFileName,header=0,low_memory=False, encoding="latin-1")

print('Rows:', IP_DATA_ALL.shape[0])
print('Columns:', IP_DATA_ALL.shape[1])
print('### Raw Data Set #####################################')
for i in range(0,len(IP_DATA_ALL.columns)):
   print(IP_DATA_ALL.columns[i],type(IP_DATA_ALL.columns[i]))
print('### Fixed Data Set #####################################')
IP_DATA_ALL_FIX=IP_DATA_ALL
for i in range(0,len(IP_DATA_ALL.columns)):
   cNameOld=IP_DATA_ALL_FIX.columns[i]
   cNameNew=cNameOld.strip().replace(" ", ".")
   IP_DATA_ALL_FIX.columns.values[i] = cNameNew
   print(IP_DATA_ALL.columns[i],type(IP_DATA_ALL.columns[i]))
print('Fixed Data Set with ID')
IP_DATA_ALL_with_ID=IP_DATA_ALL_FIX
IP_DATA_ALL_with_ID.index.names = ['RowID']
sFileName2='Retrieve_IP_DATA.csv'
IP_DATA_ALL_with_ID.to_csv(sFileName2, index = True, encoding="latin-1")
```

OUTPUT:

```
10.py
Loading : IP_DATA_ALL.csv
Rows: 1247502
Columns: 9
### Raw Data Set ####################################
Unnamed: 0 <class 'str'>
ID <class 'str'>
Country <class 'str'>
Place.Name <class 'str'>
Post.Code <class 'str'>
Latitude <class 'str'>
Longitude <class 'str'>
First.IP.Number <class 'str'>
Last.IP.Number <class 'str'>
### Fixed Data Set ##################################
Unnamed:.0 <class 'str'>
ID <class 'str'>
Country <class 'str'>
Place.Name <class 'str'>
Post.Code <class 'str'>
Latitude <class 'str'>
Longitude <class 'str'>
First.IP.Number <class 'str'>
Last.IP.Number <class 'str'>
Fixed Data Set with ID
```

PRACTICAL NO : 11

AIM : Drop the Columns Where All Elements Are Missing Values.

CODE:

```
import pandas as pd
sOutputFileName='Good-or-Bad-01.csv'
sFileName='Good-or-Bad.csv'
print('Loading :',sFileName)
RawData=pd.read_csv(sFileName,header=0)
print('!! Raw Data Values')
print(RawData)
print('!! Data Profile')
print('Rows :',RawData.shape[0])
print('Columns :',RawData.shape[1])
RawData.to_csv(sFileName, index = False)
TestData=RawData.dropna(axis=1, how='all') #0-ROW , 1-COL
TestData=TestData.dropna(axis=0, how='any') #0-ROW , 1-COL
print('!! Test Data Values')
print(TestData)
print('!! Data Profile')
print('Rows :',TestData.shape[0])
print('Columns :',TestData.shape[1])
sFileName=sOutputFileName
TestData.to_csv(sFileName, index = False)
```

OUTPUT:

```
11.py
Loading : Good-or-Bad.csv
!! Raw Data Values
      ID FieldA  FieldB FieldC  FieldD  FieldE    FieldF  FieldG
0    1.0   Good  Better   Best  1024.0     NaN   10241.0     1.0
1    2.0   Good     NaN   Best   512.0     NaN    5121.0     2.0
2    3.0   Good  Better    NaN   256.0     NaN     256.0     3.0
3    4.0   Good  Better   Best     NaN     NaN     211.0     4.0
4    5.0   Good  Better    NaN    64.0     NaN    6411.0     5.0
5    6.0   Good     NaN   Best    32.0     NaN      32.0     6.0
6    7.0    NaN  Better   Best    16.0     NaN    1611.0     7.0
7    8.0    NaN     NaN   Best     8.0     NaN    8111.0     8.0
8    9.0    NaN     NaN    NaN     4.0     NaN      41.0     9.0
9   10.0      A       B      C     2.0     NaN   21111.0    10.0
10   NaN    NaN     NaN    NaN     NaN     NaN       NaN     NaN
11  10.0   Good  Better   Best  1024.0     NaN  102411.0    12.0
12  10.0   Good     NaN   Best   512.0     NaN     512.0    13.0
13  10.0   Good  Better    NaN   256.0     NaN    1256.0    14.0
14  10.0   Good  Better   Best     NaN     NaN       NaN    15.0
15  10.0   Good  Better    NaN    64.0     NaN     164.0    16.0
16  10.0   Good     NaN   Best    32.0     NaN     322.0    17.0
17  10.0    NaN  Better   Best    16.0     NaN     163.0    18.0
18  10.0    NaN     NaN   Best     8.0     NaN     844.0    19.0
19  10.0    NaN     NaN    NaN     4.0     NaN    4555.0    20.0
20  10.0      A       B      C     2.0     NaN     111.0    21.0
!! Data Profile
Rows : 21
Columns : 8
!! Test Data Values
      ID FieldA  FieldB FieldC  FieldD    FieldF  FieldG
0    1.0   Good  Better   Best  1024.0   10241.0     1.0
9   10.0      A       B      C     2.0   21111.0    10.0
11  10.0   Good  Better   Best  1024.0  102411.0    12.0
20  10.0      A       B      C     2.0     111.0    21.0
!! Data Profile
Rows : 4
Columns : 7
```

PRACTICAL NO : 12

AIM : Drop the Columns Where Any of the Elements Is Missing Values/Keep Only the Rows That Contain a Maximum of Two Missing Values.

CODE:

```
import pandas as pd
sOutputFileName='Good-or-Bad-01.csv'
sFileName='Good-or-Bad.csv'
print('Loading :',sFileName)
RawData=pd.read_csv(sFileName,header=0)
print('!! Raw Data Values')
print(RawData)
print('!! Data Profile')
print('Rows :',RawData.shape[0])
print('Columns :',RawData.shape[1])
RawData.to_csv(sFileName, index = False)
TestData=RawData.dropna(axis=1, how='all') #0-ROW , 1-COL
TestData=TestData.dropna(axis=0, thresh=5) #0-ROW , 1-COL
print('!! Test Data Values')
print(TestData)
print('!! Data Profile')
print('Rows :',TestData.shape[0])
print('Columns :',TestData.shape[1])
sFileName=sOutputFileName
TestData.to_csv(sFileName, index = False)
```

OUTPUT:

```
Loading : Good-or-Bad.csv
!! Raw Data Values
      ID FieldA  FieldB FieldC  FieldD  FieldE    FieldF  FieldG
0    1.0   Good  Better   Best  1024.0     NaN   10241.0     1.0
1    2.0   Good     NaN   Best   512.0     NaN    5121.0     2.0
2    3.0   Good  Better    NaN   256.0     NaN     256.0     3.0
3    4.0   Good  Better   Best     NaN     NaN     211.0     4.0
4    5.0   Good  Better    NaN    64.0     NaN    6411.0     5.0
5    6.0   Good     NaN   Best    32.0     NaN      32.0     6.0
6    7.0    NaN  Better   Best    16.0     NaN    1611.0     7.0
7    8.0    NaN     NaN   Best     8.0     NaN    8111.0     8.0
8    9.0    NaN     NaN    NaN     4.0     NaN      41.0     9.0
9   10.0      A       B      C     2.0     NaN   21111.0    10.0
10   NaN    NaN     NaN    NaN     NaN     NaN       NaN     NaN
11  10.0   Good  Better   Best  1024.0     NaN  102411.0    12.0
12  10.0   Good     NaN   Best   512.0     NaN     512.0    13.0
13  10.0   Good  Better    NaN   256.0     NaN    1256.0    14.0
14  10.0   Good  Better   Best     NaN     NaN       NaN    15.0
15  10.0   Good  Better    NaN    64.0     NaN     164.0    16.0
16  10.0   Good     NaN   Best    32.0     NaN     322.0    17.0
17  10.0    NaN  Better   Best    16.0     NaN     163.0    18.0
18  10.0    NaN     NaN   Best     8.0     NaN     844.0    19.0
19  10.0    NaN     NaN    NaN     4.0     NaN    4555.0    20.0
20  10.0      A       B      C     2.0     NaN     111.0    21.0
!! Data Profile
Rows : 21
Columns : 8
```

```
!! Test Data Values
      ID FieldA  FieldB FieldC  FieldD    FieldF  FieldG
0    1.0   Good  Better   Best  1024.0   10241.0     1.0
1    2.0   Good     NaN   Best   512.0    5121.0     2.0
2    3.0   Good  Better    NaN   256.0     256.0     3.0
3    4.0   Good  Better   Best     NaN     211.0     4.0
4    5.0   Good  Better    NaN    64.0    6411.0     5.0
5    6.0   Good     NaN   Best    32.0      32.0     6.0
6    7.0    NaN  Better   Best    16.0    1611.0     7.0
7    8.0    NaN     NaN   Best     8.0    8111.0     8.0
9   10.0      A       B      C     2.0   21111.0    10.0
11  10.0   Good  Better   Best  1024.0  102411.0    12.0
12  10.0   Good     NaN   Best   512.0     512.0    13.0
13  10.0   Good  Better    NaN   256.0    1256.0    14.0
14  10.0   Good  Better   Best     NaN       NaN    15.0
15  10.0   Good  Better    NaN    64.0     164.0    16.0
16  10.0   Good     NaN   Best    32.0     322.0    17.0
17  10.0    NaN  Better   Best    16.0     163.0    18.0
18  10.0    NaN     NaN   Best     8.0     844.0    19.0
20  10.0      A       B      C     2.0     111.0    21.0
!! Data Profile
Rows : 18
Columns : 7
```

Practical 1 : Design a simple linear neural network model.

Code :

```
x=float(input("Enter value of x:"))
w=float(input("Enter value of weight w:"))
b=float(input("Enter value of bias b:"))
net = w*x+b
if(net<0):
   out=0
elif((net>=0)&(net<=1)):
   out =net
else:
   out=1
print("net=",net)
print("output=",out)
```

Output :

```
Enter value of x:2
Enter value of weight w:3
Enter value of bias b:4
net= 10.0
output= 1
```

Practical 2 : Calculate the output of a neural net using both binary and bipolar sigmoidal function.

For the network shown in the figure 1, calculate the net input to output neuron.

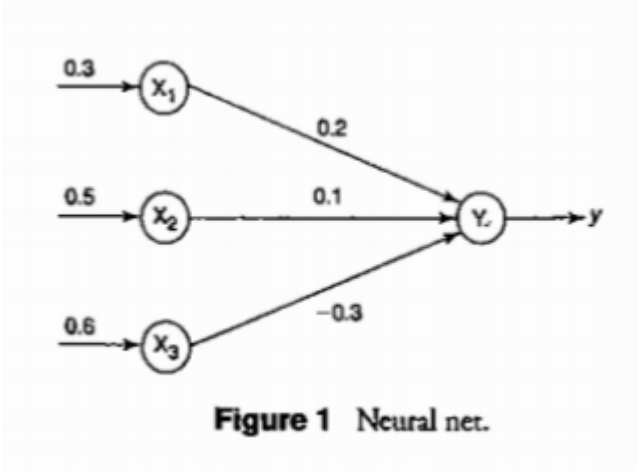


Figure 1 Neural net.

Solution :The given neural net consist of three input neurons and one output neuron. The inputs and weight are

[x1, x2, x3] = [0.3, 0.5, 0.6]
[w1, w2, w3] = [0.2, 0.1, -0.3]

The net input can be calculated as

Yin = x1w1+x2w2+x3w3
= 0.3*0.2+0.5*0.1+0.6*(-0.3)
= -0.07

Code :

```
x=[0.3, 0.5, 0.6]
w=[0.2, 0.1,-0.3]
y=0
for i in range(0,len(x)):
    y=y+x[i]*w[i]

print("x:",x)
print("w:",w)
print("y:",round(y,3))
```

Output :

```
x: [0.3, 0.5, 0.6]
w: [0.2, 0.1, -0.3]
y: -0.07
```

Practical 3 : Calculate the net input for the network shown in Figure 2 with bias included in the network.

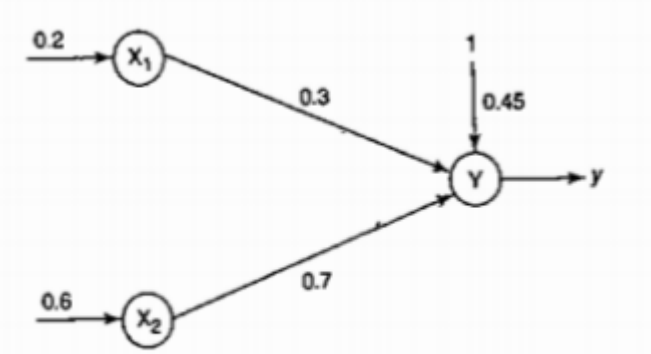


Figure 2 Simple neural net.

Solution: The given net consists of two input neurons, a bias and an output neuron. The inputs are

$[x_1, X_2] = [0.2, 0.6]$ and the weights are $[w_1, w_2] = [0.3, 0.7]$. Since the bias is included b = 0.45 and bias input x_o is equal to 1, the net input is calculated as

$$Yin = b + x_1W_1 + X_2W_2$$
$$= 0.45 + 0.2 \text{ X } 0.3 + 0.6 \text{ X } 0.7$$
$$= 0.45 + 0.06 + 0.42 = 0.93$$

Therefore $y_m = 0.93$ is the net input.

Code :

```
x=[0.2, 0.6]
w=[0.3, 0.7]
y=0
b=0.45
for i in range(0,len(x)):
    y=y+x[i]*w[i]
y=y+b
print("x:",x)
print("w:",w)
print("y:",round(y,3))
```

Output:

```
x: [0.2, 0.6]
w: [0.3, 0.7]
y: 1.38
>>>
```

Practical 4: Implement AND/NOT function using McCulloch-Pits neuron (use binary data representation).

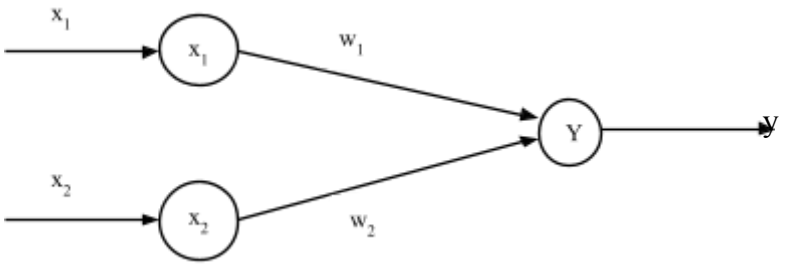
Solution:

In the case of AND/NOT function, the response is true if the first input is true and the second input is false. For all the other variations, the response is false. The truth table for ANDNOT function is given in Table below.

Truth Table:

w1	**x_1**	**x_2**	**w2**	**y**	
1	0	0	-1	**0**	
1	0	1	-1	0	-1
1	1	0	-1	**1**	1
1	1	1	-1	**0**	

The given function gives an output only when $x_1 = 1$ and $x_2 = 0$. The weights have to be decided only after the analysis. The net can be represent as shown in figure below:



Neural net (weights fixed after analysis).

Code :

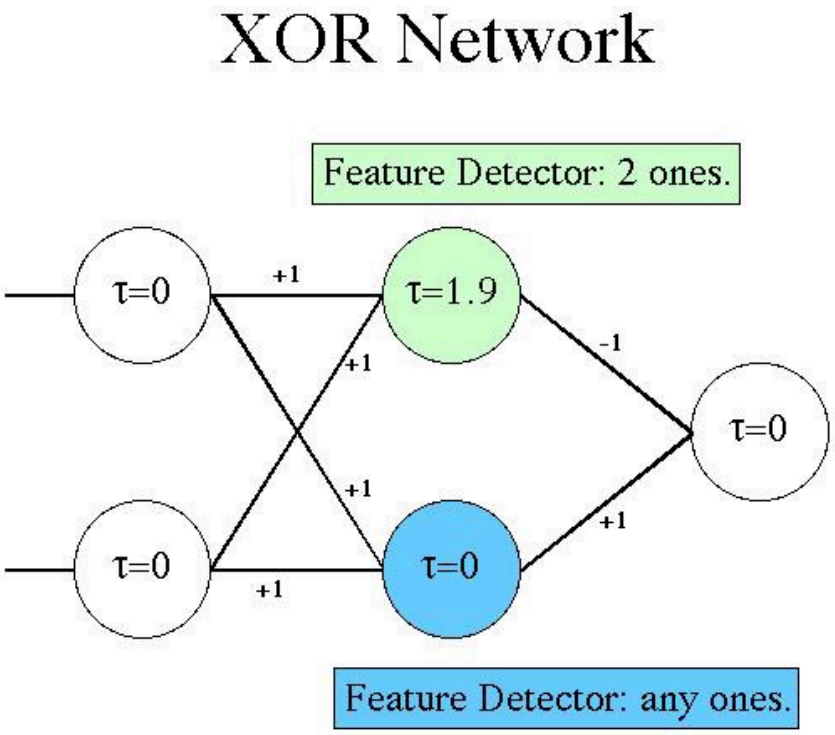
```
x1=[0,0,1,1]
x2=[0,1,0,1]
w1=1
w2=-1
y=[]
for i in range(0,len(x1)):
    xm=x1[i]*w1+x2[i]*w2
    if(xm<=0):
        y.append(0)
    else:
        y.append(1)
```

```
print("x1:",x1)
print("x2:",x2)
print("y:",y)
```

Output :

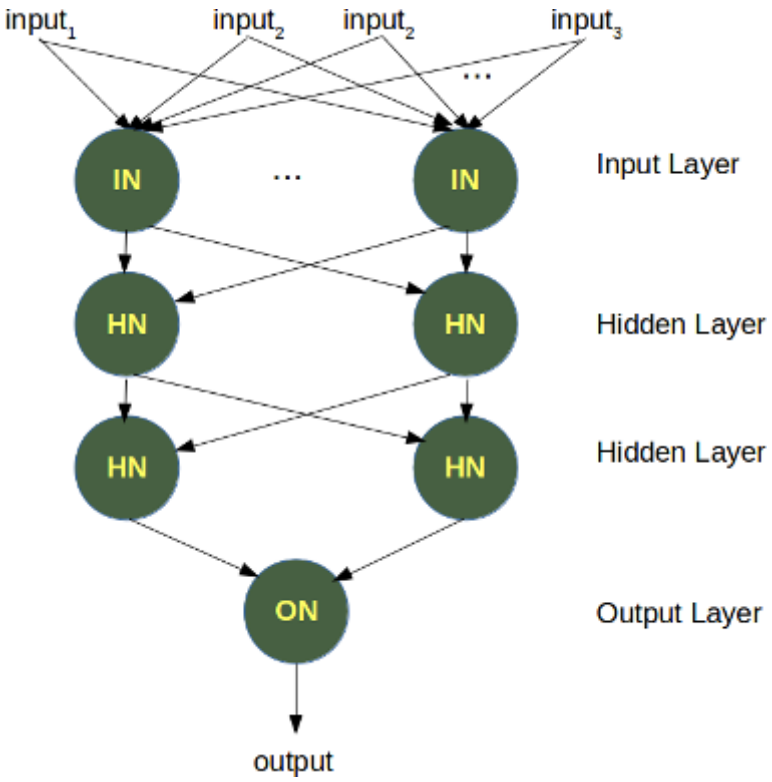
```
x1: [0, 0, 1, 1]
x2: [0, 1, 0, 1]
y: [0, 0, 1, 0]
```

Practical 5: Generate XOR function using McCulloch-Pitts neural net



The XOR (exclusive or) function is defined by the following truth table:

w1	Input1	w2	Input2	XOR Output	
1	0	1	0	**0**	
1	0	1	1	**1**	
1	1	1	0	**1**	
1	1	1	1	0	2



Code :

```
x1=[0,0,1,1]
x2=[0,1,0,1]
w1=1
w2=1
y=[]
for i in range(0,len(x1)):
    xm=x1[i]*w1+x2[i]*w2
    if(xm==1):
        y.append(1)
    else:
        y.append(0)

print("x1:",x1)
print("x2:",x2)
print("y:",y)
```

Output :

```
x1: [0, 0, 1, 1]
x2: [0, 1, 0, 1]
y: [0, 1, 1, 0]
```

Practical 6 : Write a program to implement Hebb's rule.

The Hebbian Learning Rule is a learning rule that specifies how much the weight of the connection between two units should be increased or decreased in proportion to the product of their activation. The rule builds on Hebbs's 1949 learning rule which states that the connections between two neurons might be strengthened if the neurons fire simultaneously. The Hebbian Rule works well as long as all the input patterns are orthogonal or uncorrelated. The requirement of orthogonality places serious limitations on the Hebbian Learning Rule. A more powerful learning rule is the delta rule, which utilizes the discrepancy between the desired and actual output of each output unit to change the weights feeding into it.

Hebb's rule with an analogy. Psychology and neuroscience

Hebb's rule or Hebb's law or Hebbian theory is fundamental to understand the relationship between psychology and neuroscience. To approach it we will go back to the original work of Donald O. Hebb and, later on, we will explain it through an analogy that will facilitate our understanding.

Donald Hebb is the creator of the most mentioned 'principle' in psychobiology, or behavioural neuroscience.From the so-called Hebb's law, or Hebb's rule of the Hebbian learning (Hebb learning rule).We will see it through an analogy by the end of this post.

Donald O. Hebb and his contribution

Donald Olding Hebb (1904-1985) was a Canadian psychologist pioneer of neuropsychology (of the study of the relationship between psychology and neuroscience).

The Hebb's principle or Hebb's rule

Hebb says that "when the axon of a cell A is close enough to excite a B cell and takes part on its activation in a repetitive and persistent way, some type of growth process or metabolic change takes place in one or both cells, so that increases the efficiency of cell A in the activation of B ".

'neurons that fire together wire together'

It is customary to be summarized as "neurons that fire together wire together".That is, the simultaneous activation of nearby neurons leads to an increase in the strength of synaptic connection between them.

It is important to note that the neurons must be previously connected, sufficiently close to one another, so that the synapse can be reinforced.

, Hebb's principle can be described as a method of determining how to alter the weights between model neurons. The weight between two neurons increases if the two neurons activate simultaneously, and reduces if they activate separately. Nodes that tend to be either both positive or both negative at the same time have strong positive weights, while those that tend to be opposite have strong negative weights.

```
#first pattern
x1=[1,1,1,-1,1,-1,1,1,1]
#second pattern
x2=[1,1,1,1,-1,1,1,1,1]
```

```
#initialize bais value
b=0
#define target
y0=1
y1=-1

wtold=[0,0,0,0,0,0,0,0,0]


print("First input with target =1")
for i in range(0,9):
    wtold[i]=wtold[i]+x1[i]*y0
b=b+y0
print("new wt =", wtold)
print("Bias value",b)

print("Second input with target =-1")
for i in range(0,9):
    wtold[i]=wtold[i]+x2[i]*y1
b=b+y1
print("new wt =", wtold)
print("Bias value",b)
```

OutPut :

```
First input with target =1
new wt = [ 1  1  1 -1  1 -1  1  1  1]
Bias value 1
Second input with target =-1
new wt = [ 0  0  0 -2  2 -2  0  0  0]
Bias value 0
```

Practical 7: Write a program to implement of delta rule.

```
#supervised learning
import numpy as np
import time
np.set_printoptions(precision=2)
x=np.zeros((3,))
weights=np.zeros((3,))
desired=np.zeros((3,))
actual=np.zeros((3,))

for i in range(0,3):
   x[i]=float(input("Initial inputs:"))
#1 1 1
for i in range(0,3):
   weights[i]=float(input("Initial weights:"))
#1 1 1
for i in range(0,3):
   desired[i]=float(input("Desired output:"))
#5 10 20
a=float(input("Enter learning rate:"))
#1
actual=x*weights
print("actual",actual)
print("desired",desired)

while True:
   if np.array_equal(desired,actual):
      break    #no change
   else:
      for i in range(0,3):
         weights[i]=weights[i]+a*(desired[i]-actual[i])
         # w0-5, w1-10 w2-20
```

```
    actual=x*weights
    print("weights",weights)
    print("actual",actual)
    print("desired",desired)
print("*"*30)
print("Final output")
print("Corrected weights",weights)
print("actual",actual)
print("desired",desired)
```

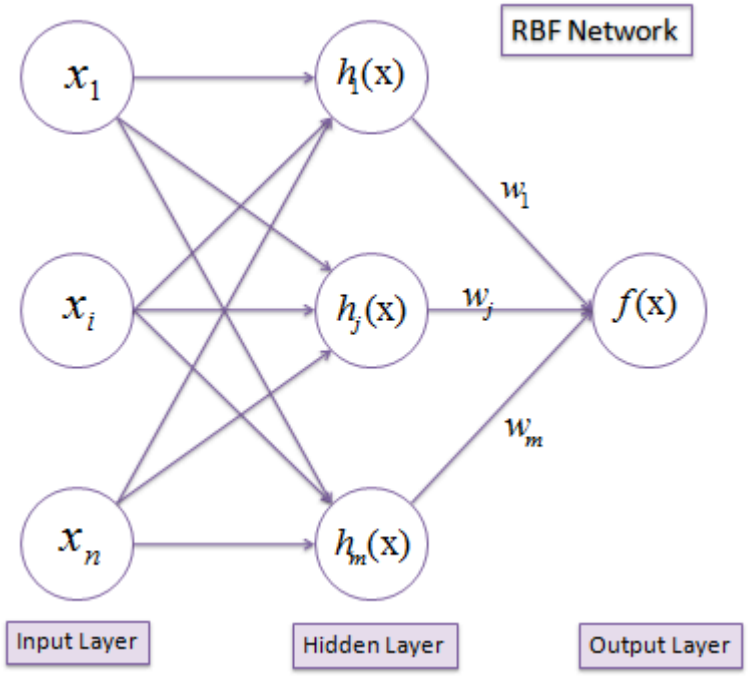
Output :

```
Initial inputs:1
Initial inputs:1
Initial inputs:1
Initial weights:1
Initial weights:1
Initial weights:1
Desired output:2
Desired output:3
Desired output:4
Enter learning rate:1
actual [1. 1. 1.]
desired [2. 3. 4.]
weights [2. 3. 4.]
actual [2. 3. 4.]
desired [2. 3. 4.]
******************************
Final output
corrected weights [2. 3. 4.]
actual [2. 3. 4.]
desired [2. 3. 4.]
```

Practical 8: Write a program for Radial Basis function

Radial Basis Function Networks (RBF)

RBF networks have three layers: input layer, hidden layer and output layer. One neuron in the input layer corresponds to each predictor variable. With respects to categorical variables, *n*-1 neurons are used where *n* is the number of categories. Hidden layer has a variable number of neurons. Each neuron consists of a radial basis function centered on a point with the same dimensions as the predictor variables. The output layer has a weighted sum of outputs from the hidden layer to form the network outputs.



$$f(x) = \sum_{j=1}^{m} w_j h_j(x)$$

$$h(x) = \exp\left(-\frac{(x-c)^2}{r^2}\right)$$

Algorithm

$h(x)$ is the Gaussian activation function with the parameters r (the radius or standard deviation) and c (the center or average taken from the input space) defined separately at each RBF unit. The learning process is based on adjusting the parameters of the network to reproduce a set of input-output patterns. There are three types of parameters; the weight w between the hidden nodes and the output nodes, the center c of each neuron of the hidden layer and the unit width r.

Unit Center (c)

Any clustering algorithm can be used to determine the RBF unit centers (e.g., K-means clustering). A set of clusters each with r-dimensional centers is determined by the number of input variables or nodes of the input layer. The cluster centers become the centers of the RBF units. The number of clusters, H, is a design parameter and determines the number of nodes in the hidden layer. The K-means clustering algorithm proceeds as follows:

1. Initialize the center of each cluster to a different randomly selected training pattern.
2. Assign each training pattern to the nearest cluster. This can be accomplished by calculating the Euclidean distances between the training patterns and the cluster centers.
3. When all training patterns are assigned, calculate the average position for each cluster center. They then become new cluster centers.

5.

4. Repeat steps 2 and 3, until the cluster centers do not change during the subsequent iterations.
Unit width (*r*)
When the RBF centers have been established, the width of each RBF unit can be calculated using the K-nearest neighbors algorithm. A number K is chosen, and for each center, the K nearest centers is found. The root-mean squared distance between the current cluster center and its K nearest neighbors is calculated, and this is the value chosen for the unit width (*r*). So, if the current cluster center is c_j, the *r* value is:
$$r_j = \sqrt{\frac{\sum_{i=1}^{k} (c_j - c_i)^2}{k}}$$
A typical value for K is 2, in which case s is set to be the average distance from the two nearest neighboring cluster centers.
Weights (*w*)
Using the linear mapping, **w** vector is calculated using the output vector (**y**) and the design matrix **H**.
$$y = wH$$ $$w = (H'H)\, H'y$$

The basis functions are (unnormalized) gaussians, the output layer is linear and the weights are learned by a simple pseudo-inverse.

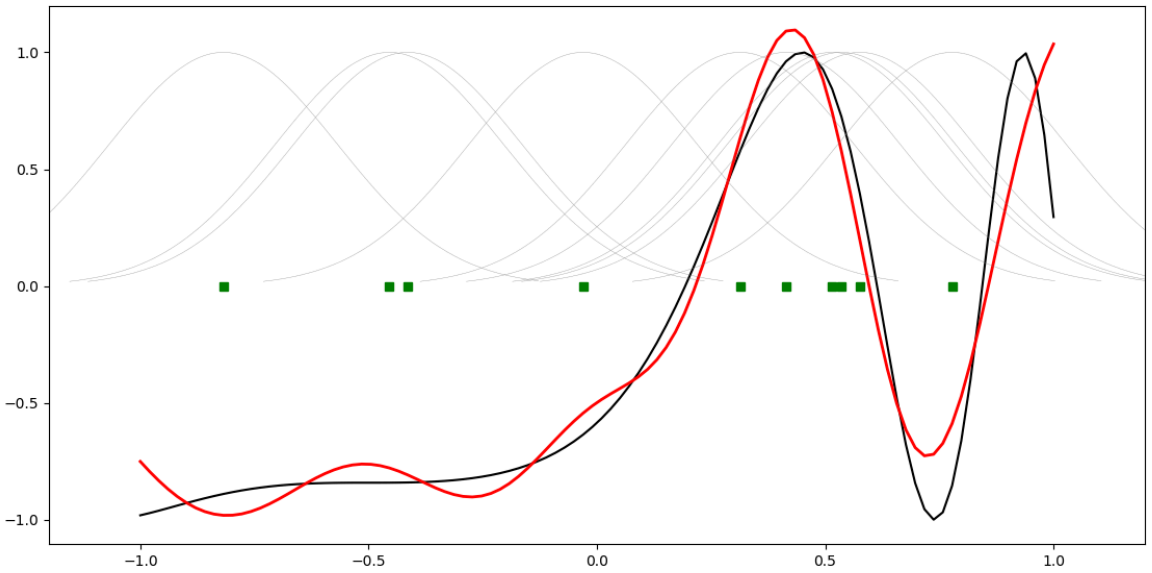
```
from scipy import *
from scipy.linalg import norm, pinv
from matplotlib import pyplot as plt
 from numpy import *
class RBF:

    def __init__(self, indim, numCenters, outdim):
        self.indim =indim
        self.outdim =outdim
        self.numCenters =numCenters
        self.centers =[random.uniform(-1, 1, indim) for i in range(numCenters)]
```

```
        self.beta =8
        self.W =random.random((self.numCenters, self.outdim))

    def _basisfunc(self, c, d):
        assert len(d) ==self.indim
        return exp(-self.beta *norm(c-d)**2)

    def _calcAct(self, X):
        # calculate activations of RBFs
        G =zeros((X.shape[0], self.numCenters), float)
        for ci, c in enumerate(self.centers):
            for xi, x in enumerate(X):
                G[xi,ci] =self._basisfunc(c, x)
        return G

    def train(self, X, Y):
        """ X: matrix of dimensions n x indim
            y: column vector of dimension n x 1 """

        # choose random center vectors from training set
        rnd_idx =random.permutation(X.shape[0])[:self.numCenters]
        self.centers =[X[i,:] for i in rnd_idx]

        print("center", self.centers)
        # calculate activations of RBFs
        G =self._calcAct(X)
        print (G)

        # calculate output weights (pseudoinverse)
        self.W =dot(pinv(G), Y)

    def test(self, X):
```

```
        """ X: matrix of dimensions n x indim """

        G =self._calcAct(X)
        Y =dot(G, self.W)
        return Y


if __name__ =='__main__':
    # ----- 1D Example ------------------------------------------------
    n =100

    x =mgrid[-1:1:complex(0,n)].reshape(n, 1)
    # set y and add random noise
    y =sin(3*(x+0.5)**3-1)
    # y += random.normal(0, 0.1, y.shape)

    # rbf regression
    rbf =RBF(1, 10, 1)
    rbf.train(x, y)
    z =rbf.test(x)

    # plot original data
    plt.figure(figsize=(12, 8))
    plt.plot(x, y, 'k-')

    # plot learned model
    plt.plot(x, z, 'r-', linewidth=2)

    # plot rbfs
    plt.plot(rbf.centers, zeros(rbf.numCenters), 'gs')

    for c in rbf.centers:
```

```
    # RF prediction lines
    cx =arange(c-0.7, c+0.7, 0.01)
    cy =[rbf._basisfunc(array([cx_]), array([c])) for cx_ in cx]
    plt.plot(cx, cy, '-', color='gray', linewidth=0.2)

plt.xlim(-1.2, 1.2)
plt.show()
```

Output :



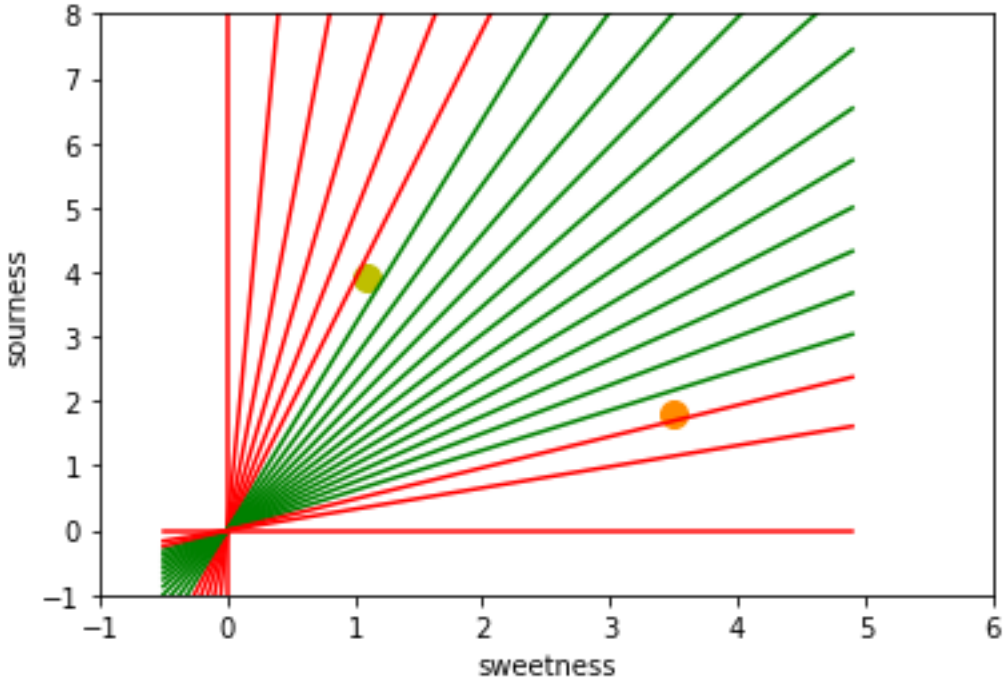
Practical 9: Line Separation

You could imagine that you have two attributes describing am eddible object like a fruit for example: "sweetness" and "sourness"
We could describe this by points in a two-dimensional space. The x axis for the sweetness and the y axis for the sourness. Imagine now that we have two fruits as points in this space, i.e. an orange at position (3.5, 1.8) and a lemon at (1.1, 3.9).
We could define dividing lines to define the points which are more lemon-like and which are more orange-like. The following program calculates and renders a bunch of lines. The red ones are completely unusable for this purpose, because they are not separating the classes. Yet, it is obvious that even the green ones are not all useful.

```
import numpy as np
import matplotlib.pyplot as plt
def create_distance_function(a, b, c):
    """ 0 = ax + by + c """
    def distance(x, y):
        """ returns tuple (d, pos)
            d is the distance
            If pos == -1 point is below the line,
            0 on the line and +1 if above the line
        """
        nom = a * x + b * y + c
        if nom == 0:
            pos = 0
        elif (nom<0 and b<0) or (nom>0 and b>0):
            pos = -1
        else:
            pos = 1
        return (np.absolute(nom) / np.sqrt( a ** 2 + b ** 2), pos)
    return distance

points = [ (3.5, 1.8), (1.1, 3.9) ]
fig, ax = plt.subplots()
ax.set_xlabel("sweetness")
ax.set_ylabel("sourness")
ax.set_xlim([-1, 6])
ax.set_ylim([-1, 8])
X = np.arange(-0.5, 5, 0.1)
colors = ["r", ""] # for the samples
size = 10
for (index, (x, y)) in enumerate(points):
    if index== 0:
        ax.plot(x, y, "o", color="darkorange", markersize=size)
    else:
        ax.plot(x, y, "oy", markersize=size)
        step = 0.05
for x in np.arange(0, 1+step, step):
    slope = np.tan(np.arccos(x))
    dist4line1 = create_distance_function(slope, -1, 0)
    #print("x: ", x, "slope: ", slope)
    Y = slope * X
```

```
    results = []
    for point in points:
        results.append(dist4line1(*point))
    #print(slope, results)
    if (results[0][1] != results[1][1]):
        ax.plot(X, Y, "g-")
    else:
        ax.plot(X, Y, "r-")

plt.show()
```

Output :



Practical 10: Membership and Identity operators in, not in.

```
# Python program to illustrate
# Finding common member in list
# without using 'in' operator

# Define a function() that takes two lists
def overlapping(list1,list2):
	c=0
	d=0
	for i in list1:
		c+=1
	for i in list2:
		d+=1
	for i in range(0,c):
		for j in range(0,d):
			if(list1[i]==list2[j]):
				return 1
	return 0
list1=[1,2,3,4,5]
list2=[6,7,8,9]
if(overlapping(list1,list2)):
	print("overlapping")
else:
	print("not overlapping")
```

```
# Python program to illustrate
# Finding common member in list
# without using 'in' operator

# Define a function() that takes two lists
def overlapping(list1,list2):

	c=0
	d=0
	for i in list1:
		c+=1
	for i in list2:
		d+=1
	for i in range(0,c):
		for j in range(0,d):
			if(list1[i]==list2[j]):
				return 1
	return 0
list1=[1,2,3,4,5]
list2=[6,7,8,9]
if(overlapping(list1,list2)):
	print("overlapping")
else:
	print("not overlapping")
```

Practical 8b: Membership and Identity Operators is, is not

```
# Python program to illustrate the use
# of 'is' identity operator
x = 5
if (type(x) is int):
        print ("true")
else:
        print ("false")
```

```
# Python program to illustrate the
# use of 'is not' identity operator
x = 5.2
if (type(x) is not int):
        print ("true")
else:
        print ("false")
```

Output :

```
[1, 2, 3, 4, 5]
[6, 7, 1, 8, 9]
not overlapping
Enter any number:3
3  is present in the list
true
```

Practical 11: Solve Tipping Problem using fuzzy logic

Fuzzy Control Systems: The Tipping Problem
==========

The 'tipping problem' is commonly used to illustrate the power of fuzzy logic principles to generate complex behavior from a compact, intuitive set of expert rules.

If you're new to the world of fuzzy control systems, you might want to check out the `Fuzzy Control Primer <../userguide/fuzzy_control_primer.html>`_ before reading through this worked example.

The Tipping Problem

Let's create a fuzzy control system which models how you might choose to tip at a restaurant. When tipping, you consider the service and food quality, rated between 0 and 10. You use this to leave a tip of between 0 and 25%.

We would formulate this problem as:

* Antecednets (Inputs)
  - `service`
    * Universe (ie, crisp value range): How good was the service of the wait staff, on a scale of 0 to 10?
    * Fuzzy set (ie, fuzzy value range): poor, acceptable, amazing
  - `food quality`
    * Universe: How tasty was the food, on a scale of 0 to 10?
    * Fuzzy set: bad, decent, great
* Consequents (Outputs)
  - `tip`
    * Universe: How much should we tip, on a scale of 0% to 25%
    * Fuzzy set: low, medium, high
* Rules
  - IF the *service* was good *or* the *food quality* was good, THEN the tip will be high.
  - IF the *service* was average, THEN the tip will be medium.
  - IF the *service* was poor *and* the *food quality* was poor THEN the tip will be low.
* Usage
  - If I tell this controller that I rated:
    * the service as 9.8, and
    * the quality as 6.5,
  - it would recommend I leave:
    * a 20.2% tip.

Creating the Tipping Controller Using the skfuzzy control API

We can use the `skfuzzy` control system API to model this. First, let's
define fuzzy variables
"""

```
import numpy as np
import skfuzzy as fuzz
from skfuzzy import control as ctrl

# New Antecedent/Consequent objects hold universe variables and membership
# functions
quality = ctrl.Antecedent(np.arange(0, 11, 1), 'quality')
service = ctrl.Antecedent(np.arange(0, 11, 1), 'service')
tip = ctrl.Consequent(np.arange(0, 26, 1), 'tip')

# Auto-membership function population is possible with .automf(3, 5, or 7)
quality.automf(3)
service.automf(3)

# Custom membership functions can be built interactively with a familiar,
# Pythonic API
tip['low'] = fuzz.trimf(tip.universe, [0, 0, 13])
tip['medium'] = fuzz.trimf(tip.universe, [0, 13, 25])
tip['high'] = fuzz.trimf(tip.universe, [13, 25, 25])
```

"""
To help understand what the membership looks like, use the ``view`` methods.
"""

```
# You can see how these look with .view()
quality['average'].view()
```

"""
.. image:: PLOT2RST.current_figure
"""

```
service.view()
```

"""
.. image:: PLOT2RST.current_figure
"""

```
tip.view()
```

"""
.. image:: PLOT2RST.current_figure

Fuzzy rules

Now, to make these triangles useful, we define the *fuzzy relationship*
between input and output variables. For the purposes of our example, consider
three simple rules:

1. If the food is poor OR the service is poor, then the tip will be low
2. If the service is average, then the tip will be medium
3. If the food is good OR the service is good, then the tip will be high.

Most people would agree on these rules, but the rules are fuzzy. Mapping the

```
imprecise rules into a defined, actionable tip is a challenge. This is the
kind of task at which fuzzy logic excels.
"""

rule1 = ctrl.Rule(quality['poor'] | service['poor'], tip['low'])
rule2 = ctrl.Rule(service['average'], tip['medium'])
rule3 = ctrl.Rule(service['good'] | quality['good'], tip['high'])

rule1.view()

"""
.. image:: PLOT2RST.current_figure

Control System Creation and Simulation
---------------------------------------

Now that we have our rules defined, we can simply create a control system
via:
"""

tipping_ctrl = ctrl.ControlSystem([rule1, rule2, rule3])

"""
In order to simulate this control system, we will create a
``ControlSystemSimulation``.  Think of this object representing our controller
applied to a specific set of cirucmstances.  For tipping, this might be tipping
Sharon at the local brew-pub.  We would create another
``ControlSystemSimulation`` when we're trying to apply our ``tipping_ctrl``
for Travis at the cafe because the inputs would be different.
"""

tipping = ctrl.ControlSystemSimulation(tipping_ctrl)

"""
We can now simulate our control system by simply specifying the inputs
and calling the ``compute`` method.  Suppose we rated the quality 6.5 out of 10
and the service 9.8 of 10.
"""
# Pass inputs to the ControlSystem using Antecedent labels with Pythonic API
# Note: if you like passing many inputs all at once, use .inputs(dict_of_data)
tipping.input['quality'] = 6.5
tipping.input['service'] = 9.8

# Crunch the numbers
tipping.compute()

"""
Once computed, we can view the result as well as visualize it.
"""
print tipping.output['tip']
tip.view(sim=tipping)
"""
```

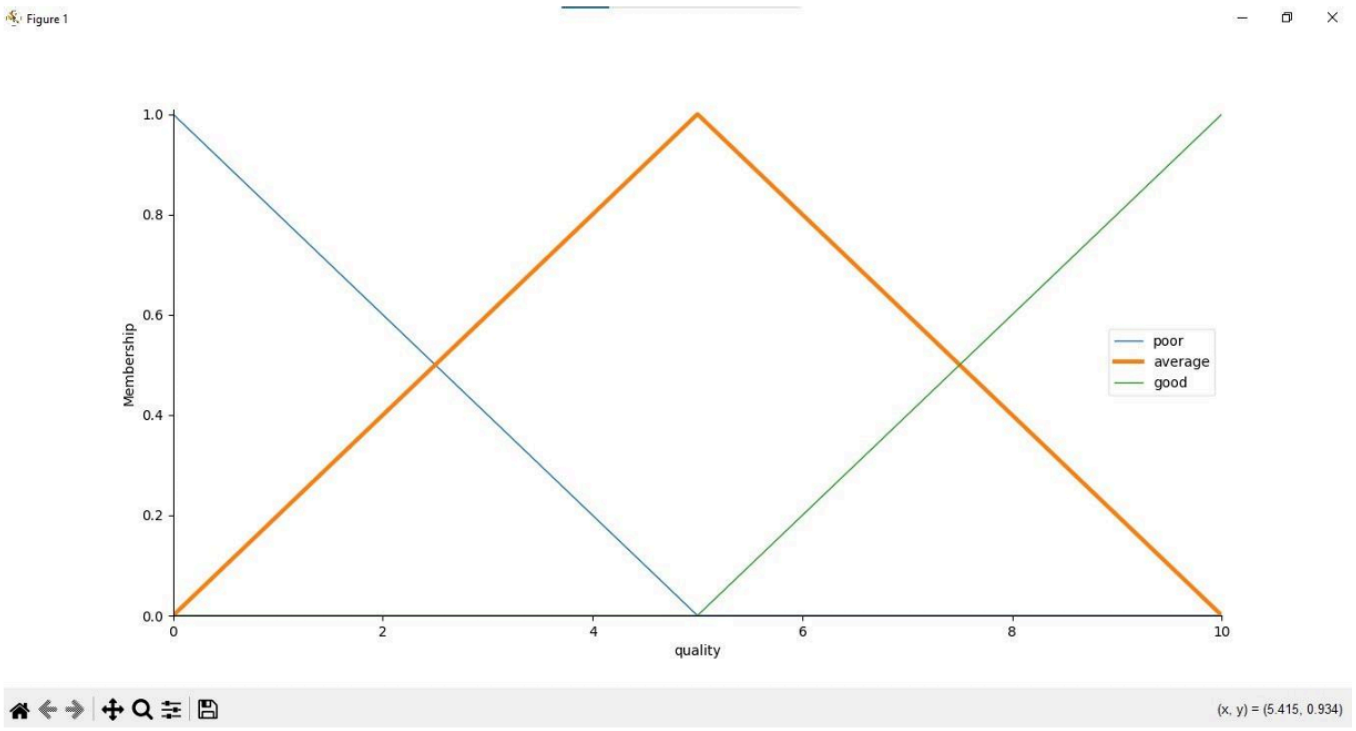
.. image:: PLOT2RST.current_figure

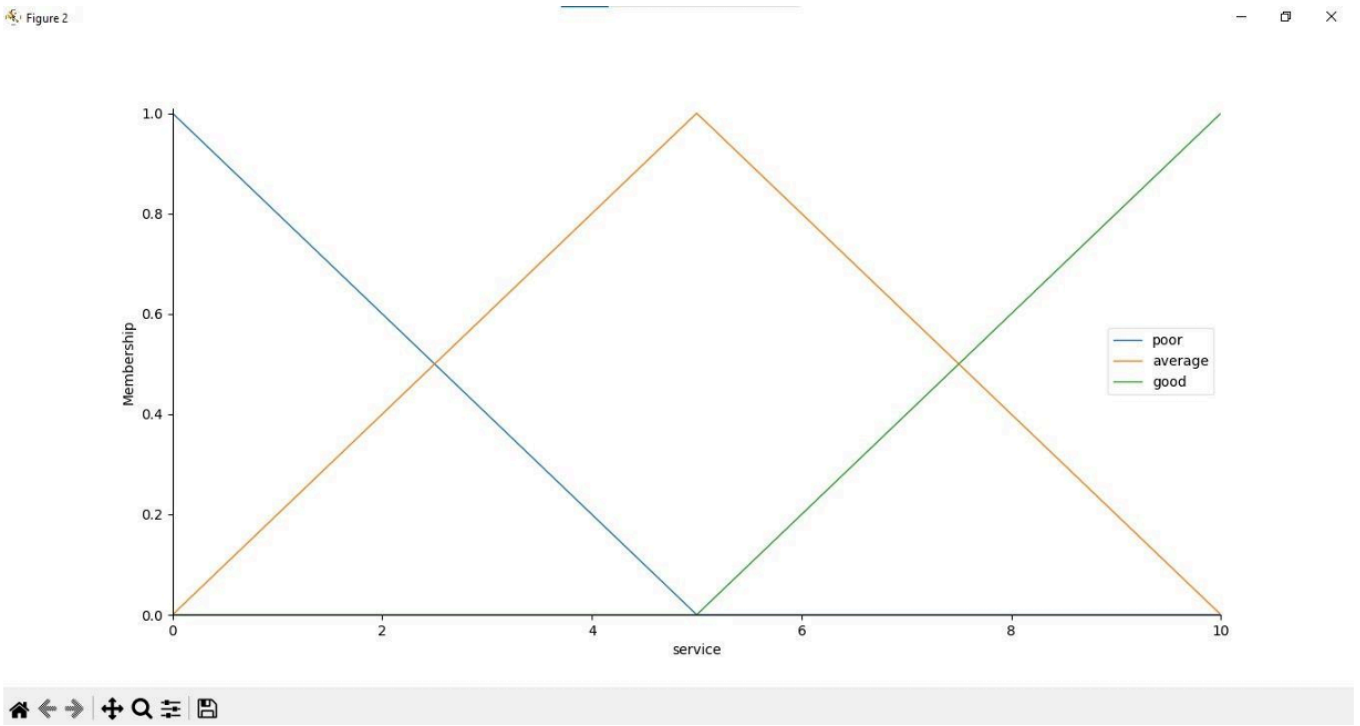
The resulting suggested tip is **20.24%**.

```
#pip3 install scikit-fuzzy
import numpy as np
import skfuzzy as fuzz
from skfuzzy import control as ctrl

quality = ctrl.Antecedent(np.arange(0, 11, 1), 'quality')
service = ctrl.Antecedent(np.arange(0, 11, 1), 'service')
tip = ctrl.Consequent(np.arange(0, 26, 1), 'tip')
quality.automf(3)
service.automf(3)

# Custom membership functions can be built interactively with a familiar,
# Pythonic API
tip['low'] = fuzz.trimf(tip.universe, [0, 0, 13])
tip['medium'] = fuzz.trimf(tip.universe, [0, 13, 25])
tip['high'] = fuzz.trimf(tip.universe, [13, 25, 25])

quality['average'].view()
service.view()
tip.view()
rule1 = ctrl.Rule(quality['poor'] | service['poor'], tip['low'])
rule2 = ctrl.Rule(service['average'], tip['medium'])
rule3 = ctrl.Rule(service['good'] | quality['good'], tip['high'])

rule1.view()

tipping_ctrl = ctrl.ControlSystem([rule1, rule2, rule3])

tipping = ctrl.ControlSystemSimulation(tipping_ctrl)

tipping.input['quality'] = 6.5
```

```
tipping.input['service'] = 9.8
tipping.compute()
print (tipping.output['tip'])
tip.view(sim=tipping)
```

Output :





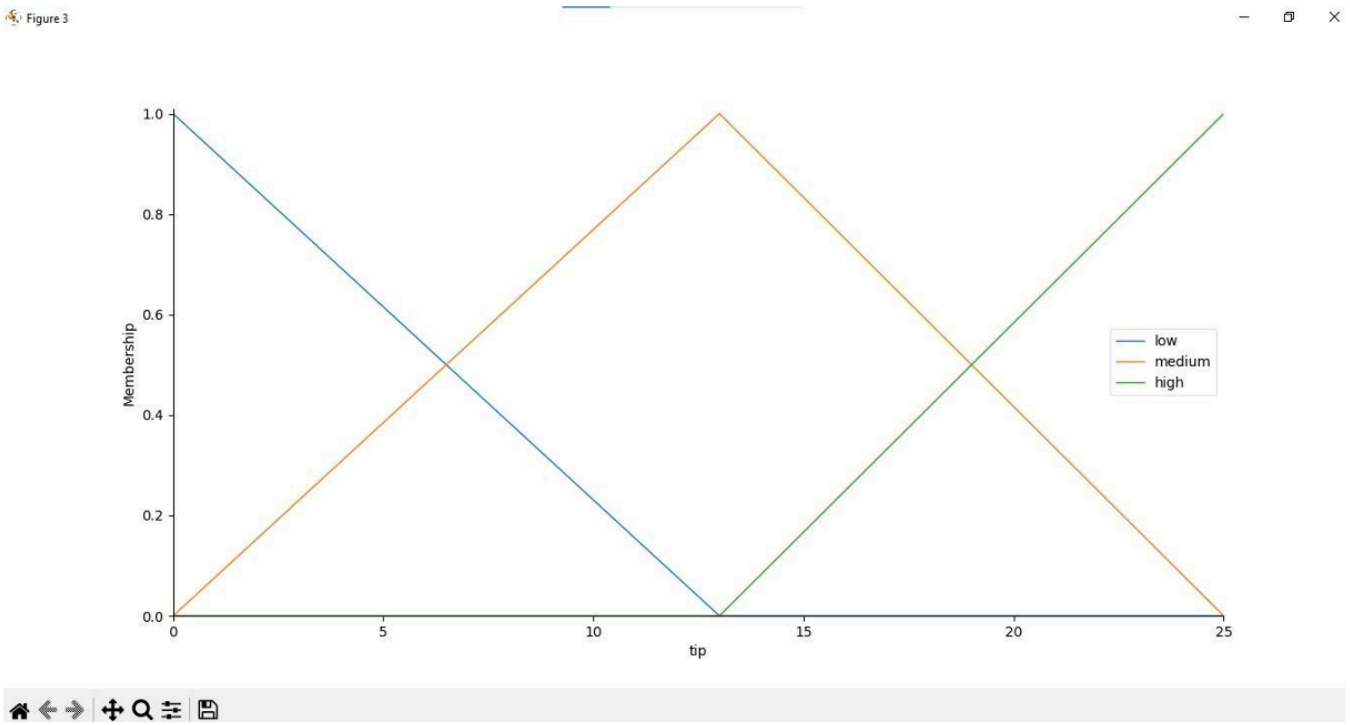


Figure 3
1.0
0.8
0.6
0.4
0.2
0.0
Membership
0
5
10
15
20
25
tip
low
medium
high

Practical No. 1

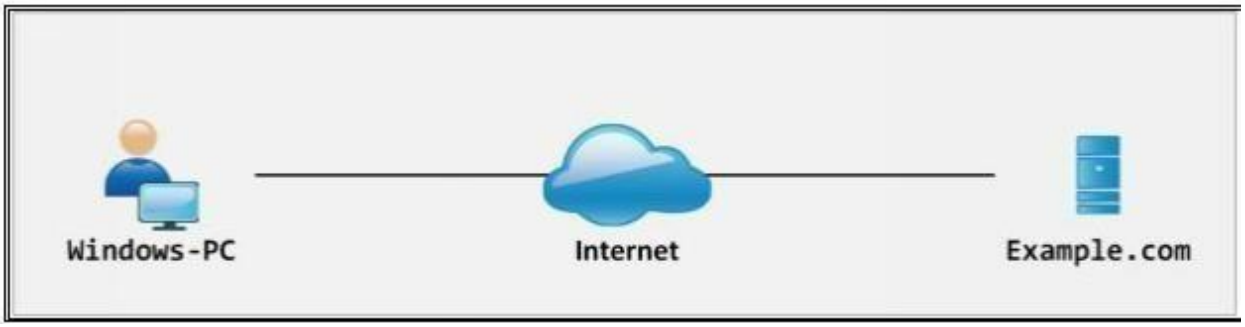
A. Tools to perform footprinting and reconnaissance

Footprinting and reconnaissance are used to collect basic information about the target systems in order to exploit them. The target information is IP location information, routing information, business information, address, phone number and DNS records.

i. Windows Command Line Utilities

Consider a network where you have access to a Windows PC connected to the Internet. Using Windows-based tools, let's gather some information about the target. You can assume any target domain or IP address, in our case, we are using **example.com** as a target.

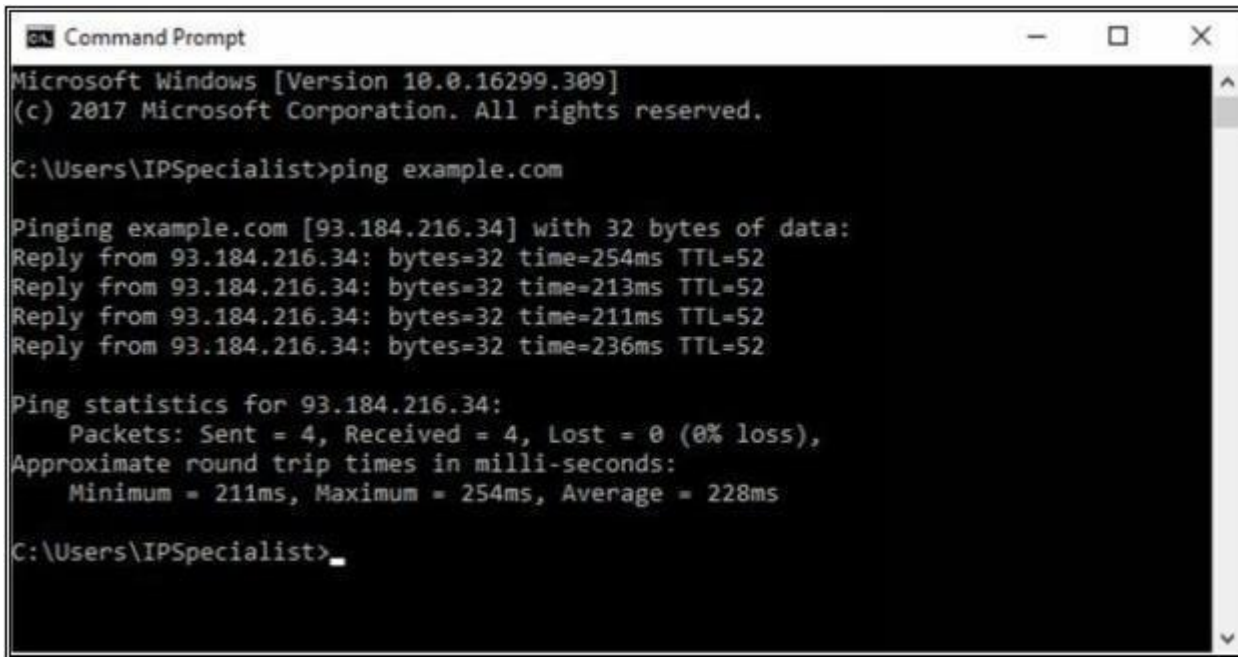
Topology Diagram:



- **Ping**

1- Open Windows Command Line (cmd) from Windows PC

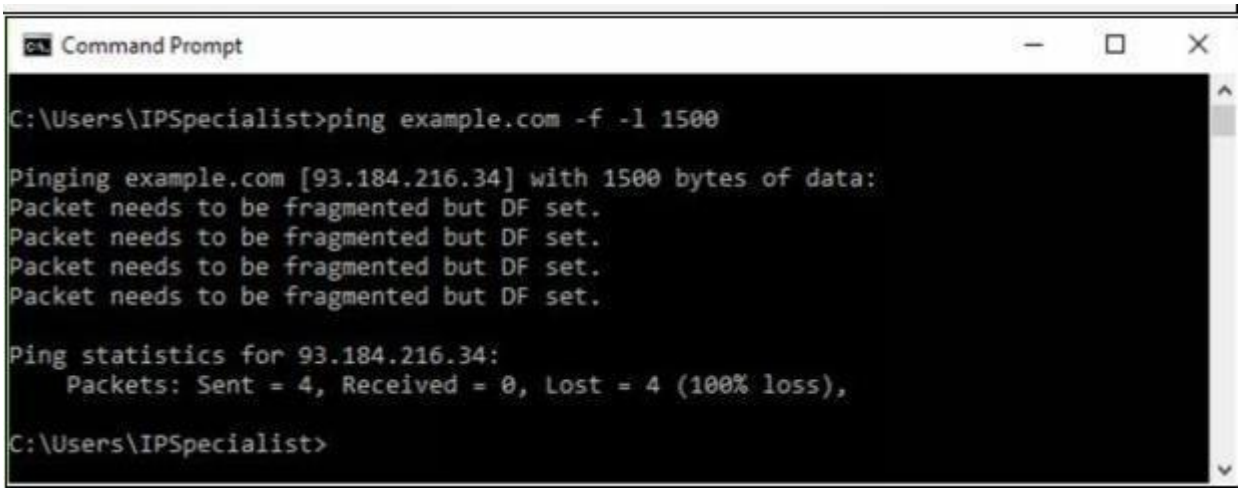
2 -Enter the command " **Ping example.com** " to ping.



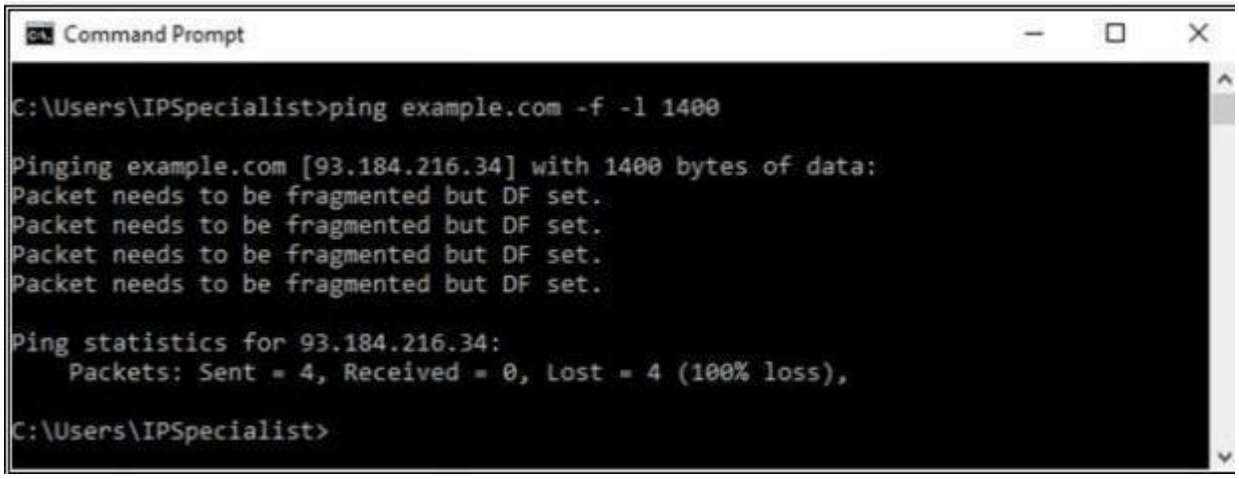
From the output, you can observe and extract the following information:

- Example.com is live
- IP address of example.com.
- Round Trip Time
- TTL value
- Packet loss statistics

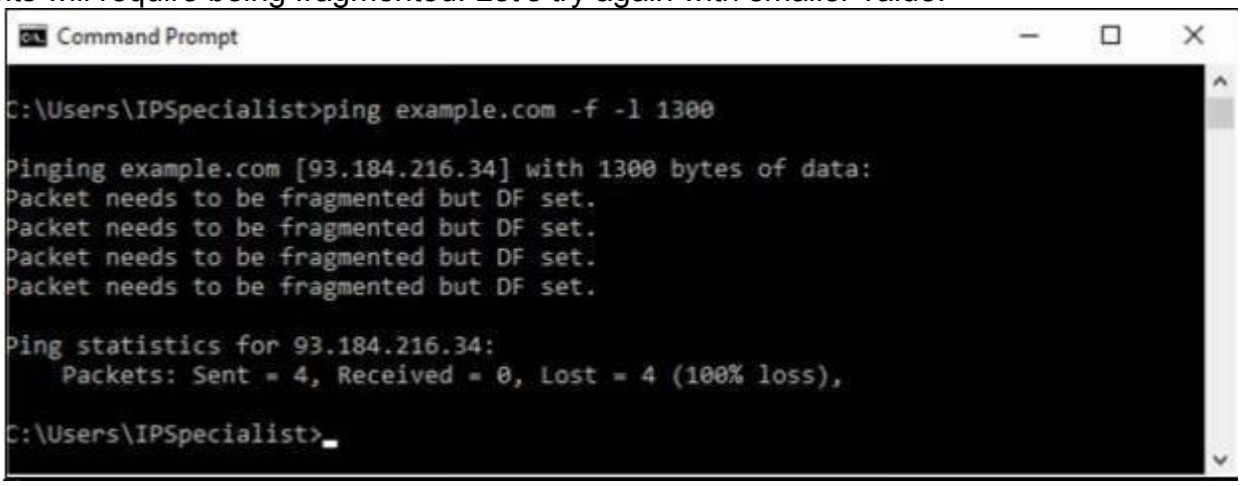
3- Now, Enter the command " Ping example.com –f –l 1500 " to check the value of fragmentation.



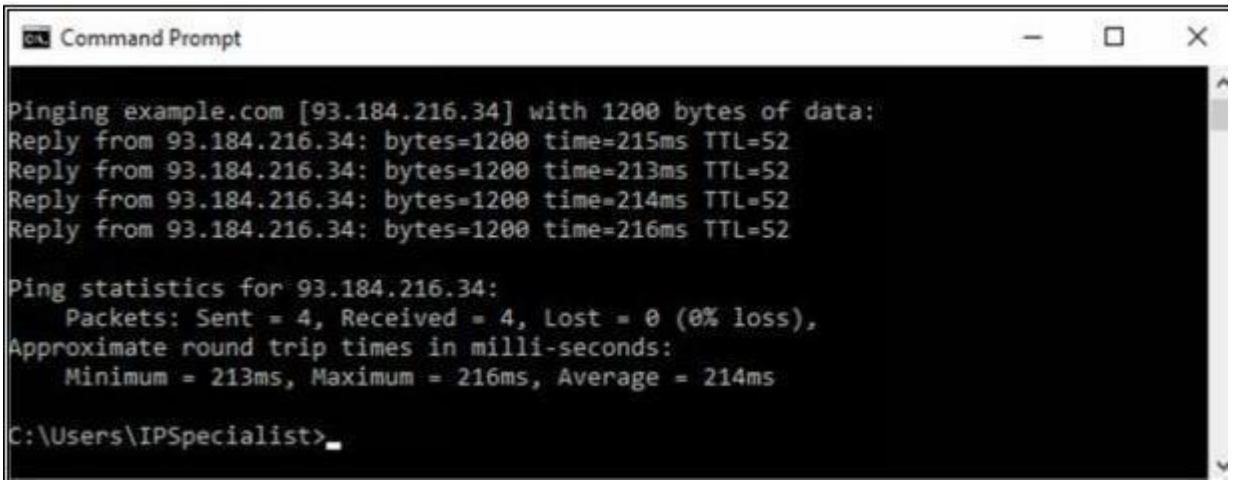
The output shows " **Packet needs to be fragmented but DF set** " which means 150o bits will require being fragmented. Let's try again with smaller value:



Output again shows " **Packet needs to be fragmented but DF set** " which means 140o bits will require being fragmented. Let's try again with smaller value:



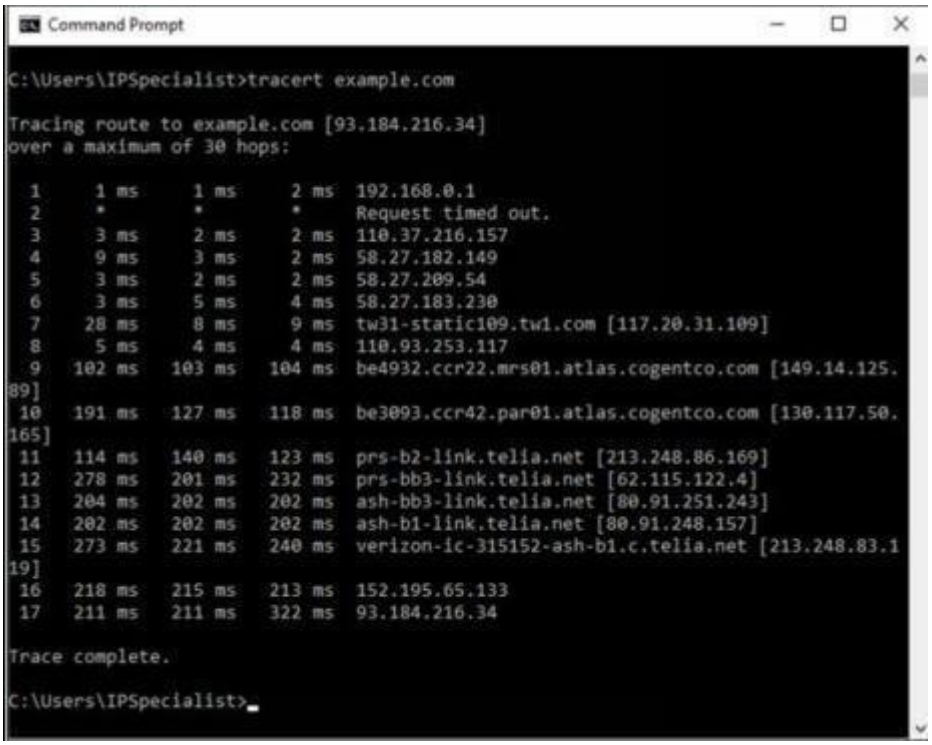
Output again shows " **Packet needs to be fragmented but DF set** " which means 130o bits will require being fragmented. Let's try again with smaller value:



The output shows the reply now, which means 1200 bits will not require being fragmented. You can try again to get the more appropriate fragment value.

- **Tracert using Ping**

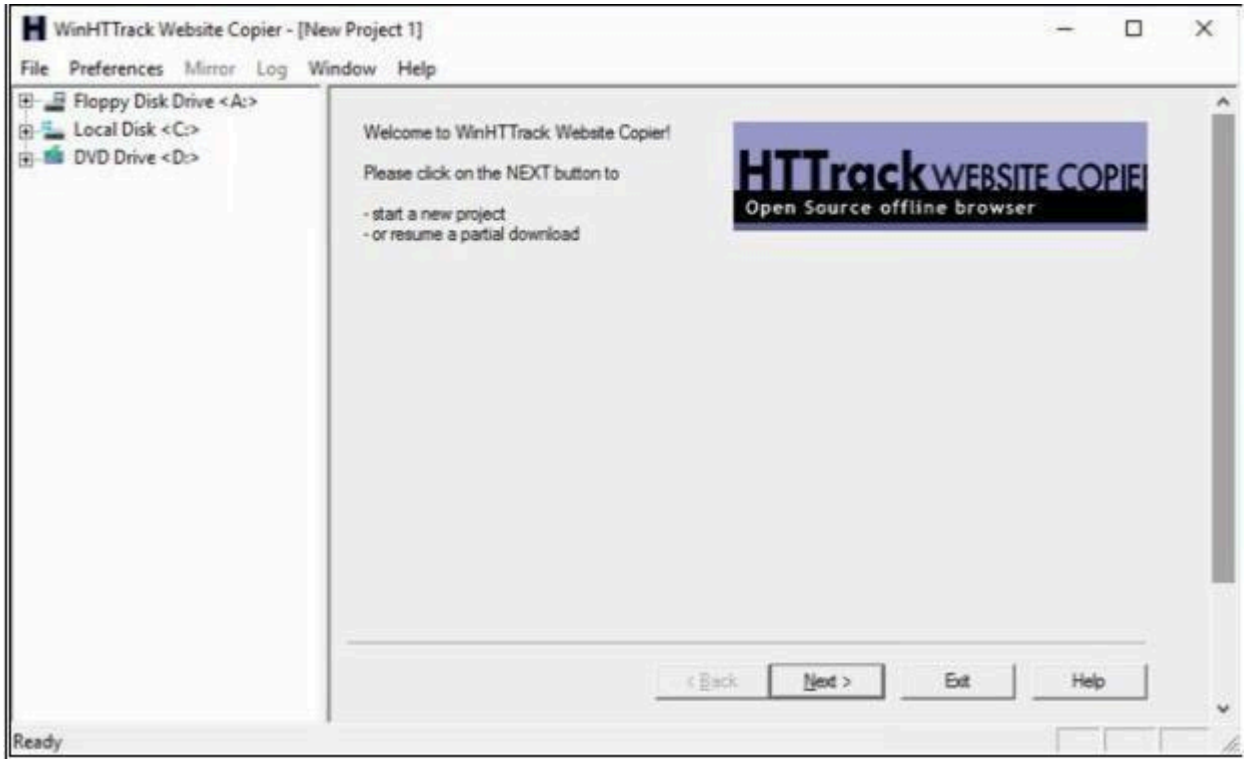
Enter the command " **Tracert example.com** " to trace the target.



From the output, you can get the information about hops between the source (your PC) and the destination (example.com), response times and other information.

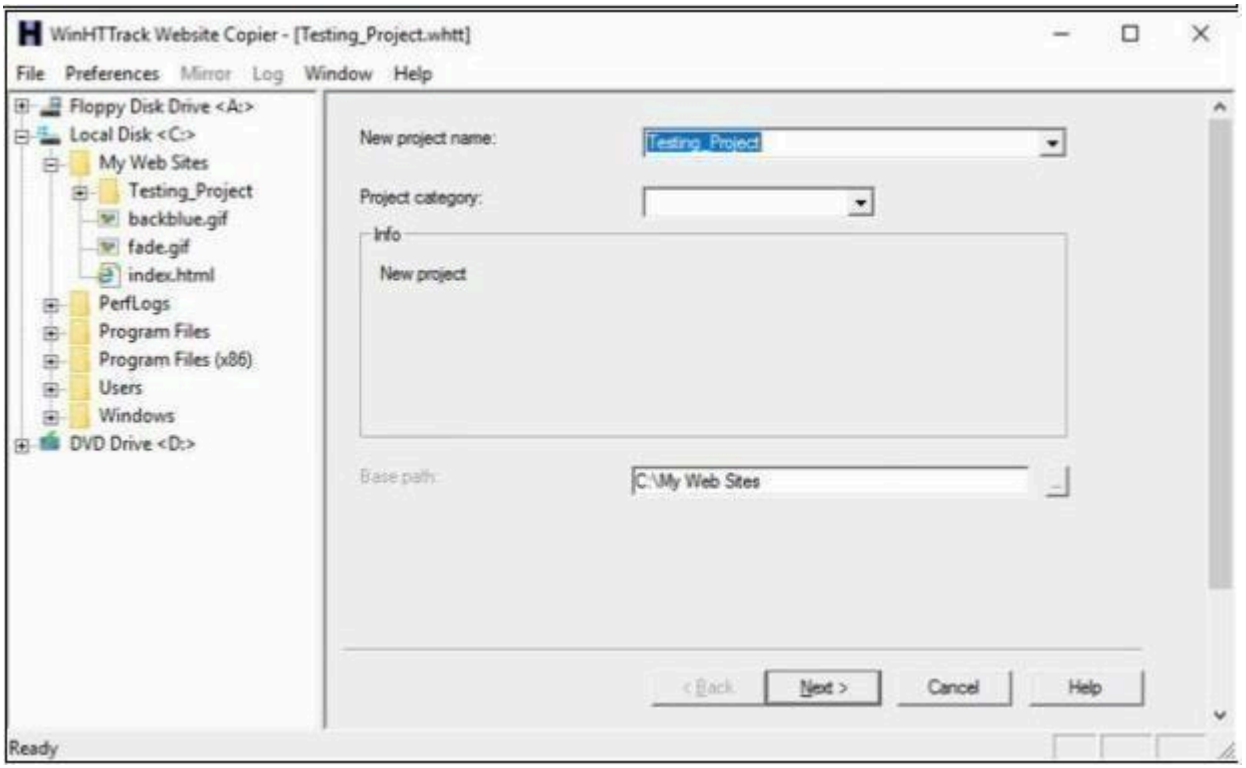
ii. Website Copier tool (HTTrack)

1- Download and Install the WinHTTrack Website Copier Tool from the website http://www.httrack.com. You can check the compatibility of HTTrack Website copier tool on different platforms such as Windows, Linux, and Android from the website.

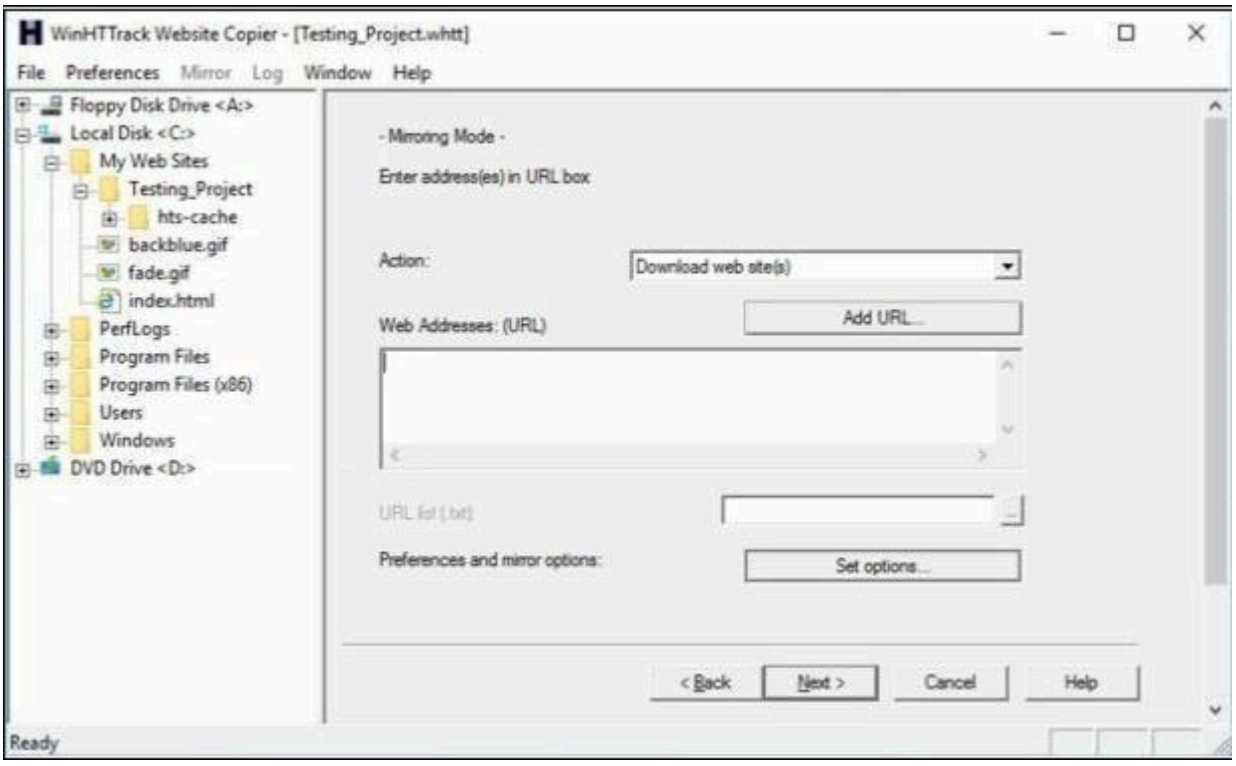


2- HTTrack Website Copier tool installation.

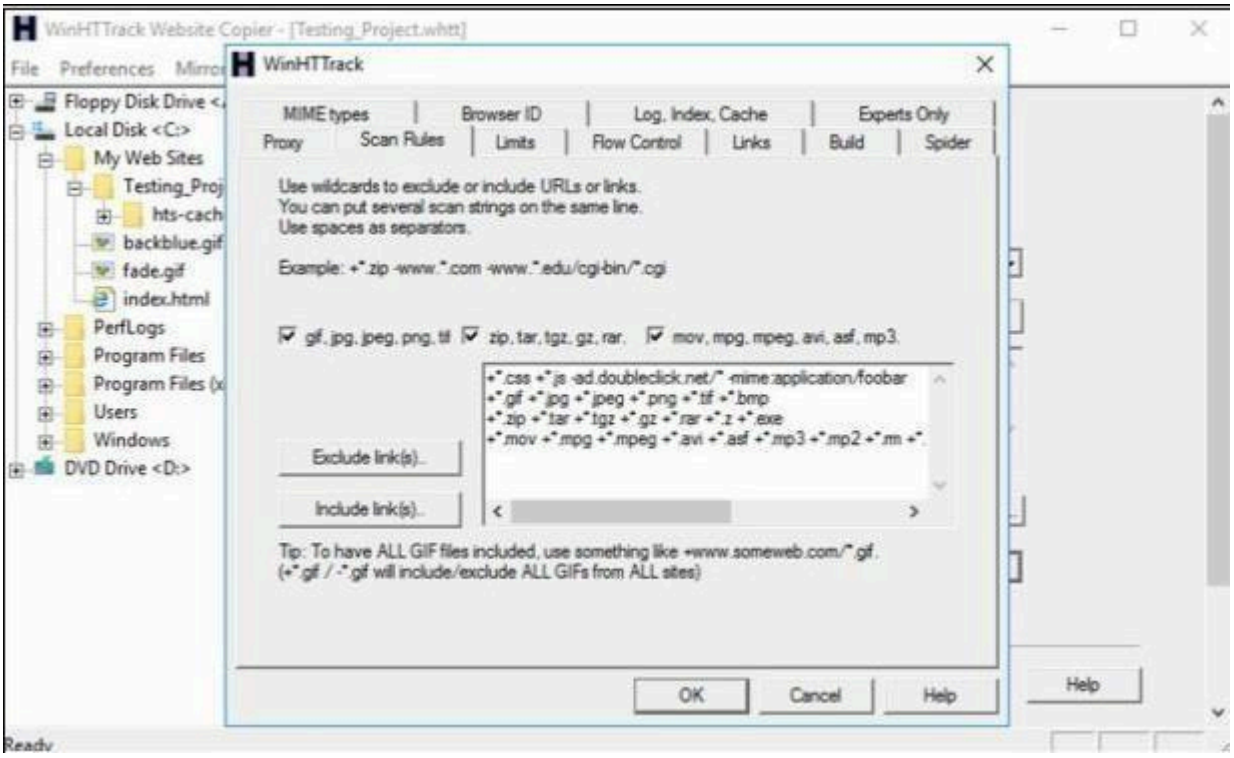
3- Click Next



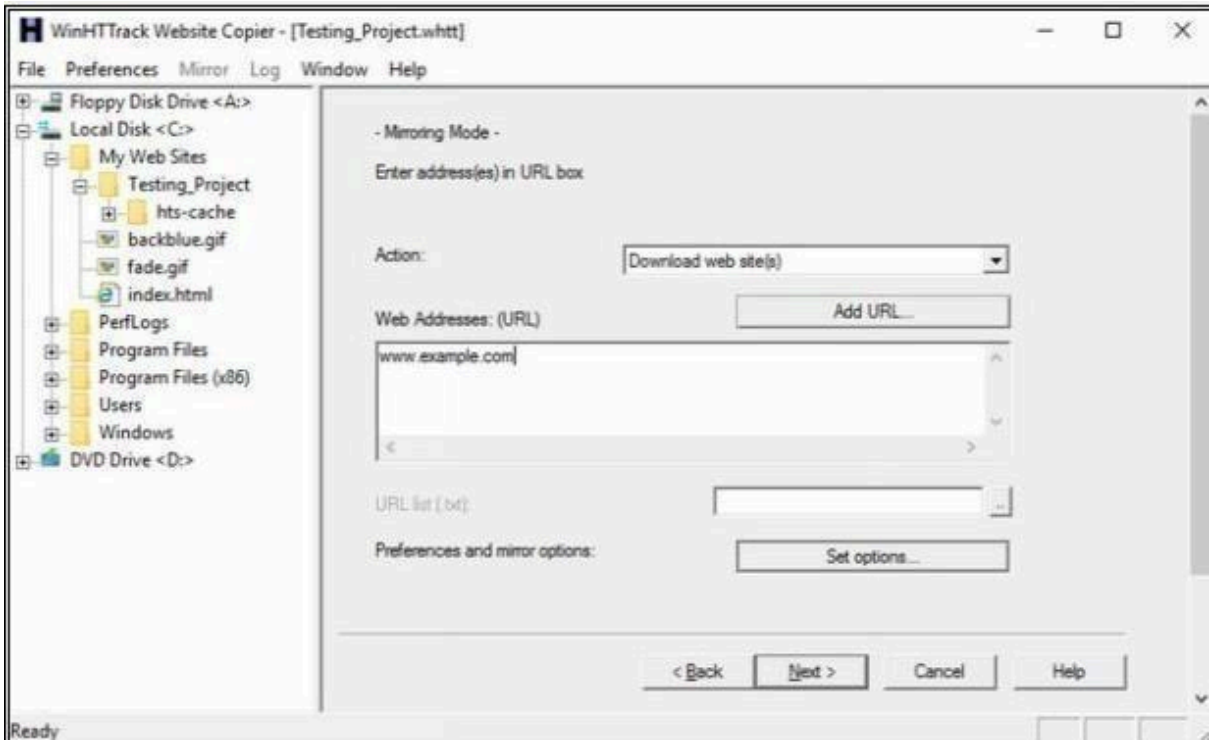
4- Enter a Project name, as in our case, **Testing_Project.**



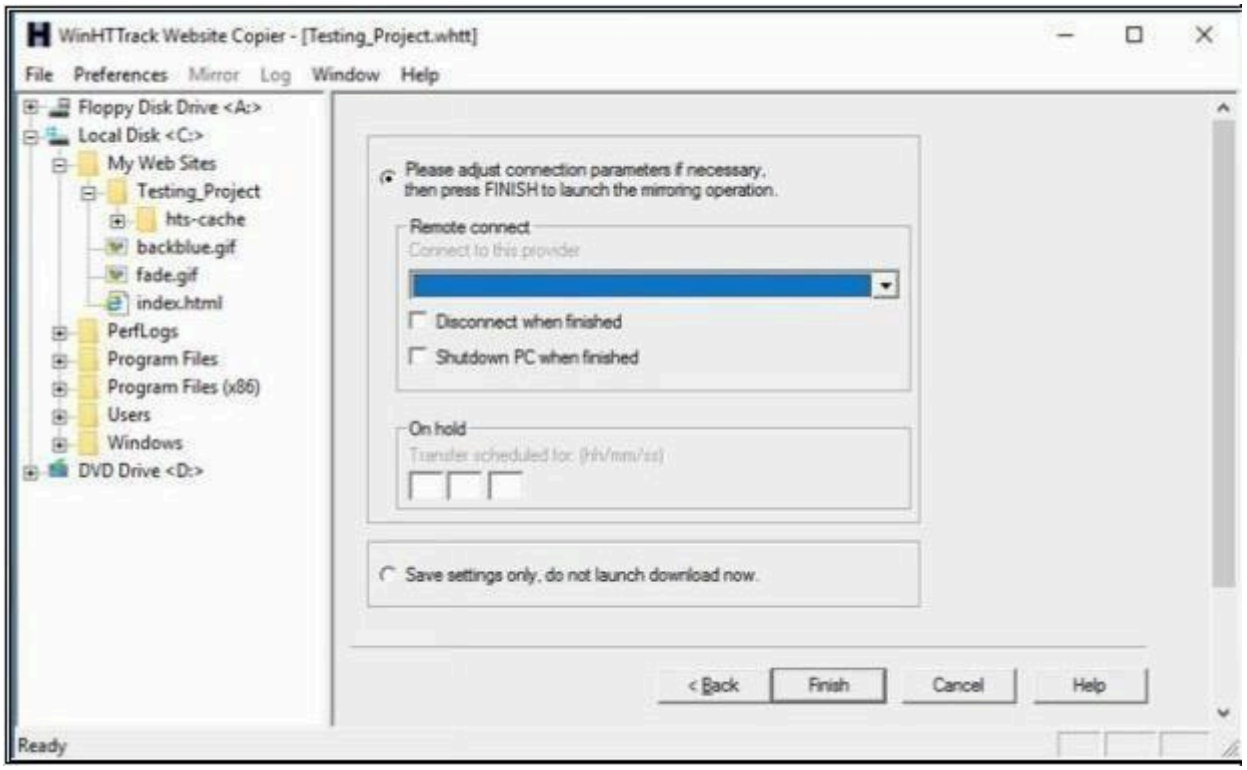
5- Click on **Set Options** button.



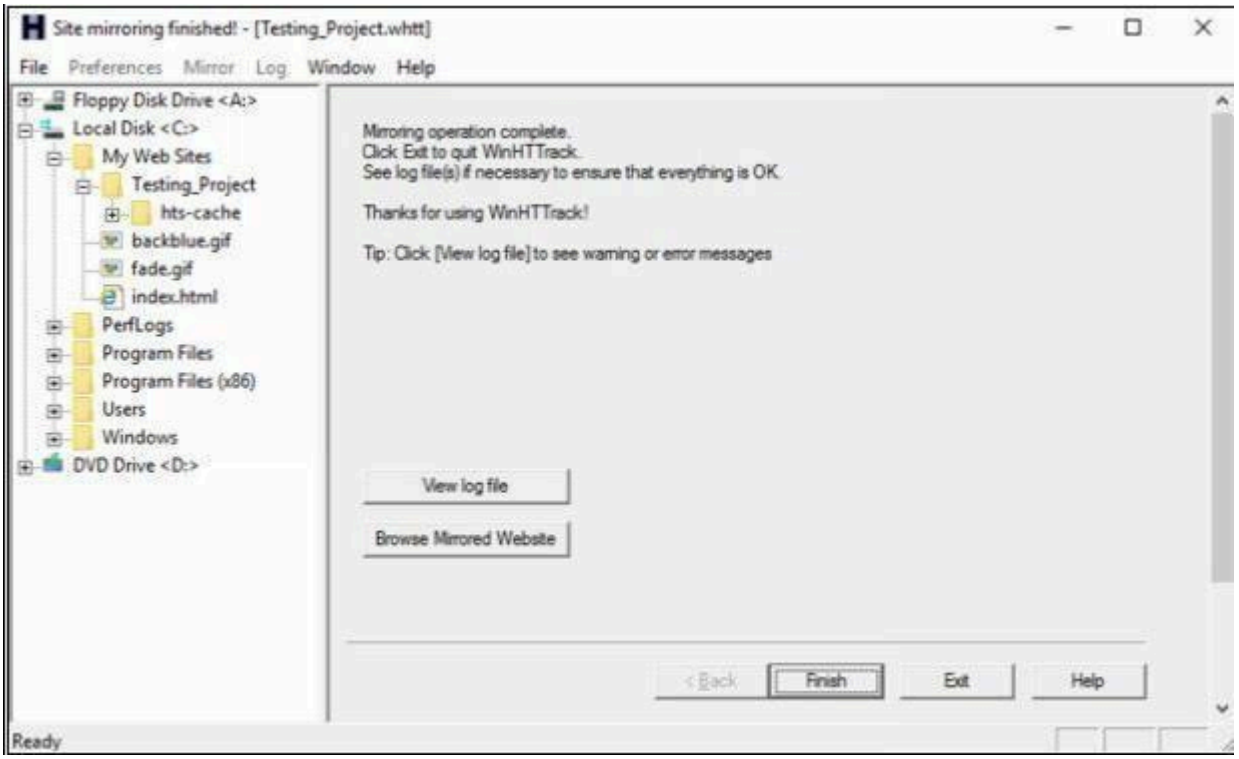
1- Go to **Scan Rules** Tab and Select options as required.



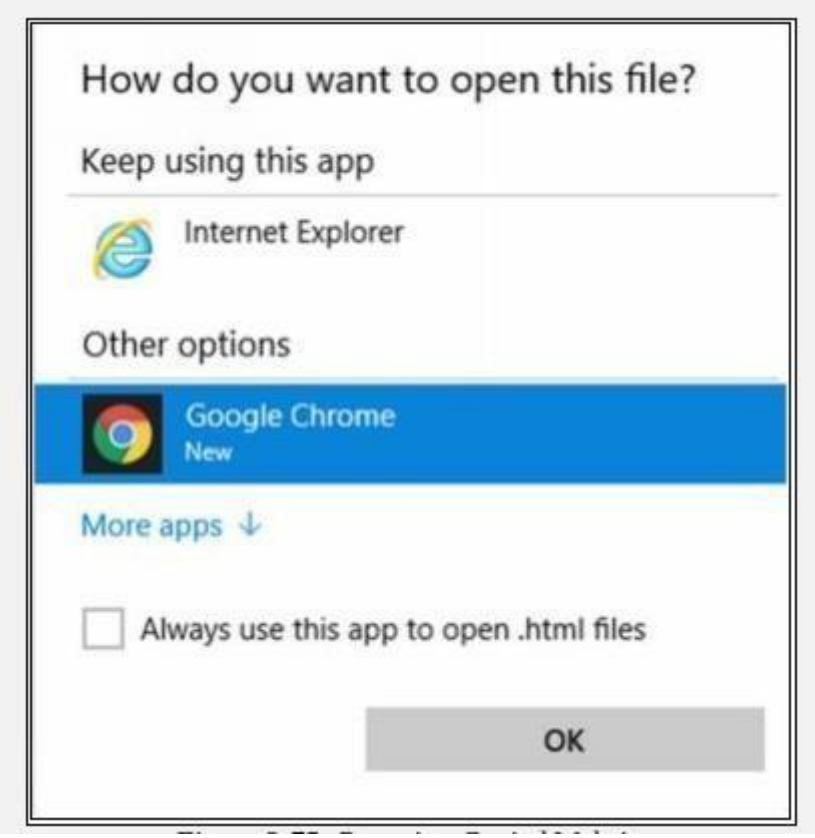
2- Enter the Web Address in the field and Click Next.



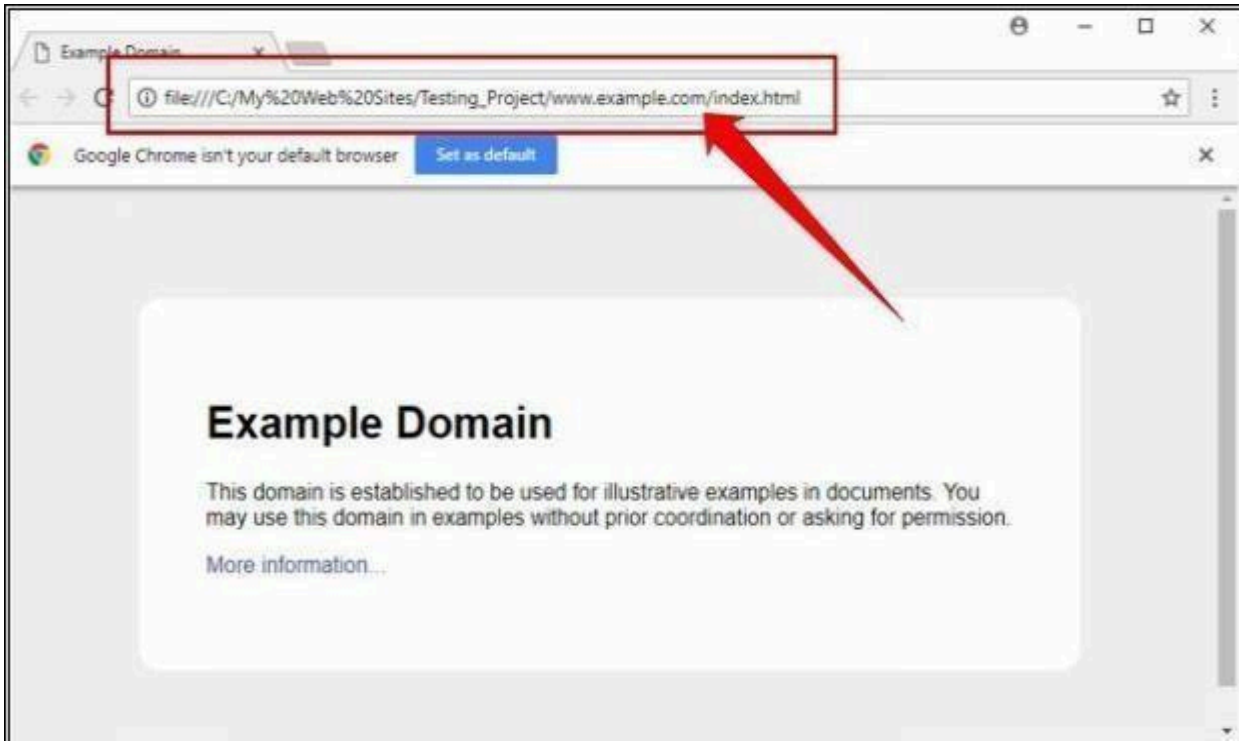
3- Click Next.



4- Click **Browse Mirrored Website**.



5- Select your favorite web browser.

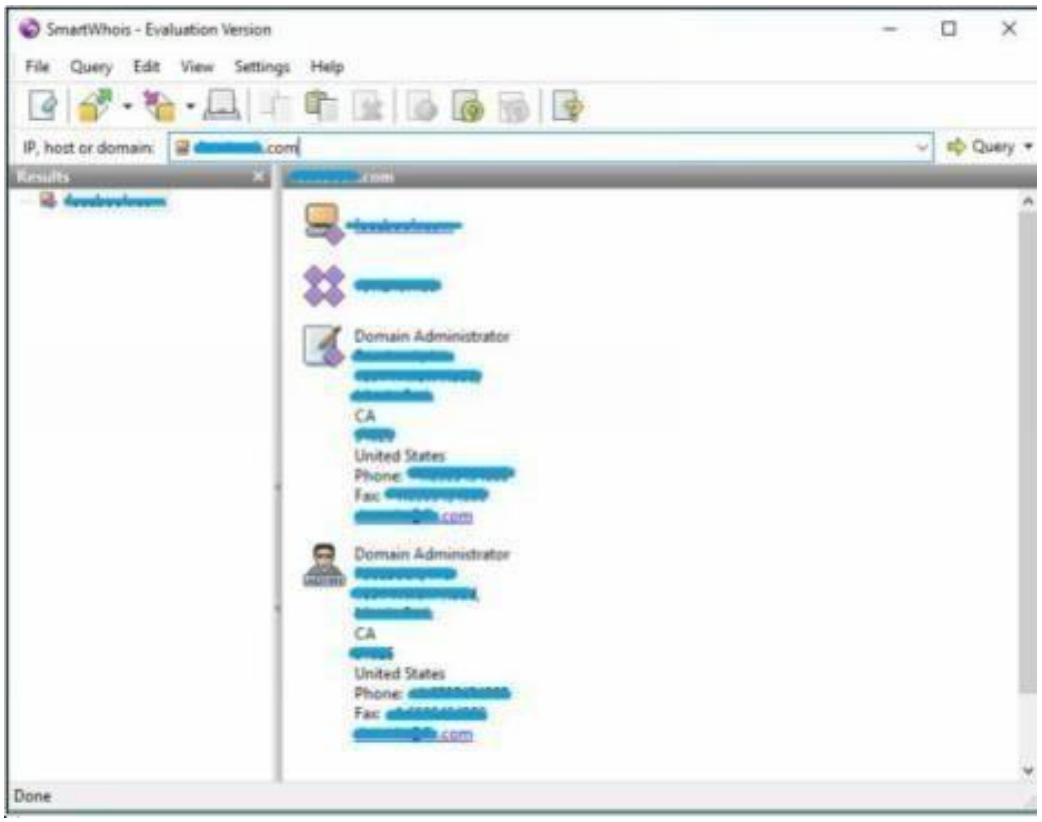


Observed the above output. Example.com website is copied into a local directory and browsed from there. Now you can explore the website in an offline environment for the structure of the website and other parameters.

To make sure, compare the website to the original example.com website. Open a new tab and go to URL example.com.

iii. Smart Whois

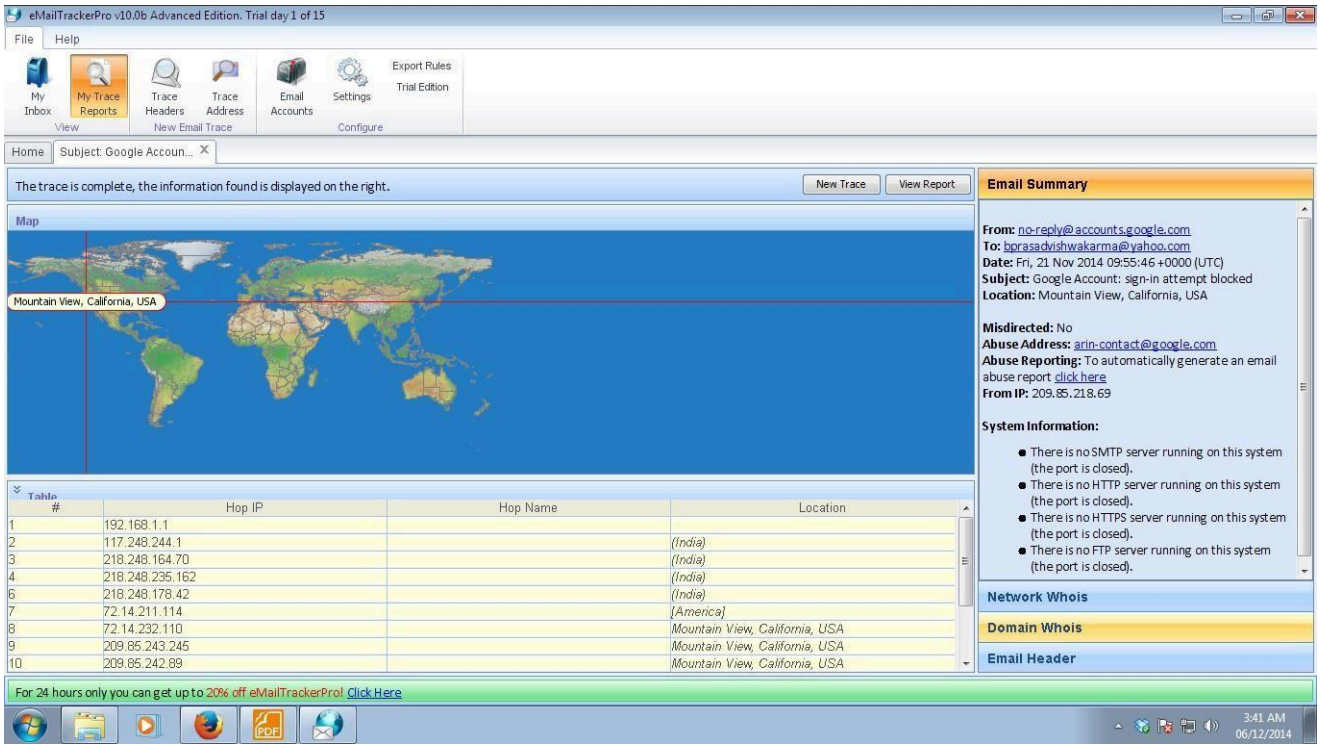
You can download software "***SmartWhois***" from ***www.tamos.com*** for Whois lookup as shown in the figure below: -



iv. eMailTracker Pro

eMailTrackerPro is **a Windows based email tracker that can be used to monitor employees, senders and recipients**. This powerful tool can be used in conjunction with other programs such as Windows Nuke (also known as Spamwasher) to quickly identify where a computer has been and how it has been used.

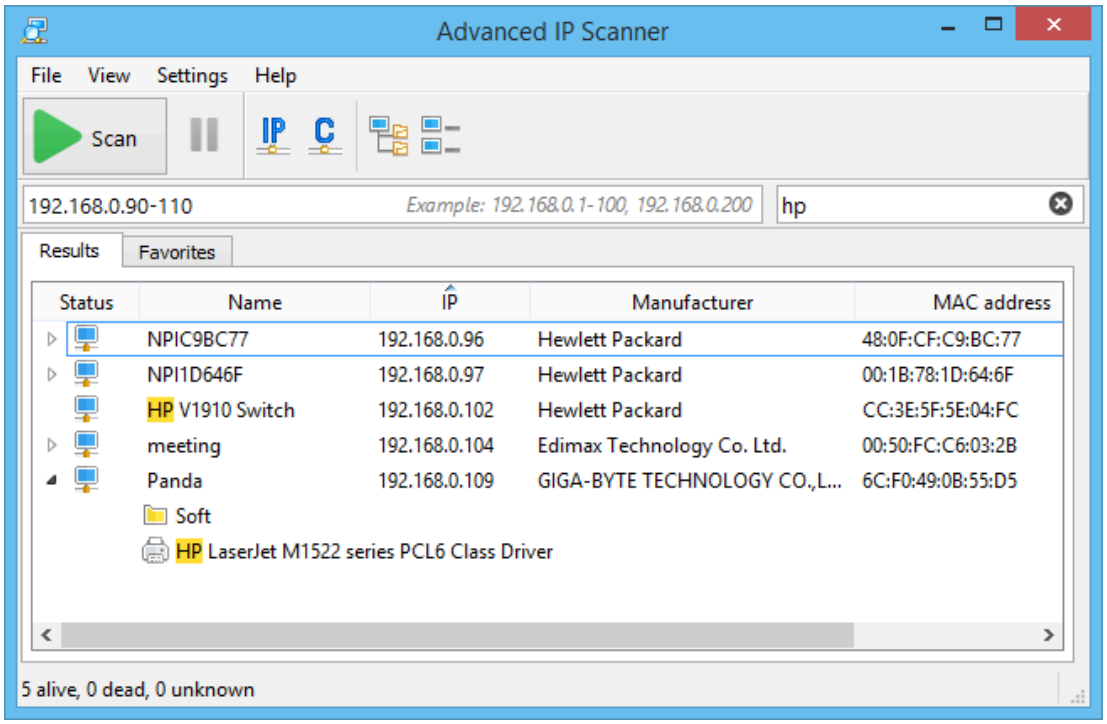
Click on Trace Headers/Trace email address and enter the Message Header and click Okay. The Status of the Trace will be shown inside Trace Reports



b. Scan the network using the following tools:

i.Advanced IP Scanner

Advanced IP Scanner is **a fast and powerful network scanner with a user-friendly interface**. In seconds, Advanced IP Scanner can locate all computers on your wired or wireless local network and scan their ports. The program provides easy access to variou s network resources such as HTTP, HTTPS, FTP, and shared folders.

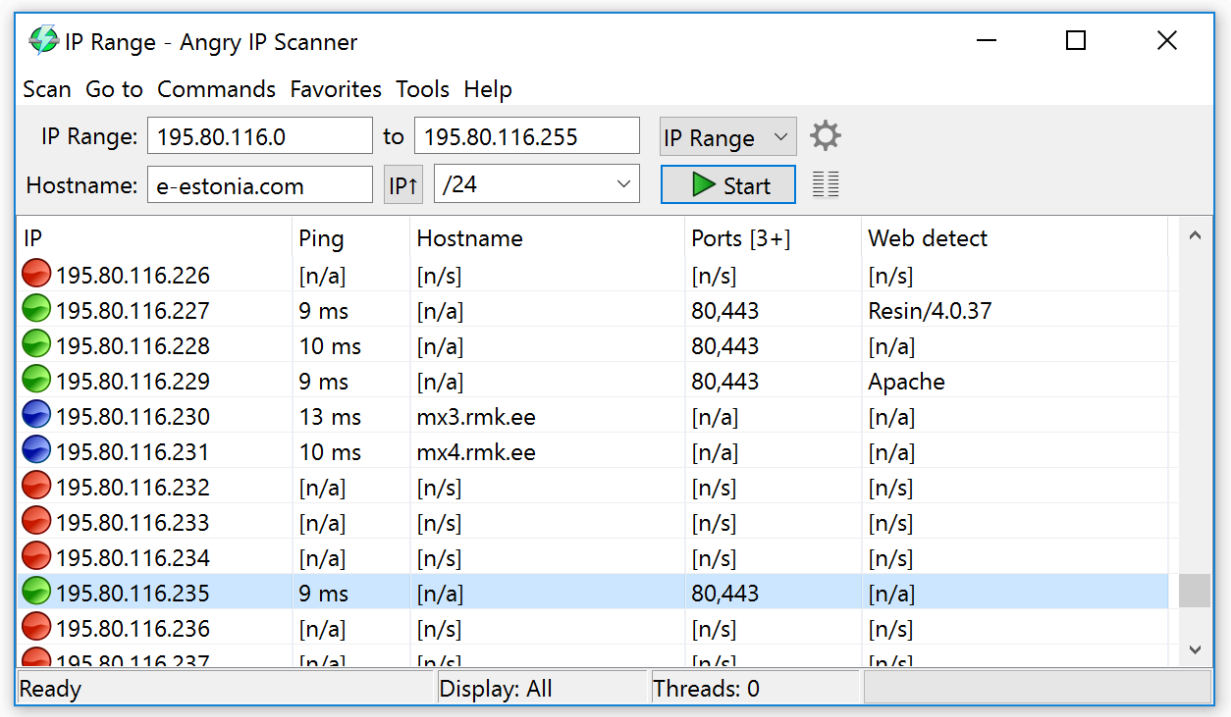


ii. Angry IP Scanner

Angry IP Scanner (or simply ipscan) is an open-source and cross-platform network scanner designed to be fast and simple to use. It scans IP addresses and ports as well as has many other features.

It is widely used by network administrators and just curious users around the world, including large and small enterprises, banks, and government agencies.

It runs on Linux, Windows, and Mac OS X, possibly supporting other platforms as well.



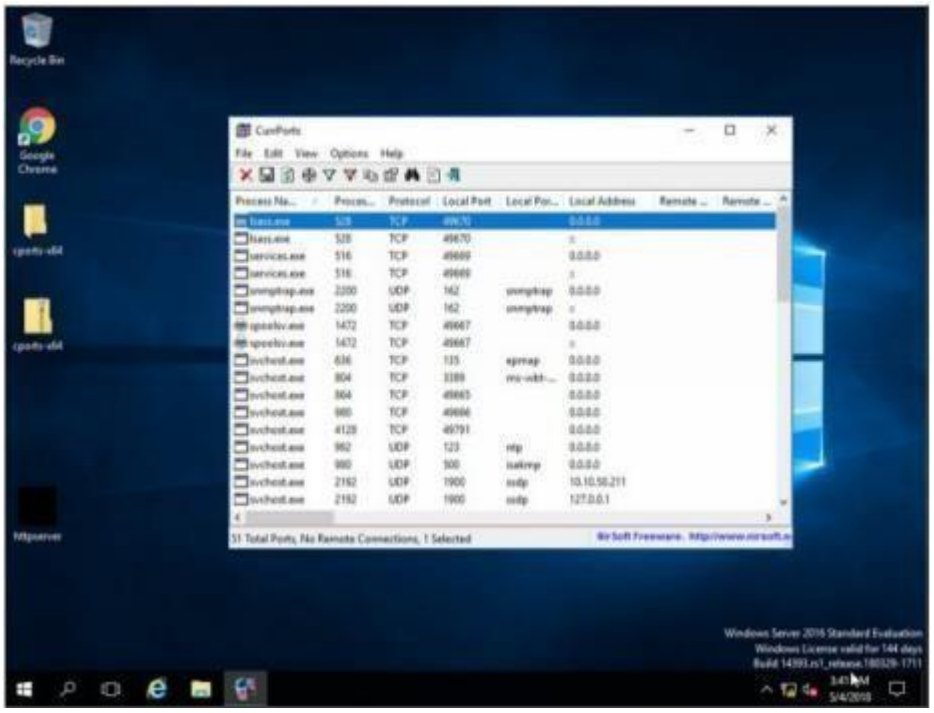
iii. CurrPorts

Case Study: Using the Previous lab, we are going to re-execute HTTP Remote Access Trojan (RAT) on Windows 12 machine (10.10.50.211) and observed the TCP/IP connections to detect and kill the connection.

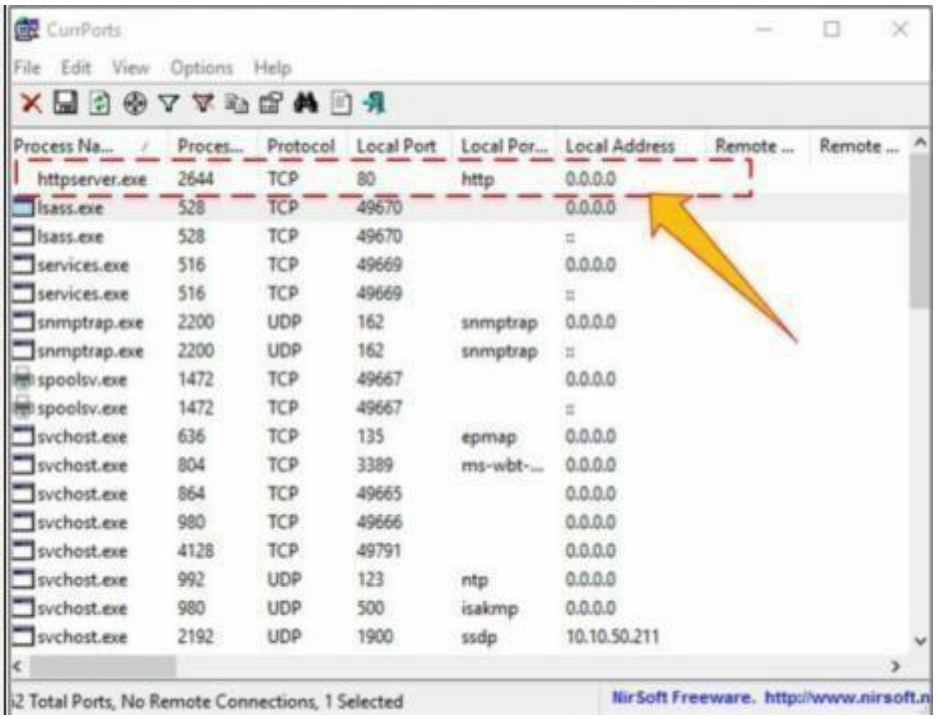
Topology:

Configuration:

1. Run the application **Currports** on Windows Server 2016 and observe theprocesses.



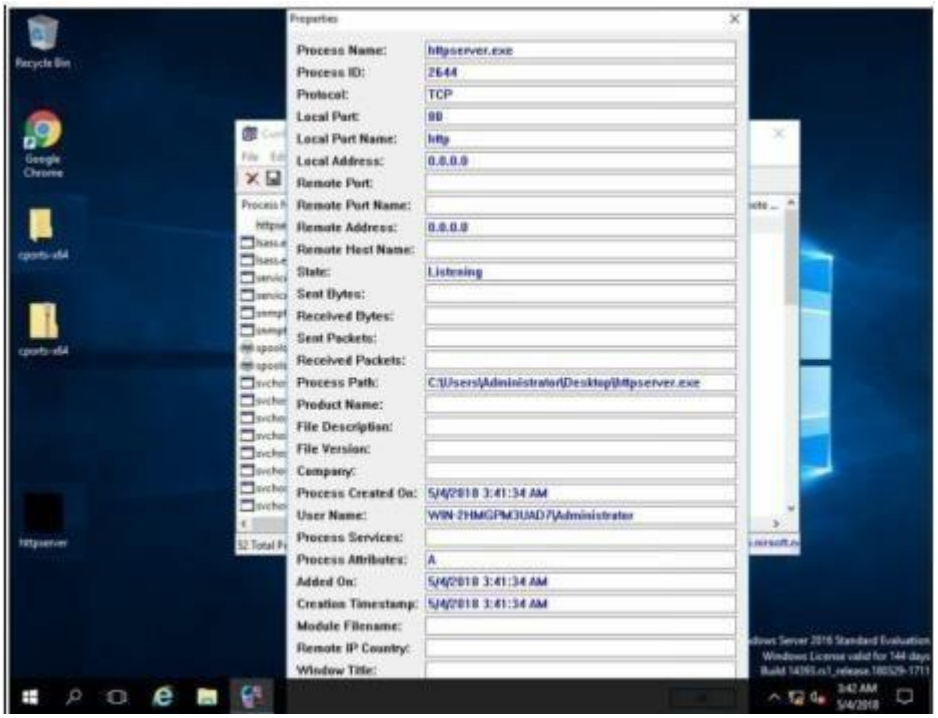
2. Run the HTTP Trojan created in the previous lab



The new process is added to the list.

You can observe the process name, Protocol, Local and remote port and IP address information.

3. For more detail, right click on httpserver.exe and go to properties

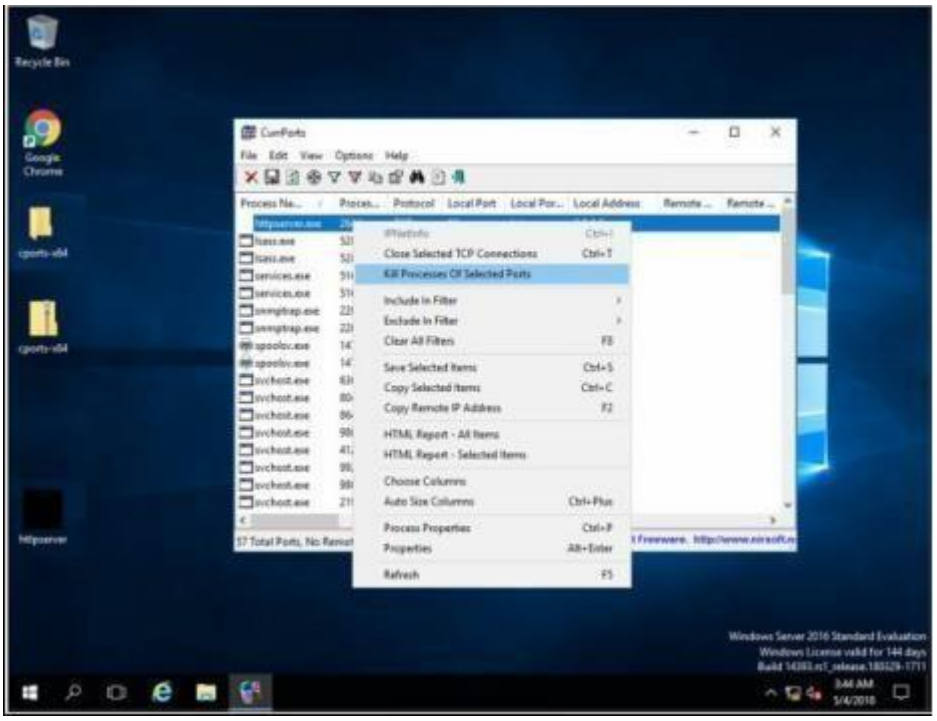


Properties are showing more details about tcp connection.

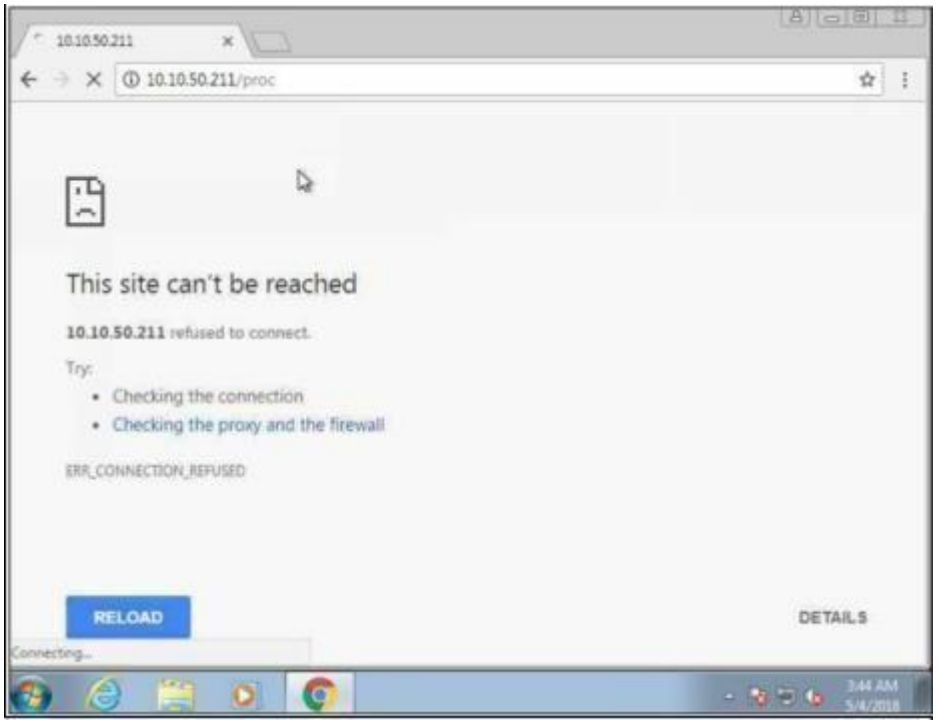
4. Go to Windows 7 machine and initiate the connection as mentioned in the previous lab using a web browser.

Connection successfully established.

5. Back to Windows Server 2016, Kill the connection.

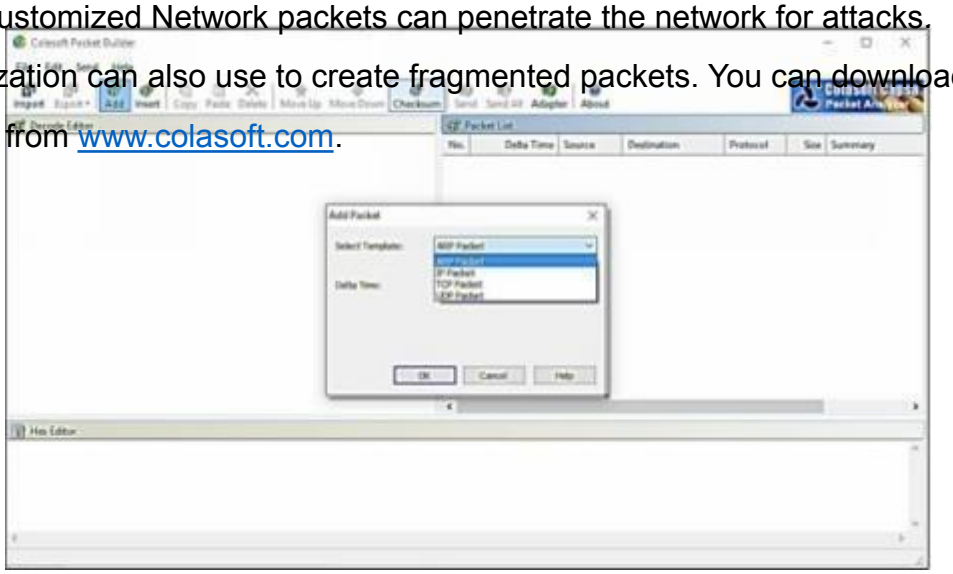


6. To verify, retry to establish the connection from windows 7.



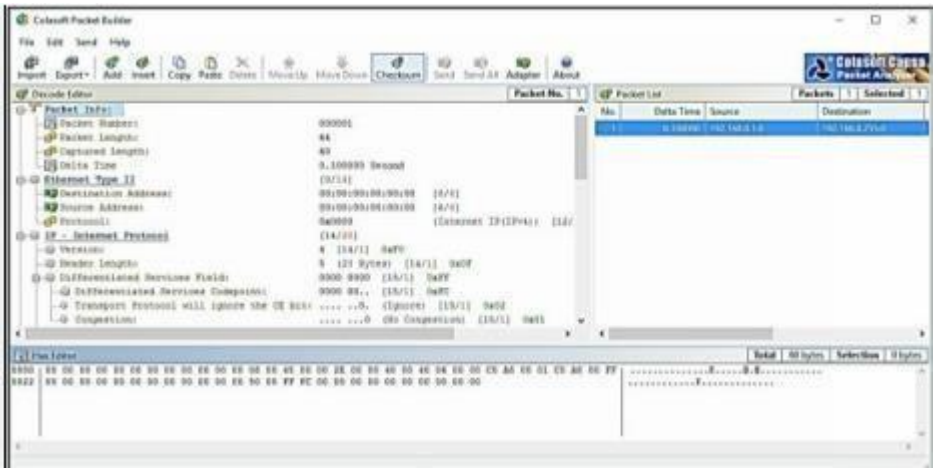
iv. Colasoft Packet Builder

Colasoft Packet Builder software enables to create the customized network packets. These Customized Network packets can penetrate the network for attacks. Customization can also use to create fragmented packets. You can download the software from www.colasoft.com.



Colasoft packet builder offers Import and Export options for a set of packets. You can also add a new packet by clicking **Add**/button. Select the Packet type from the drop-down option. Available options are: -

- ARP Packet
- IP Packet
- TCP Packet
- UDP Packet



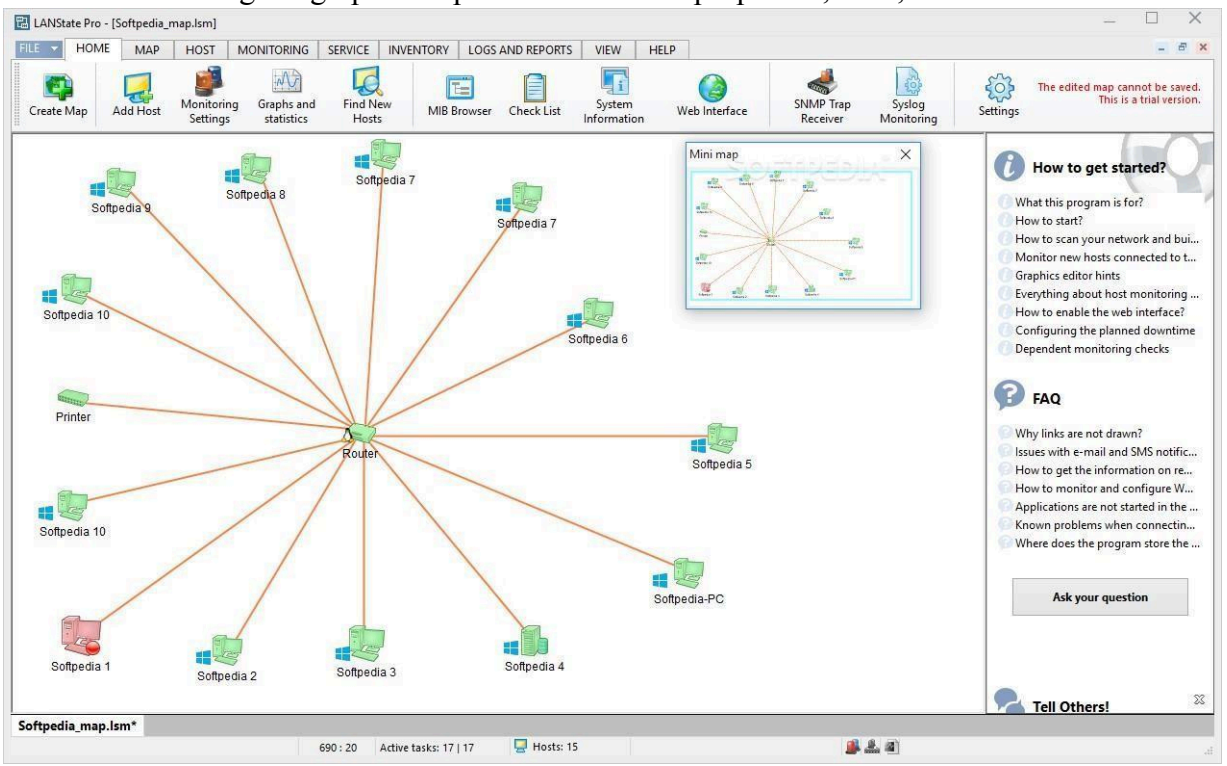
After Selecting the Packet Type, now you can customize the packet, Select the Network Adapter and Send it towards the destination.

Practical No. 2

a. Perform Network Discovery using the following tools:

i. LANState Pro

LANState is **a simple network topology mapping, host monitoring, and management program**. Monitor the service availability. Manage servers, computers, switches, and other devices easier using the graphic map. Access devices' properties, RDP, web UI faster.

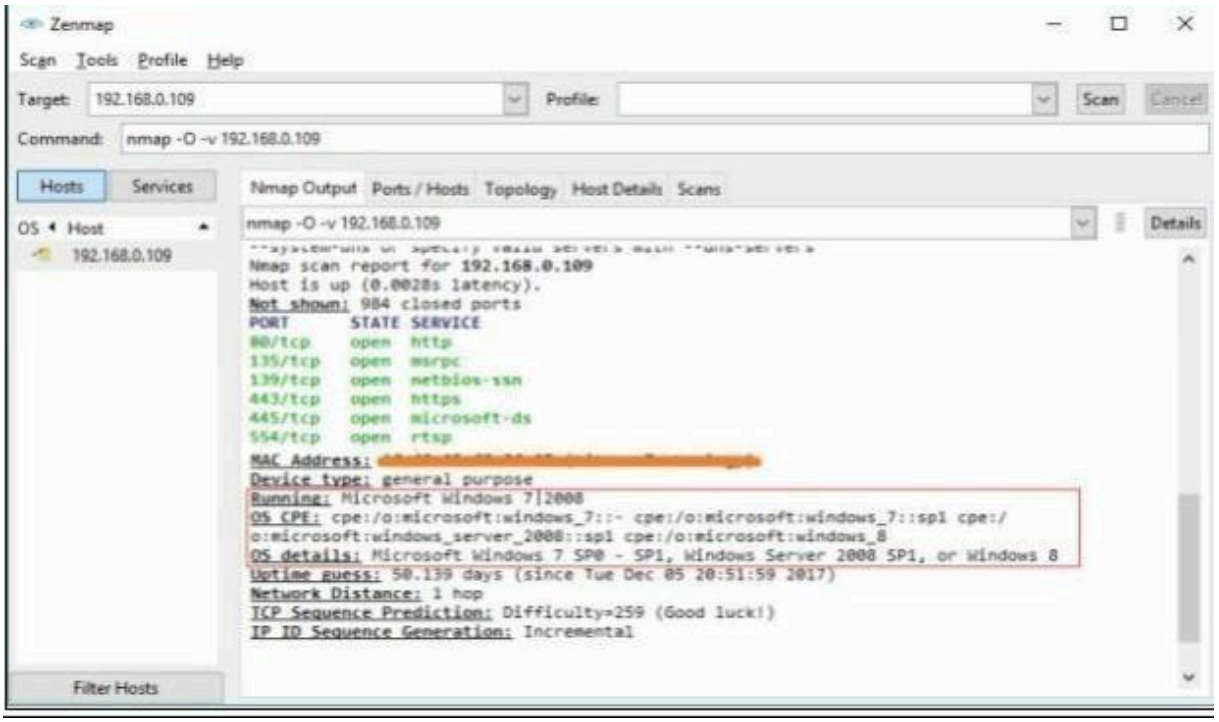


Practical No. 3

a. Perform Enumeration using the following tools:

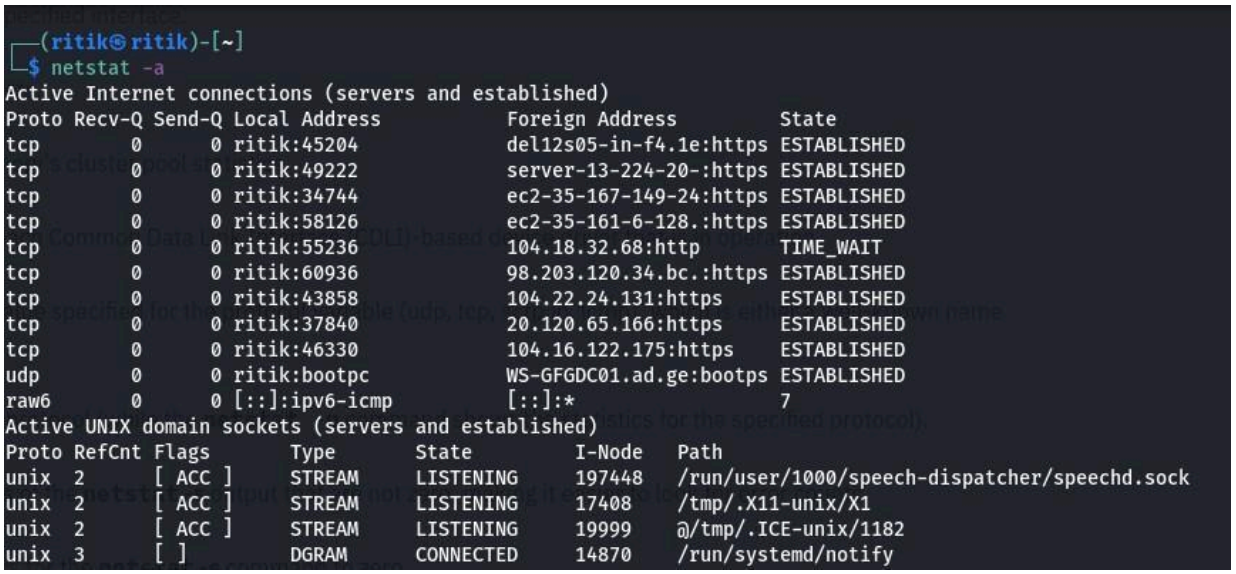
i.Nmap

NMAP, as we know, is a powerful networking tool which supports many features and commands. Operating System detection capability allows to send TCP and UDP packet and observe the response from the targeted host. A detailed assessment of this response bring some clues regarding nature of an operating system disclosing the type an OS. To perform OS detection with nmap perform the following: nmap -O<ip address>



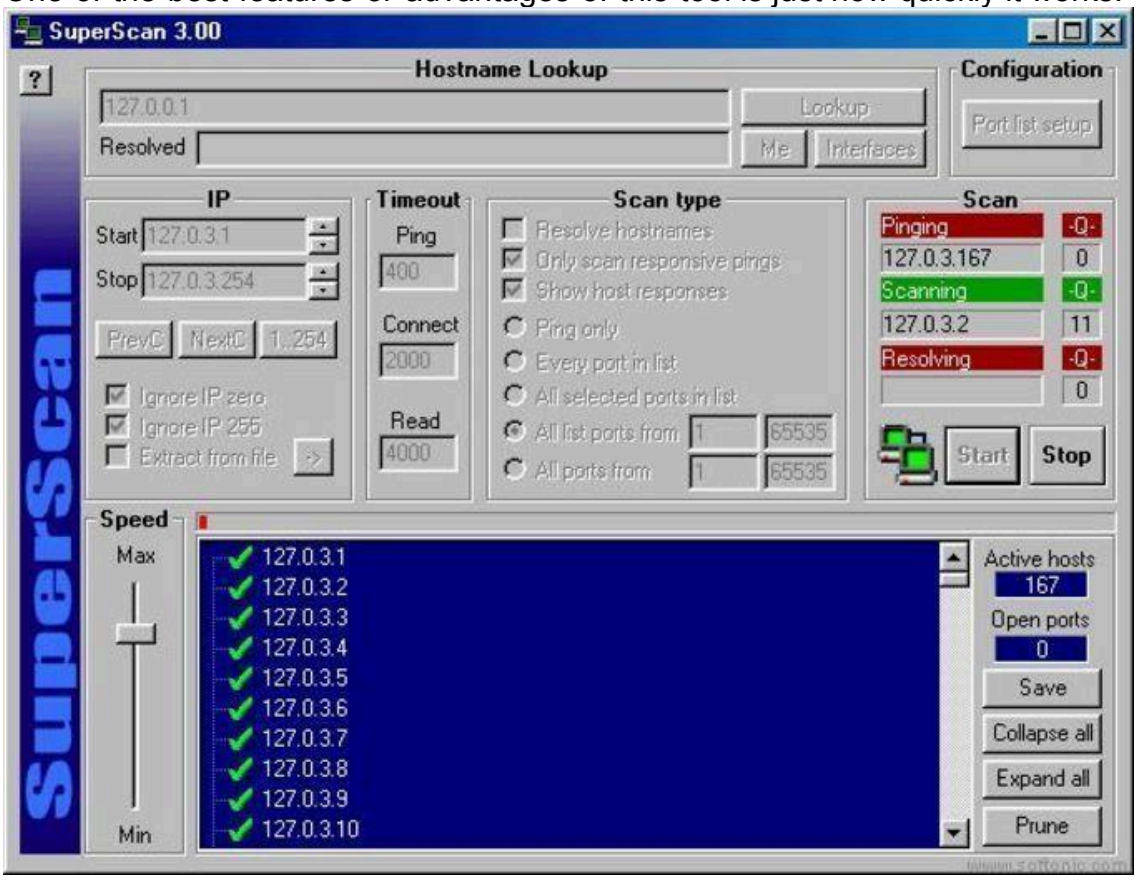
ii. NetBIOS Enumeration Tool

NetBIOS stands for Network Basic Input Output System. It **Allows computer communication over a LAN and allows them to share files and printers**. NetBIOS names are used to identify network devices over TCP/IP (Windows).



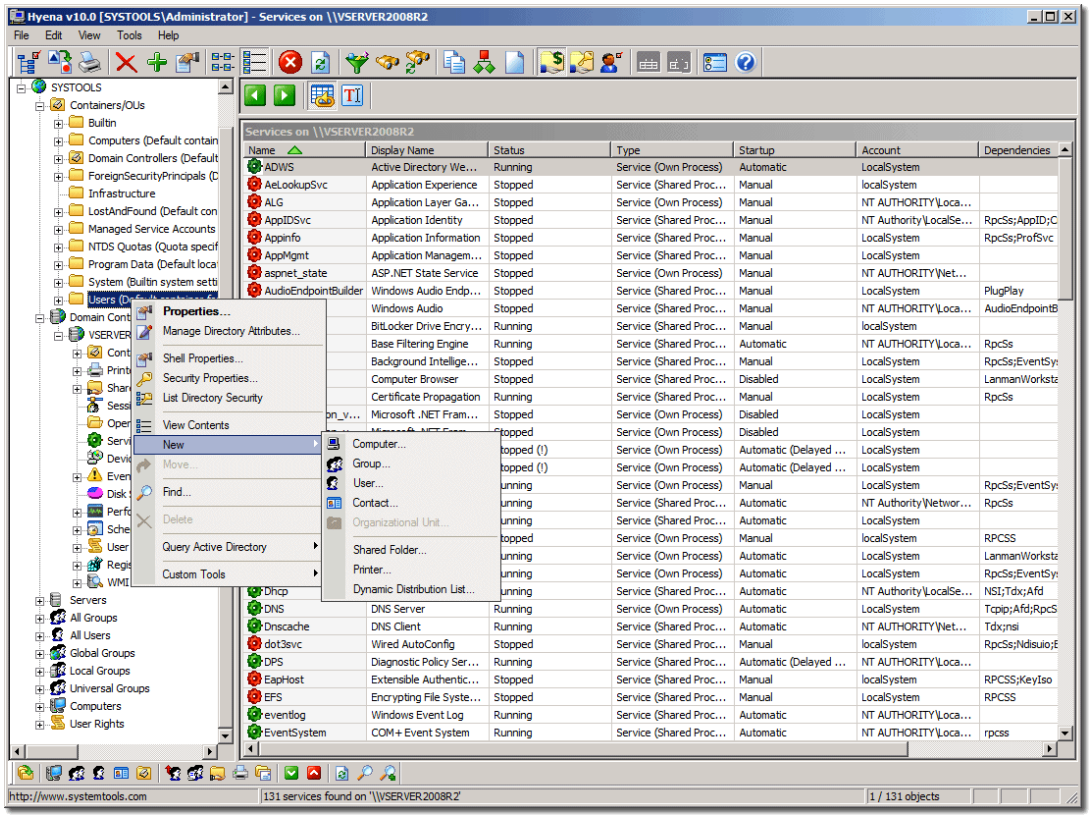
iii. SuperScan

SuperScan is a multi-functional tool that will help you manage your network and make sure your connections and TCP ports are working as well as they should be. One of the best features or advantages of this tool is just how quickly it works. The out



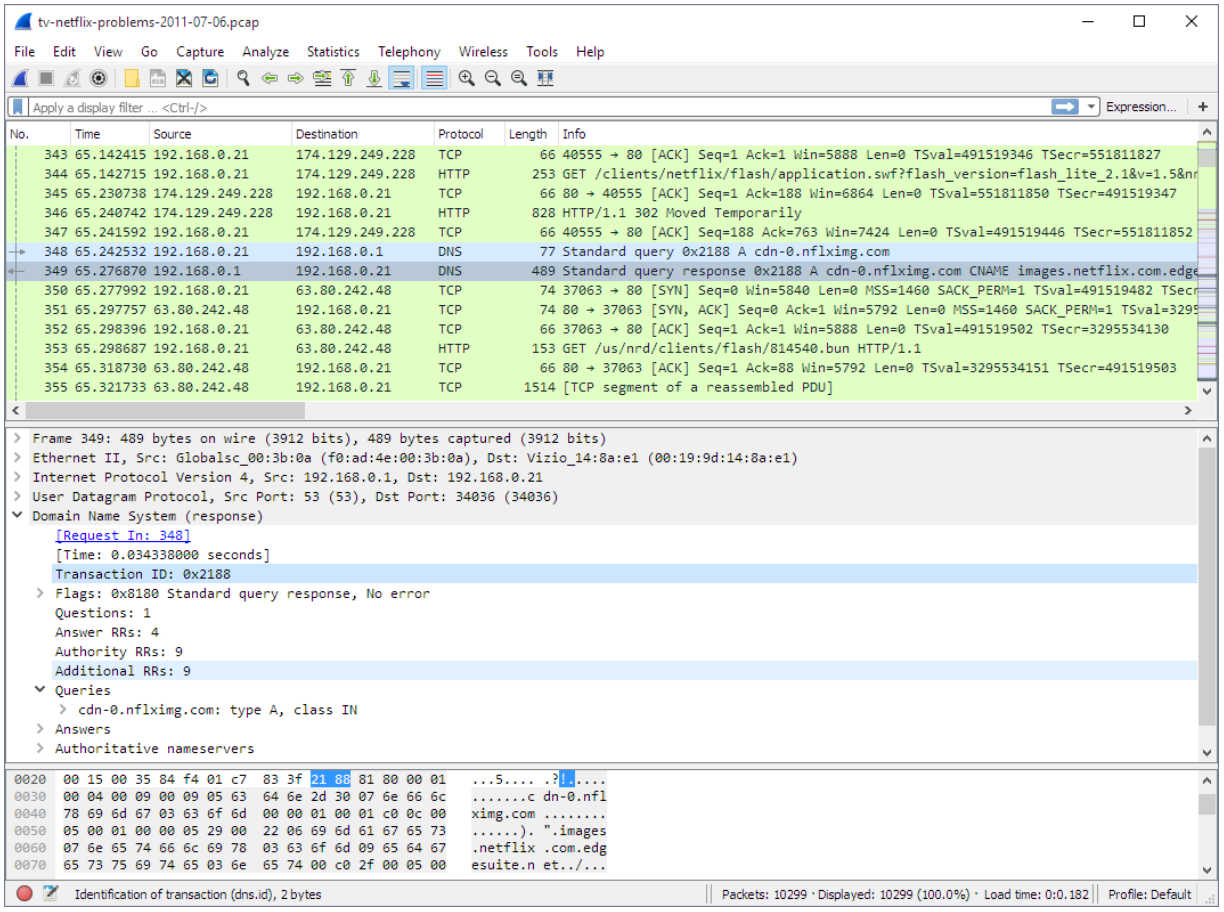
iv. Hyena

Hyena is GUI based, NetBIOS Enumeration tool that shows Shares, User login information and other related information



viii. Wireshark

Wireshark is a free and open-source packet analyzer. It is used for network troubleshooting, analysis, software and communications protocol development, and education. Originally named Ethereal, the project was renamed Wireshark in May 2006 due to trademark issues



Practical No. 4

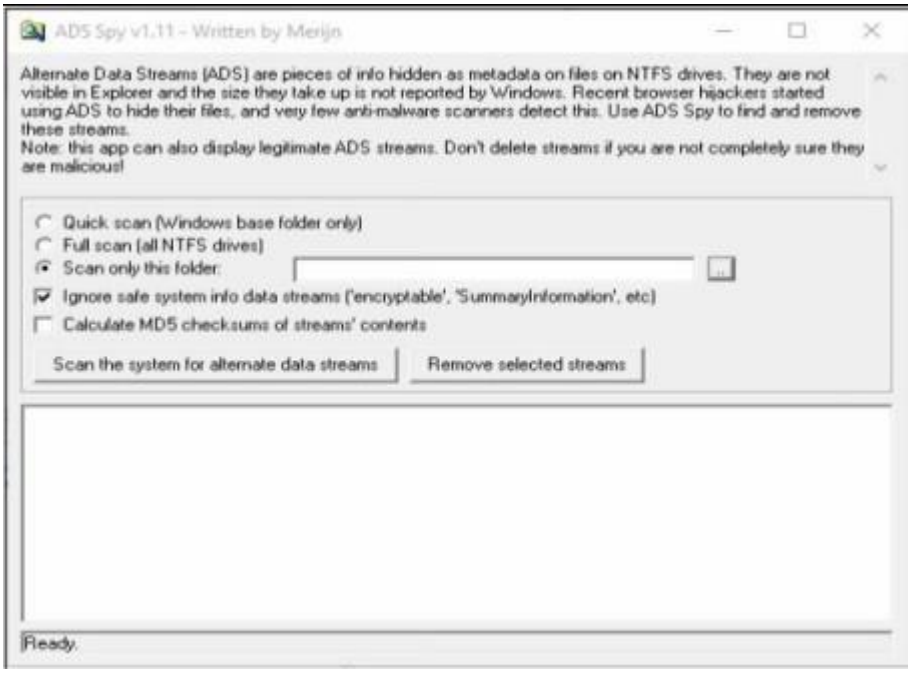
a. Perform the System Hacking using the following tools:

i.ADS Spy

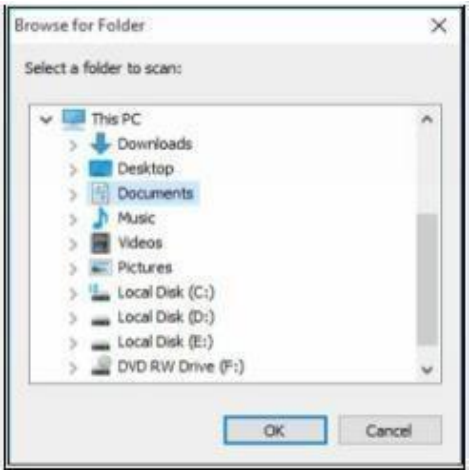
AdSpy offers the most search options of any Ad Intelligence Tool, so you can find the data you want, how you want. Search in the usual way: ad text, URL, page name. Search true data from user reactions in advert comments. Be as rigorous as you need to: search or filter by affiliate network, affiliate ID, Offer ID, landing page technologies - whatever helps you find the information you can work with. Open ADS

Spy application and select the option if you want to:

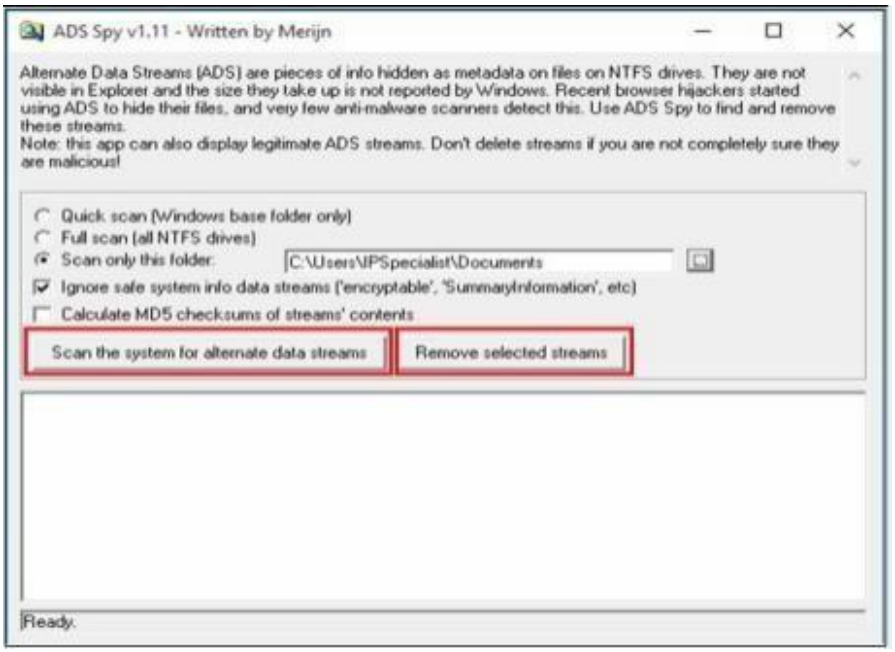
- Quick Scan
- Full Scan
- Scan Specific Folder



As we store the file in the Document folder, Selecting Document folder to scan particular folder only.



Select an Option, if you want to scan for ADS, click "**Scan the system for ADS**"/ or click **removes**



button to remove the file

As shown in the figure below, ADS Spy has detected the **Testfile.txt:hidden.txt** file from the directory.

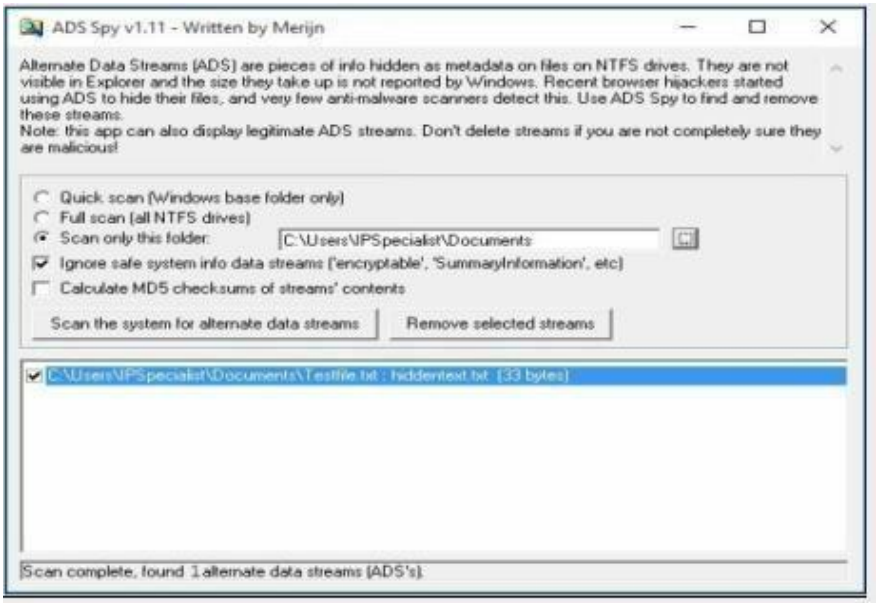


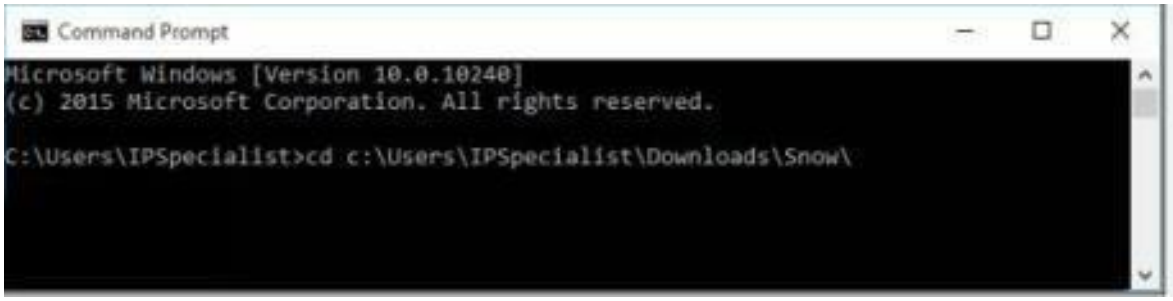
Figure 6-50 ADS Detection

ii. Snow

Create a text file with some data in the same directory where Snow Tool is installed.

Go to Command Prompt

Change the directory to run Snow tool



Type the command

Snow –C –m “text to be hide” –p “password” <Sourcefile> <Destinationfile>

The source file is a Hello.txt file as shown above. Destination file will be the exact copy of source file containing hidden information.

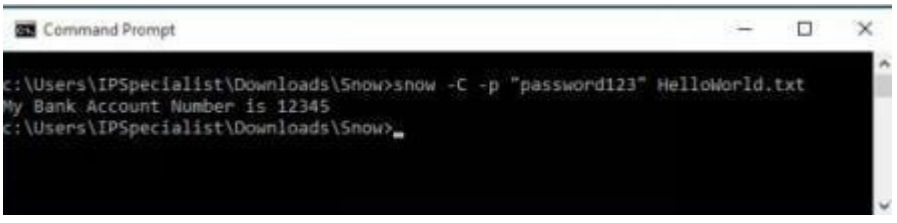
Go to the directory; you will a new file **HelloWorld.txt**. Open the File

New File has the same text as an original file without any hidden information. This file can be sent to the target.

Recovering Hidden Information

On destination, Receiver can reveal information by using the command

Snow –C –p “password123” HelloWorld.txt



As shown in the above figure, File decrypted, showing hidden information encrypted in the previous section

Practical No. 5

a. Use SMAC for MAC Spoofing.

SMAC is a MAC address changer that has a simple-to-use graphical interface that enables the less experienced user all the way up to the guru to change a piece of hardware's MAC address. The less experienced user will appreciate the random generator whereas the guru will appreciate the ability to hand enter a new MAC address

Once it is installed, you will find the application launcher in a Start Menu subdirectory called KLC. Click on that folder and you will see SMAC 2.0. Click on that launcher and the SMAC main window (**Figure A**) will open.

Using SMAC can be very simple, depending on how you want to use it. The simplest way to use SMAC is to assign a random MAC address to a piece of hardware. Before we actually assign a new address, let's take a look at the other hardware on the machine. In the main window there is a check box that tells SMAC to show only active hardware. This checkbox is checked by default. Uncheck that box and your listing will grow, depending on the hardware on your machine. Take a look at **Figure B** to see how much the listing grows on my laptop that includes wireless, wired, and dial-up connections.

Figure A

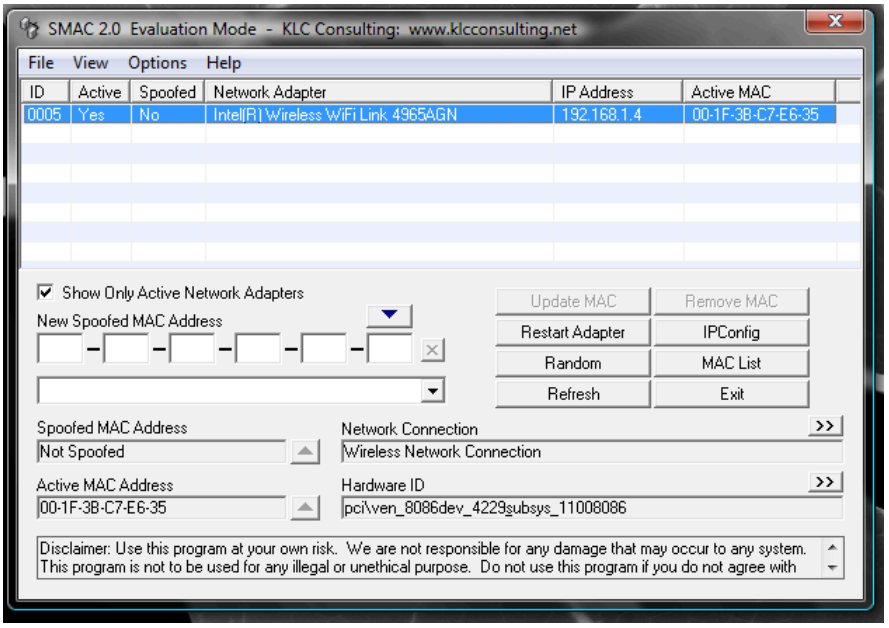
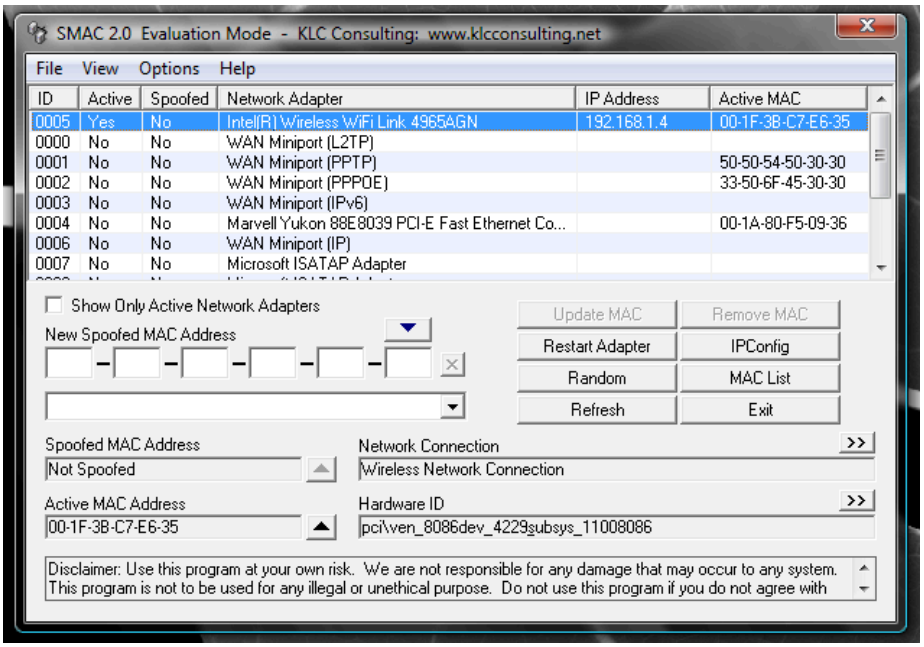


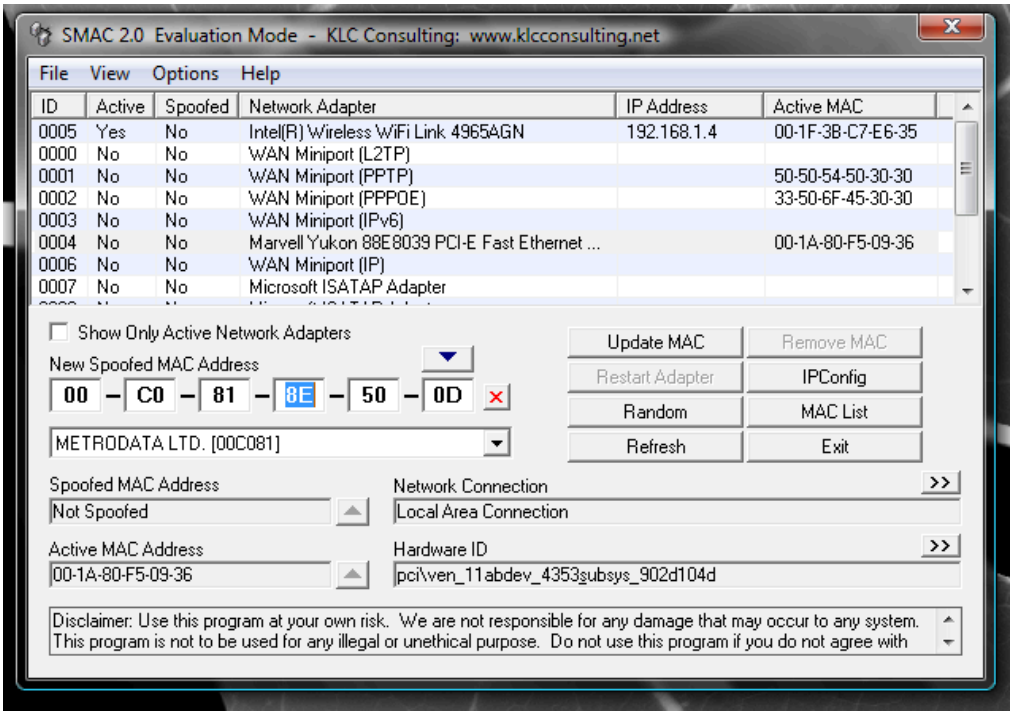
Figure B



When you click on a different listing, the information about that hardware will be displayed below.

Let's change the MAC address of the Wired Marvell Yukon PCI-E Faster Ethernet Controller. To do this, select that entry from the list and click the Random button. As you can see in **Figure C**, the new, random MAC address is displayed in the New Spoofed MAC Address section.

Figure C



The address listed will correspond to a manufacturer list that you can choose from.

If you know you want to spoof your MAC address to that of a specific manufacturer you can select a different manufacturer from the drop-down list. When you make this selection, the address listed will change. You can keep hitting Random until you get an address you like (or you can just take the first random address you get).

Once you have your address, select the Options menu and make sure Automatically Restart Adapter is checked. Once that is checked, hit the Update MAC Address button and the new MAC address will be applied.

Practical No. 6

a. Use the following tools to protect attacks on the web servers:

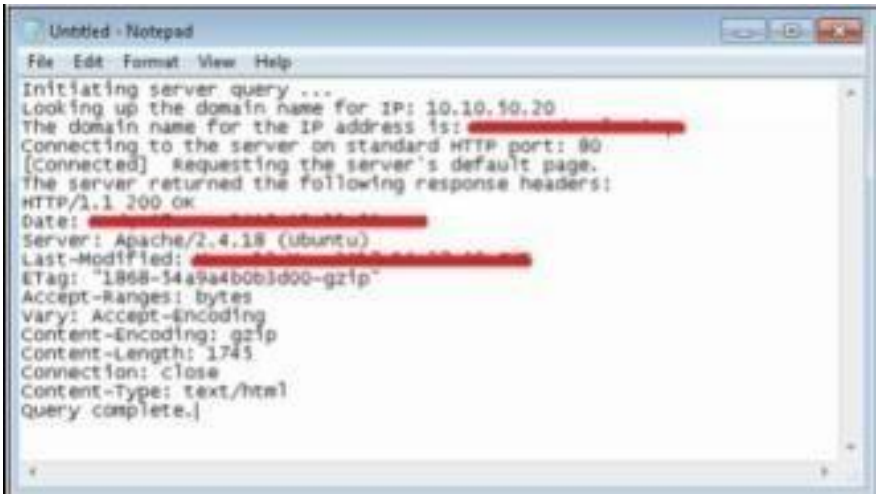
i. ID Server

Download and install ID Server tool.

1. Enter URL or IP address of the target server

2. Enter the **Query The Server**/button.

3. Copy the Extracted information.



Information such as Domain name, open ports, Server type and other information are extracted.

Practical No. 7

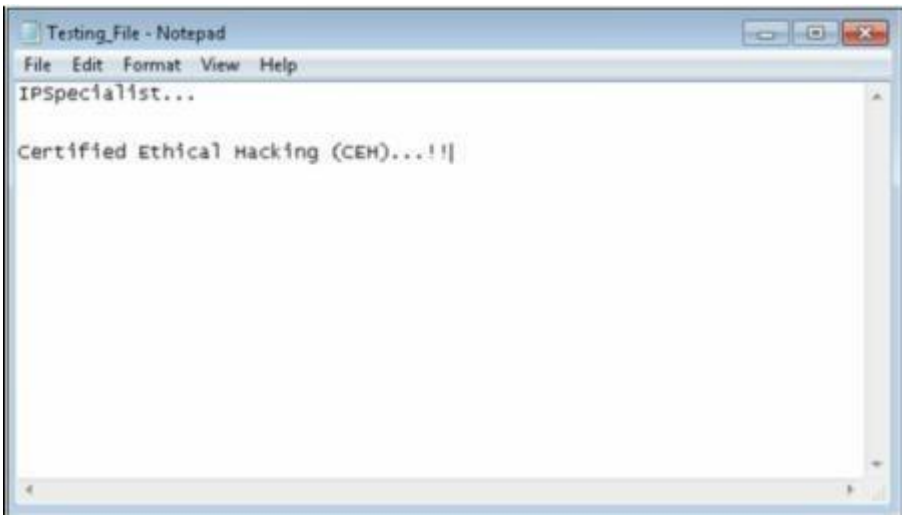
Use the following tools for cryptography

i.HashCalc

Calculating MD5 value using HashCalc

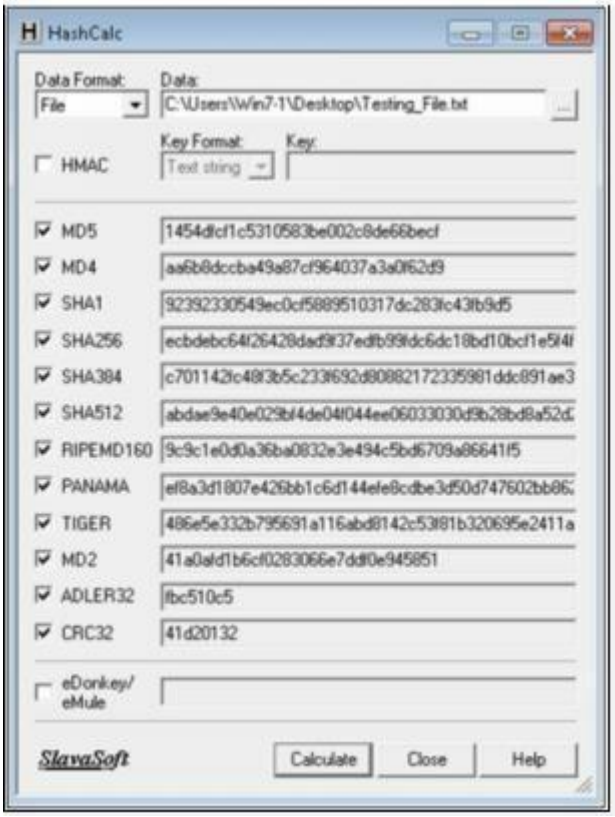
1. Open HashCalc tool.

2. Create a new file with some content in it as shown in below.



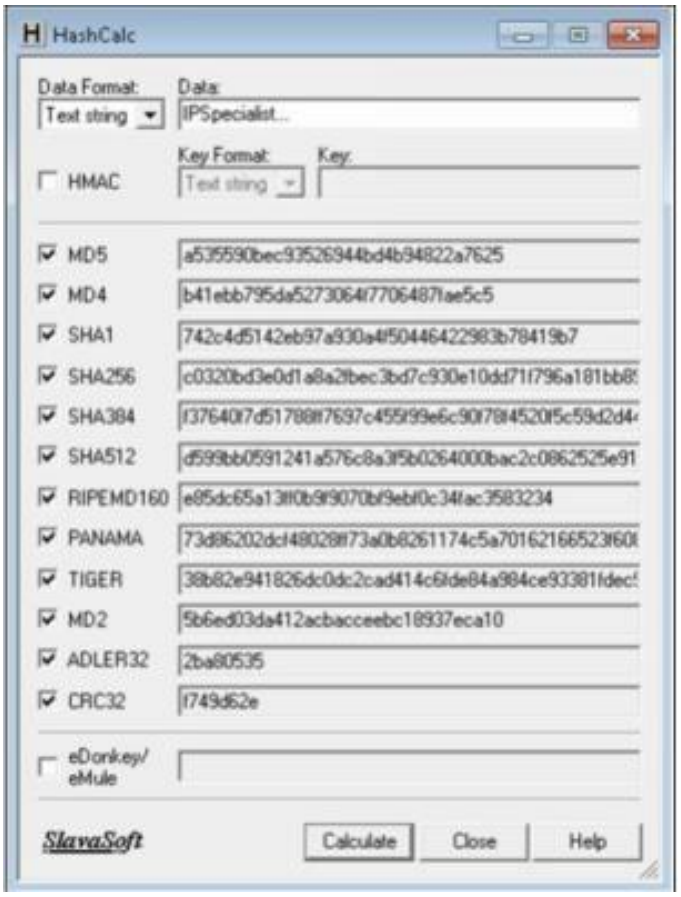
3. Select Data Format as “File” and upload your file

4. Select Hashing Algorithm and Click Calculate

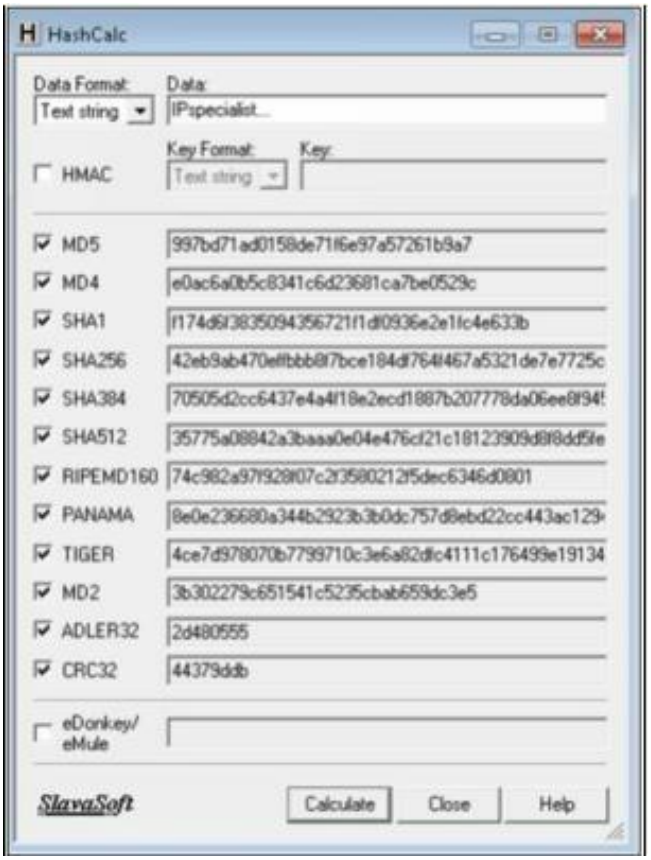


5. Now Select the Data Format to "**Text String**" and Type "**IPSpecialist…**" into Data filed and

Calculate MD5.



MD5 Calculated for the text string "**IPSpecialist…**" is "**a535590bec93526944bd4b94822a7625**"



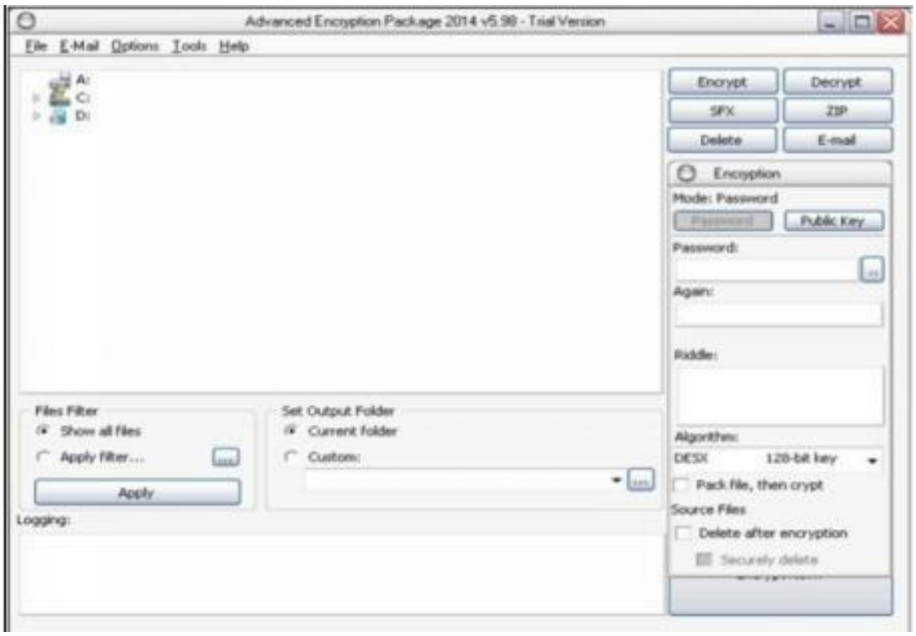
6. Now, let's see how MD5 value is changed from minor change.

Just lowering the case of single alphabet changes entire hashing value. MD5 Calculated for the text string "**IPspecialist…**" is

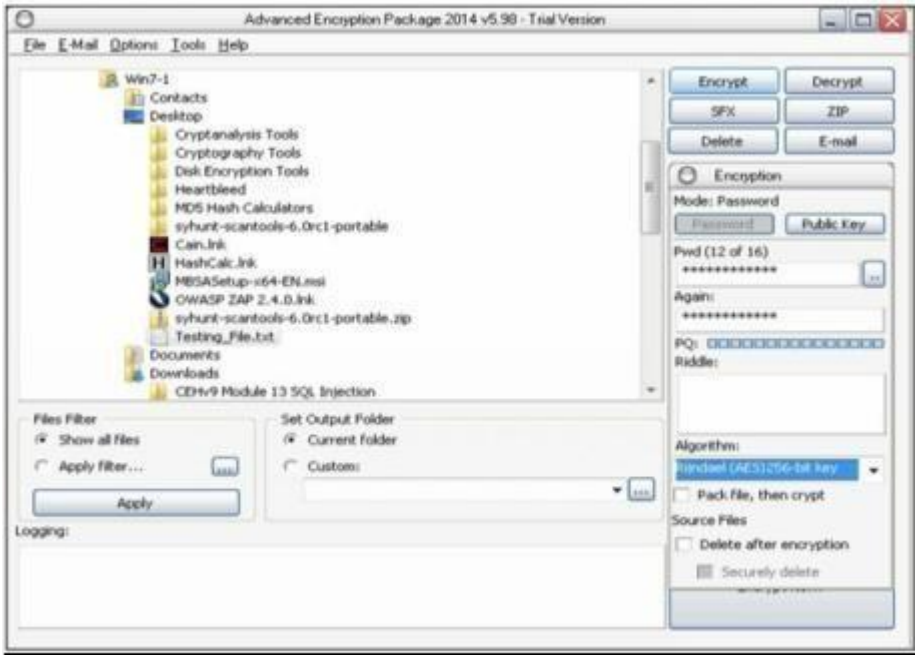
"**997bd71ad0158de71f6e97a57261b9a7"**

ii. Advanced Encryption Package

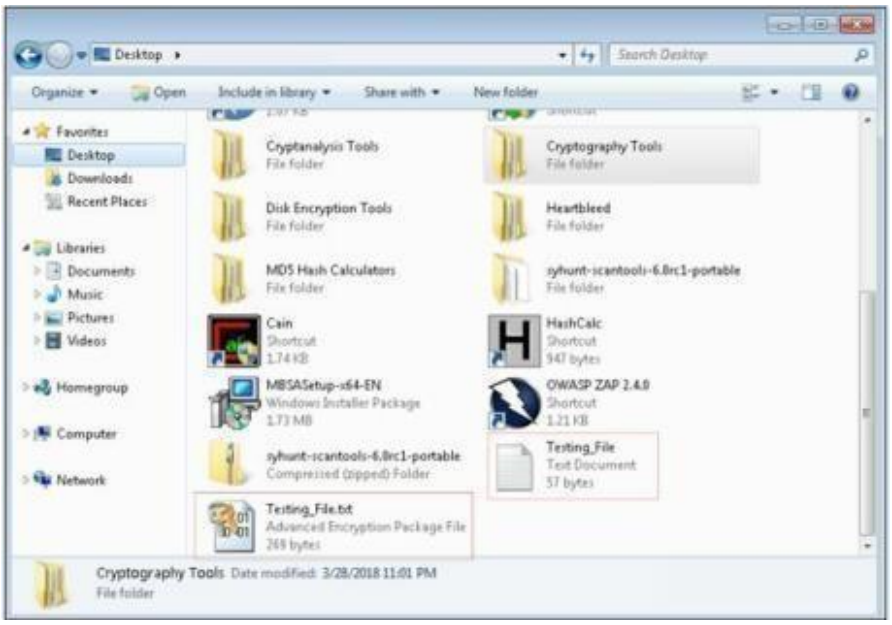
1. Download and Install Advance Encryption Package Latest Version. In this Lab, we are using Advanced Encryption Package 2014 and 2017 to ensure compatibilities on Windows 7 and Windows 10.



2. Select the File you want to Encrypt.
3. Set password
4. Select Algorithm



5. Click Encrypt

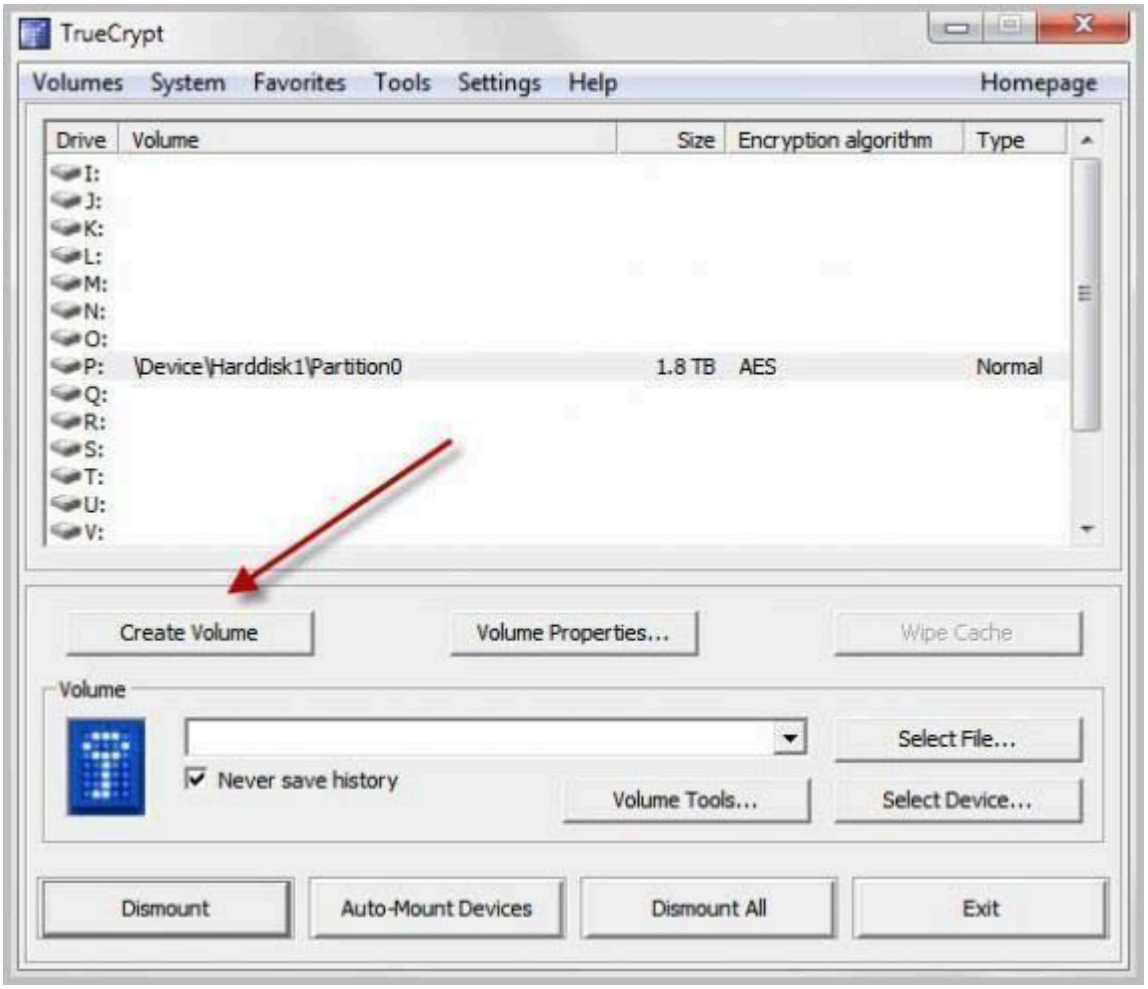


6. Compare both Files
7. Now, After forwarding it to another PC, in our case, in Windows 10 PC, decrypting it using Advanced Encryption package 2017.
8. Enter password

9. File Successfully decrypted.

iii. TrueCrypt

TrueCrypt is a leading disk encryption software program that lets you secure disk partitions on your Windows computer. There are times when your hard drive is accessible by other people, such as in an office setting, while travelling, or at home. The data you have on the PC may be vulnerable to attack and compromise your privacy. However, in these moments of risk, **TrueCrypt may just be the tool to protect your data.**



Click Next two times on the following screens to create an encrypted file container with a standard TrueCrypt volume (those are the default options). Click Select File and browse to a location where you want to create the new container. **Make sure it is not in the Dropbox**

folder if Dropbox is running. You can name the container anyway you want, e.g. holiday2010.avi.

Click Next on the encryption options page unless you want to change the encryption algorithm or hash algorithm. Select the volume size on the next screen. I suggest you keep it at a few hundred Megabytes tops.

You need to enter a secure password on the next screen. It is suggested to use as many characters as possible (24+) with upper and lower letters, numbers and special characters. The maximum length of a True Crypt password is 64 characters.

Now it is time to select the volume format on the next screen. If you only use Windows computers you may want to select NTFS as the file system. If you use others you may be better of with FAT. Juggle the mouse around a bit and click on format once you are done with that.

Congratulations, the new True Crypt volume has been created.

iv. CrypTool

Cryptool is a free e-learning tool to illustrate the concepts of cryptography. Try Various Encryption/Decryption algorithms.