Practical 1

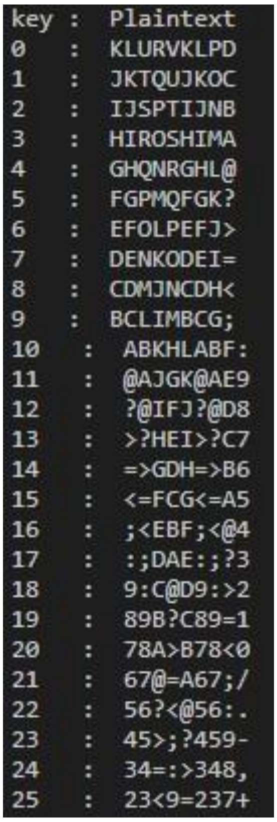
Date:15/12/2022

Aim: Apply attacks for cryptanalysis to decrypt the original message from a given cipher text using Caesar-Cipher. Soldier from field wants to send message to base. Implement the cipher to decrypt message. Decrypt message: KLURVKLPD for caesar cipher.

Solution:

```
#include <iostream>
#include <stdio.h>
using namespace std;

int main()
{
    string str = "KLURVKLPD";
    cout << "key :  Plaintext" << endl;
    for (int i = 0; i <= 25; i++)
    {
        string res;
        for (auto ch : str)
        {
            res += (ch - char(i) % char(26));
        }
        cout << i << "   :  " << res << endl;
    }
    return 0;
}
```

Output:



```
key :  Plaintext
0   :  KLURVKLPD
1   :  JKTQUJKOC
2   :  IJSPTIJNB
3   :  HIROSHIMA
4   :  GHQNRGHL@
5   :  FGPMQFGK?
6   :  EFOLPEFJ>
7   :  DENKODEI=
8   :  CDMJNCDH<
9   :  BCLIMBCG;
10   :  ABKHLABF:
11   :  @AJGK@AE9
12   :  ?@IFJ?@D8
13   :  >?HEI>?C7
14   :  =>GDH=>B6
15   :  <=FCG<=A5
16   :  ;<EBF;<@4
17   :  :;DAE:;?3
18   :  9:C@D9:>2
19   :  89B?C89=1
20   :  78A>B78<0
21   :  67@=A67;/
22   :  56?<@56:.
23   :  45>;?459-
24   :  34=:>348,
25   :  23<9=237+
```

Conclusion/Summary:

Student Signature & Date	**Marks:**	**Evaluator Signature & Date**

Practical 2

Date:22/12/2022

Aim: Soldier from field wants to send message to base. Implement the cipher to decrypt message. Using Playfair, Decrypt the message: BWPNRSMUALAW, Use key: pearlharbour

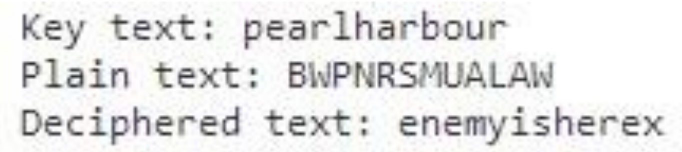
Solution:

```
#include <stdio.h>
#include <iostream>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#define SIZE 30

void toLowerCase(char plain[], int ps)
{
    for (int i = 0; i < ps; i++)
    {
        if (plain[i] > 64 && plain[i] < 91)
            plain[i] += 32;
    }
}

int removeSpaces(char *plain, int ps)
{
    int count = 0;
    for (int i = 0; i < ps; i++)
        if (plain[i] != ' ')
            plain[count++] = plain[i];
    plain[count] = '\0';
    return count;
}

void generateKeyTable(char key[], int ks, char keyT[5][5])
{
    int i, j, k, flag = 0, *dicty;
    dicty = (int *)calloc(26, sizeof(int));

    for (i = 0; i < ks; i++)
    {
        if (key[i] != 'j')
            dicty[key[i] - 97] = 2;
    }
    dicty['j' - 97] = 1;

    i = 0;
    j = 0;
    for (k = 0; k < ks; k++)
    {
        if (dicty[key[k] - 97] == 2)
        {
            dicty[key[k] - 97] -= 1;
            keyT[i][j] = key[k];
            j++;
            if (j == 5)
            {
                i++;
                j = 0;
            }
        }
    }
    for (k = 0; k < 26; k++)
    {
        if (dicty[k] == 0)
        {
            keyT[i][j] = (char)(k + 97);
            j++;
            if (j == 5)
            {
                i++;
                j = 0;
            }
        }
    }
}
```

```c
void search(char keyT[5][5], char a, char b, int arr[])
{
    int i, j;

    if (a == 'j')
        a = 'i';
    else if (b == 'j')
        b = 'i';

    for (i = 0; i < 5; i++)
    {
        for (j = 0; j < 5; j++)
        {
            if (keyT[i][j] == a)
            {
                arr[0] = i;
                arr[1] = j;
            }
            else if (keyT[i][j] == b)
            {
                arr[2] = i;
                arr[3] = j;
            }
        }
    }
}

int mod5(int a)
{
    if (a < 0)
        a += 5;
    return (a % 5);
}

void decrypt(char str[], char keyT[5][5], int ps)
{
    int i, a[4];
    for (i = 0; i < ps; i += 2)
    {
        search(keyT, str[i], str[i + 1], a);
        if (a[0] == a[2])
        {
            str[i] = keyT[a[0]][mod5(a[1] - 1)];
            str[i + 1] = keyT[a[0]][mod5(a[3] - 1)];
        }
        else if (a[1] == a[3])
        {
            str[i] = keyT[mod5(a[0] - 1)][a[1]];
            str[i + 1] = keyT[mod5(a[2] - 1)][a[1]];
        }
        else
        {
            str[i] = keyT[a[0]][a[3]];
            str[i + 1] = keyT[a[2]][a[1]];
        }
    }
}

void decryptByPlayfairCipher(char str[], char key[])
{
    char ps, ks, keyT[5][5];

    // Key
    ks = strlen(key);
    ks = removeSpaces(key, ks);
    toLowerCase(key, ks);

    // ciphertext
    ps = strlen(str);
    toLowerCase(str, ps);
    ps = removeSpaces(str, ps);

    generateKeyTable(key, ks, keyT);

    decrypt(str, keyT, ps);
}
```

```
int main()
{
    char str[SIZE], key[SIZE];

    strcpy(key, "pearlharbour");
    printf("Key text: %s\n", key);

    strcpy(str, "BWPNRSMUALAW");
    printf("Plain text: %s\n", str);

    decryptByPlayfairCipher(str, key);
    printf("Deciphered text: %s\n", str);
    return 0;
}
```

Output:



```
Key text: pearlharbour
Plain text: BWPNRSMUALAW
Deciphered text: enemyisherex
```

Conclusion/Summary:

Student Signature & Date	**Marks:**	**Evaluator Signature & Date**

Practical 3

Date:29/12/2022

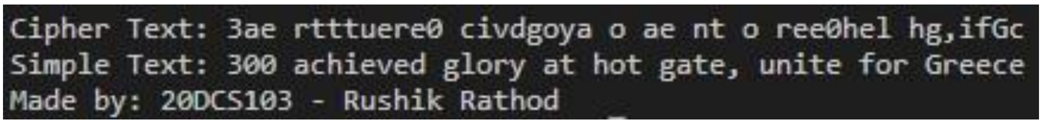
Aim: The Rail Fence Cipher was invented in ancient times. It was used by the Greeks, who created a special tool, called scytale, to make message encryption and decryption easier. The letters are arranged in a way which is similar to the shape of the top edge of the rail fence. If king Leonidas want to send message to Sparta as “300 achieved glory at hot gate, unite for Greece” then what will be cipher text when it is encrypted using 3 rows. Also implement decryption of message.

Solution:

```
#include <stdio.h>
#include <iostream>
#include <string>

using namespace std;

string encryptRailFence(string text, int key)
{
    char rail[key][(text.length())];

    for (int i = 0; i < key; i++)
        for (int j = 0; j < text.length(); j++)
            rail[i][j] = '\n';

    bool dir_down = false;
    int row = 0, col = 0;

    for (int i = 0; i < text.length(); i++)
    {
        if (row == 0 || row == key - 1)
            dir_down = !dir_down;

        rail[row][col++] = text[i];
        dir_down ? row++ : row--;
    }

    string result;
    for (int i = 0; i < key; i++)
        for (int j = 0; j < text.length(); j++)
            if (rail[i][j] != '\n')
```

```
                result.push_back(rail[i][j]);

    return result;
}

string decryptRailFence(string cipher, int key)
{
    char rail[key][cipher.length()];

    for (int i = 0; i < key; i++)
        for (int j = 0; j < cipher.length(); j++)
            rail[i][j] = '\n';

    bool dir_down;
    int row = 0, col = 0;

    for (int i = 0; i < cipher.length(); i++)
    {
        if (row == 0)
            dir_down = true;
        if (row == key - 1)
            dir_down = false;

        rail[row][col++] = '*';
        dir_down ? row++ : row--;
    }

    int index = 0;
    for (int i = 0; i < key; i++)
    {
        for (int j = 0; j < cipher.length(); j++)
        {
            if (rail[i][j] == '*' && index < cipher.length())
                rail[i][j] = cipher[index++];
        }
    }

    string result;
    row = 0, col = 0;
    for (int i = 0; i < cipher.length(); i++)
    {
        if (row == 0)
            dir_down = true;
        if (row == key - 1)
            dir_down = false;

        if (rail[row][col] != '*')
            result.push_back(rail[row][col++]);

        dir_down ? row++ : row--;
    }
    return result;
}

int main()
{
    string simpleText = "300 achieved glory at hot gate, unite for Greece";
    string cipherText = encryptRailFence(simpleText, 3);
    cout << "Cipher Text: " << cipherText << endl;
    cout << "Simple Text: " << decryptRailFence(cipherText, 3) << endl;
    cout << "Made by: 20DCS103 - Rushik Rathod";
    return 0;
}
```

Output:



```
Cipher Text: 3ae rtttuere0 civdgoya o ae nt o ree0hel hg,ifGc
Simple Text: 300 achieved glory at hot gate, unite for Greece
Made by: 20DCS103 - Rushik Rathod
```

Conclusion/Summary:

Student Signature & Date	**Marks:**	**Evaluator Signature & Date**

Practical 4

Date:05/01/2023

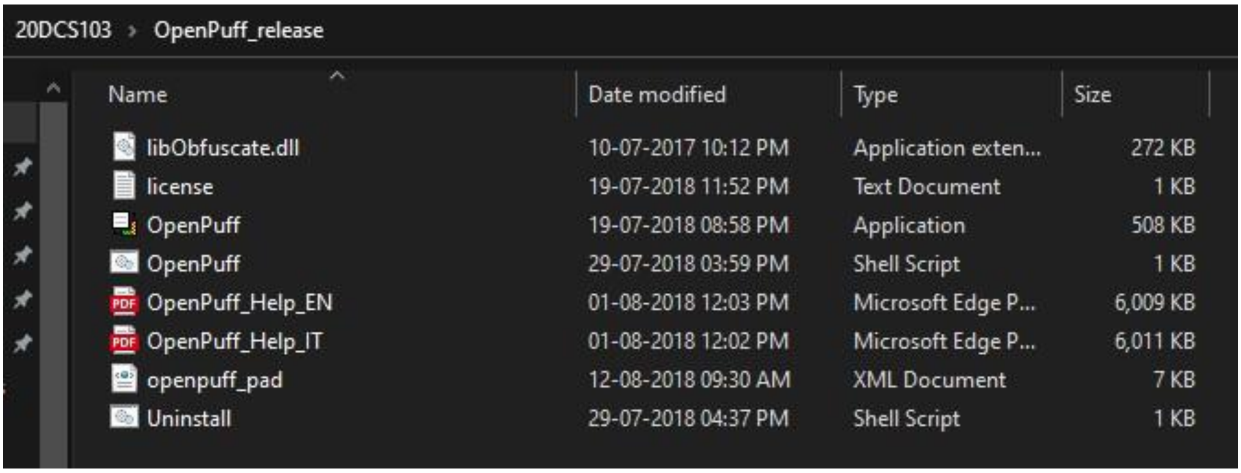
Aim: The transmission of information needs to be secure over the communication channel and the data has to be confidential. Study and implement the practical approach for

Steganography.

- Using OpenPuff Tool

Steps to Hide:

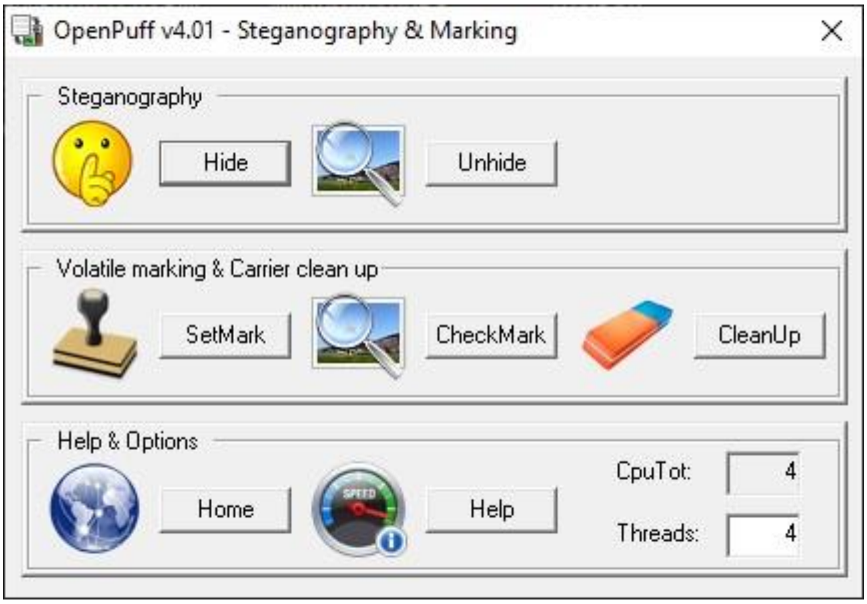
1) **After unzipping OpenPuff into a directory.**



20DCS103 › OpenPuff_release

Name	Date modified	Type	Size
libObfuscate.dll	10-07-2017 10:12 PM	Application exten...	272 KB
license	19-07-2018 11:52 PM	Text Document	1 KB
OpenPuff	19-07-2018 08:58 PM	Application	508 KB
OpenPuff	29-07-2018 03:59 PM	Shell Script	1 KB
OpenPuff_Help_EN	01-08-2018 12:03 PM	Microsoft Edge P...	6,009 KB
OpenPuff_Help_IT	01-08-2018 12:02 PM	Microsoft Edge P...	6,011 KB
openpuff_pad	12-08-2018 09:30 AM	XML Document	7 KB
Uninstall	29-07-2018 04:37 PM	Shell Script	1 KB

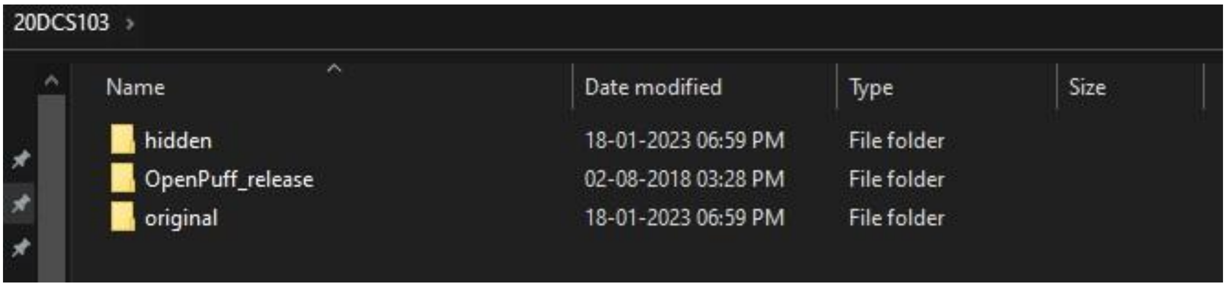
2) **Click on OpenPuff.**


OpenPuff v4.01 - Steganography & Marking
Steganography
Hide
Unhide
Volatile marking & Carrier clean up
SetMark
CheckMark
CleanUp
Help & Options
Home
Help

CpuTot: 4

Threads: 4

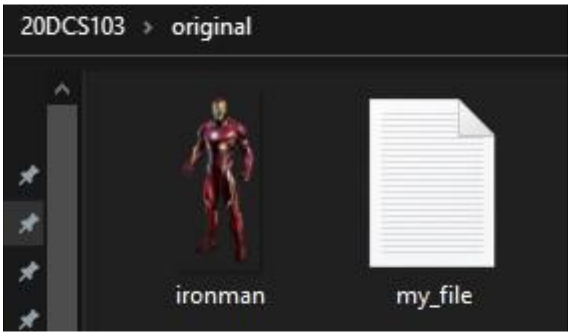
3) **Create hidden and original folders as shown below.**



20DCS103 >

Name	Date modified	Type	Size
hidden	18-01-2023 06:59 PM	File folder	
OpenPuff_release	02-08-2018 03:28 PM	File folder	
original	18-01-2023 06:59 PM	File folder	

4) **Create a new text file and add an image into folder named original.**

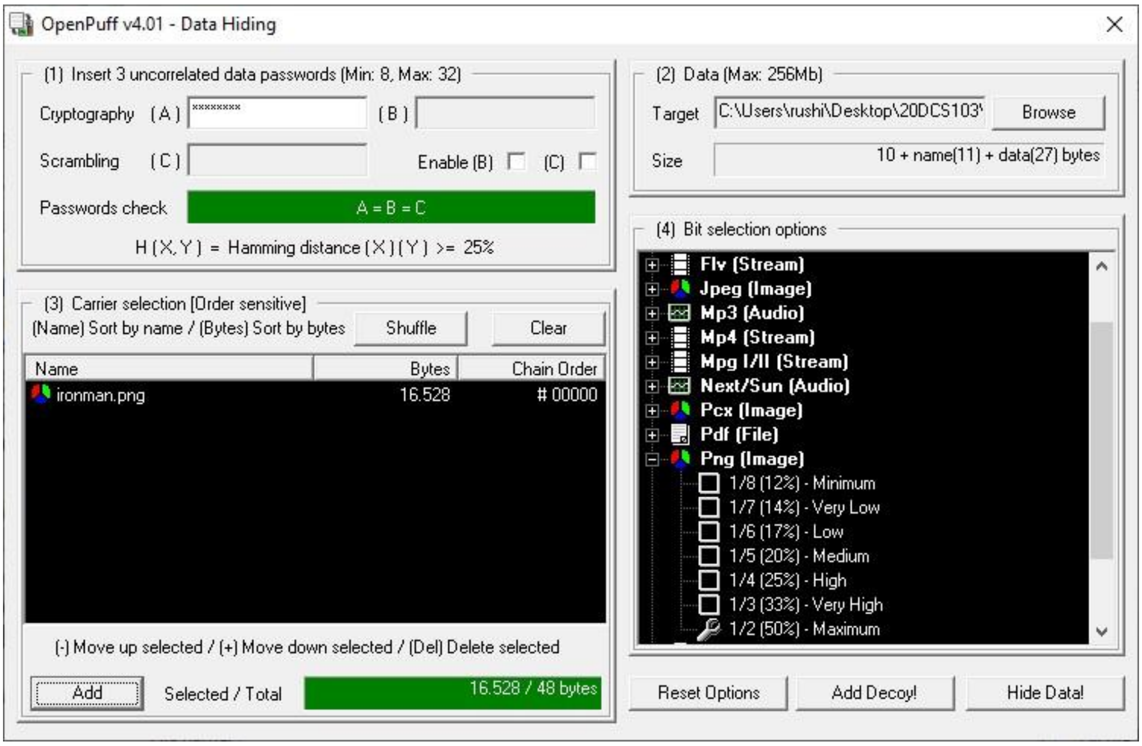


20DCS103 > original

ironman

my_file

5) **Now, click on Hide in OpenPuff.**
Enter password → Select the target textfile from the original folder → Select Carrier image from the original folded → Select the type of the image.



OpenPuff v4.01 - Data Hiding

(1) Insert 3 uncorrelated data passwords (Min: 8, Max: 32)

Cryptography (A) ******** (B)

Scrambling (C) Enable (B) □ (C) □

Passwords check A = B = C

H (X,Y) = Hamming distance (X) (Y) >= 25%

(2) Data (Max: 256Mb)

Target C:\Users\rushi\Desktop\20DCS103\ Browse

Size 10 + name(11) + data(27) bytes

(3) Carrier selection (Order sensitive)

(Name) Sort by name / (Bytes) Sort by bytes Shuffle Clear

Name	Bytes	Chain Order
ironman.png	16.528	# 00000

(-) Move up selected / (+) Move down selected / (Del) Delete selected

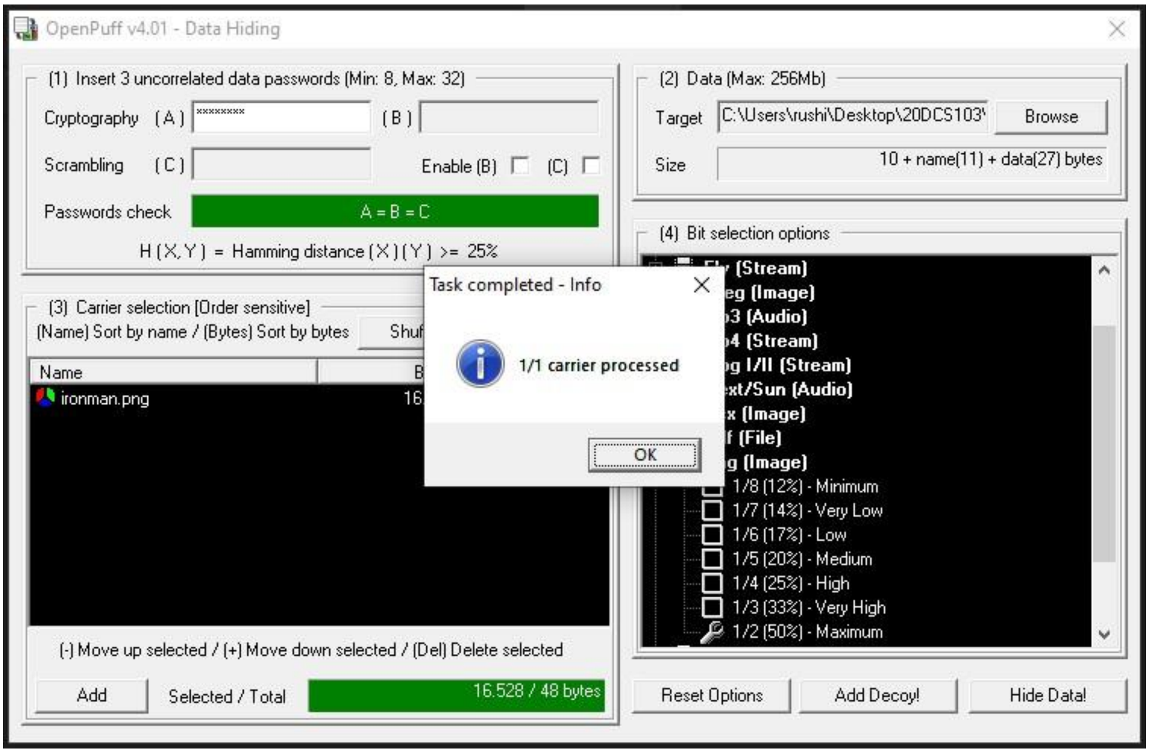
Add Selected / Total 16.528 / 48 bytes

(4) Bit selection options

- Flv (Stream)
- Jpeg (Image)
- Mp3 (Audio)
- Mp4 (Stream)
- Mpg I/II (Stream)
- Next/Sun (Audio)
- Pcx (Image)
- Pdf (File)
- Png (Image)
  - □ 1/8 (12%) - Minimum
  - □ 1/7 (14%) - Very Low
  - □ 1/6 (17%) - Low
  - □ 1/5 (20%) - Medium
  - □ 1/4 (25%) - High
  - □ 1/3 (33%) - Very High
  - 1/2 (50%) - Maximum

Reset Options Add Decoy! Hide Data!

6) On clicking Hide Data → Select the output directory named hidden and after confirming, carrier is processed.



OpenPuff v4.01 - Data Hiding

(1) Insert 3 uncorrelated data passwords (Min: 8, Max: 32)

Cryptography (A) ********* (B)

Scrambling (C) Enable (B) □ (C) □

Passwords check A = B = C

H (X, Y) = Hamming distance (X) (Y) >= 25%

(2) Data (Max: 256Mb)

Target C:\Users\rushi\Desktop\20DCS103\ Browse

Size 10 + name(11) + data(27) bytes

(3) Carrier selection [Order sensitive]

(Name) Sort by name / (Bytes) Sort by bytes Shuf

Name	B
ironman.png	16

(-) Move up selected / (+) Move down selected / (Del) Delete selected

Add Selected / Total 16.528 / 48 bytes

(4) Bit selection options

□ 1/8 (12%) - Minimum
□ 1/7 (14%) - Very Low
□ 1/6 (17%) - Low
□ 1/5 (20%) - Medium
□ 1/4 (25%) - High
□ 1/3 (33%) - Very High
1/2 (50%) - Maximum

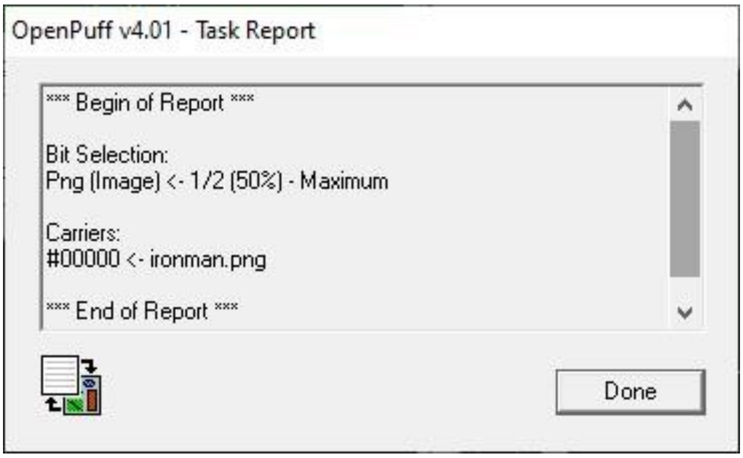
Reset Options Add Decoy! Hide Data!

Task completed - Info

1/1 carrier processed

OK

7) Viewing Report generated for Carrier.

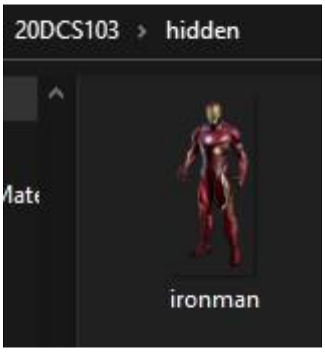


OpenPuff v4.01 - Task Report

```
*** Begin of Report ***

Bit Selection:
Png (Image) <- 1/2 (50%) - Maximum

Carriers:
#00000 <- ironman.png

*** End of Report ***
```

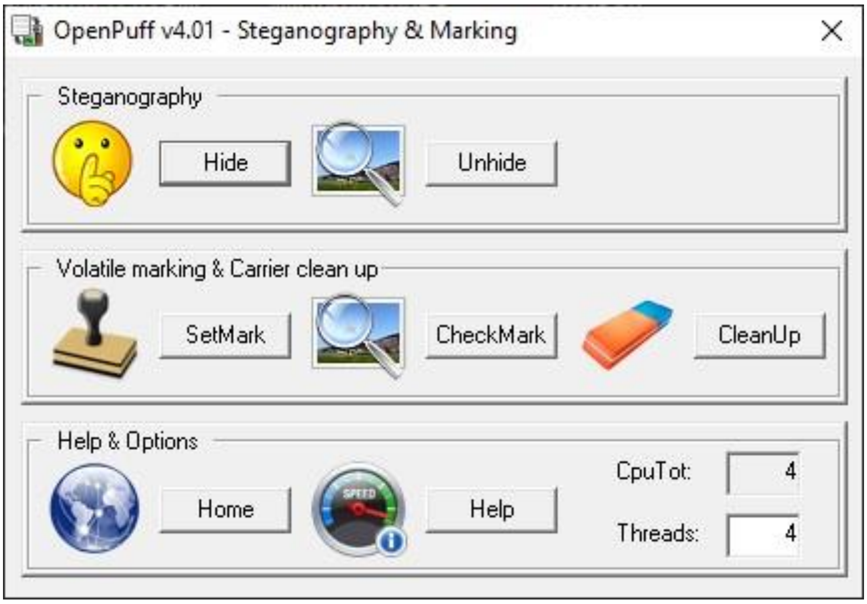
Done

8) Viewing Carrier Generated file in Directory.

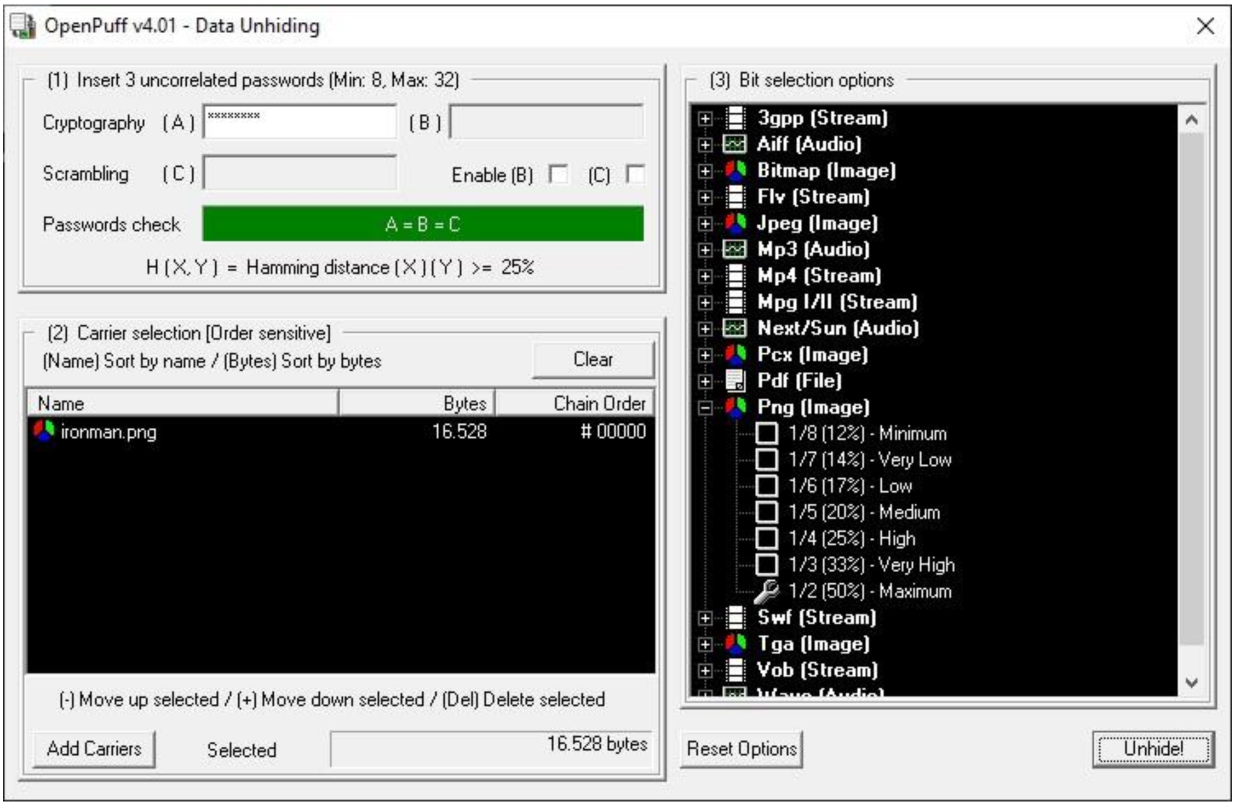


Steps to Unhide:

1) **Open OpenPuff → Select unhide.**



2) **Enter password → Select output directory hidden → Click OK**



OpenPuff v4.01 - Data Unhiding

(1) Insert 3 uncorrelated passwords (Min: 8, Max: 32)

Cryptography (A) ******** (B)

Scrambling (C) Enable (B) □ (C) □

Passwords check A = B = C

H (X, Y) = Hamming distance (X) (Y) >= 25%

(2) Carrier selection [Order sensitive]

(Name) Sort by name / (Bytes) Sort by bytes

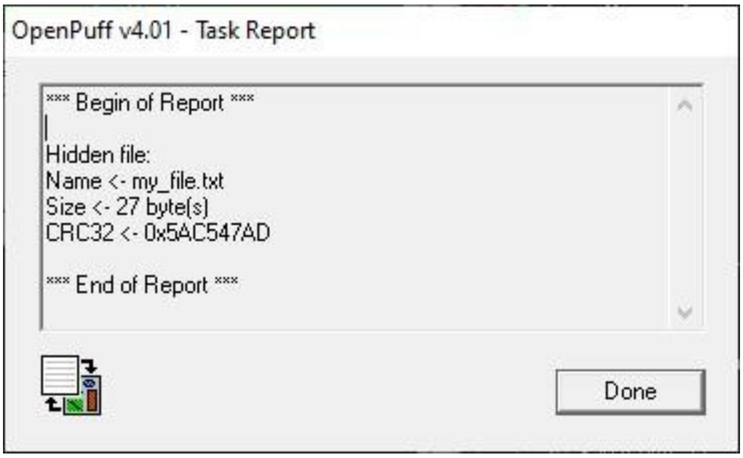
Name	Bytes	Chain Order
ironman.png	16.528	# 00000

(-) Move up selected / (+) Move down selected / (Del) Delete selected

Selected 16.528 bytes

(3) Bit selection options

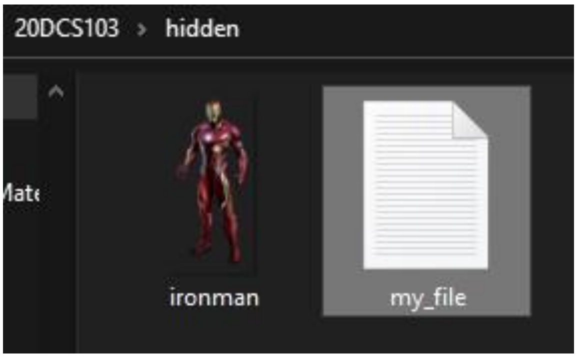
3) Unhide Report.



OpenPuff v4.01 - Task Report

```
*** Begin of Report ***

Hidden file:
Name <- my_file.txt
Size <- 27 byte(s)
CRC32 <- 0x5AC547AD

*** End of Report ***
```

4) Checking output directory 'hidden' and a text file.



Conclusion/Summary:

Student Signature & Date	**Marks:**	**Evaluator Signature & Date**

Practical 5

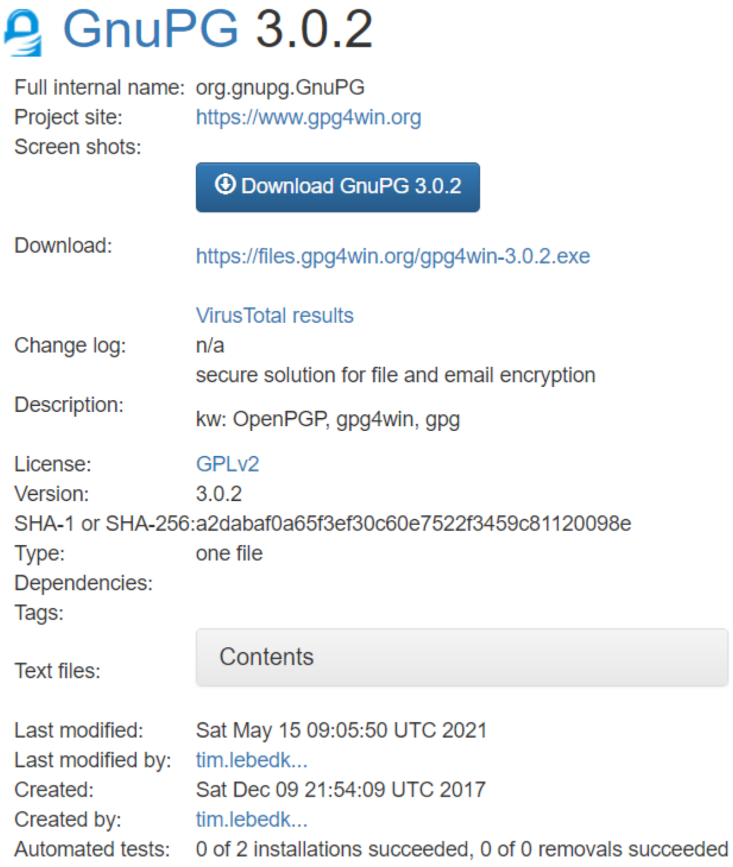
Date: / /2023

Aim: Implement GPG for windows.

Solution:

Steps:

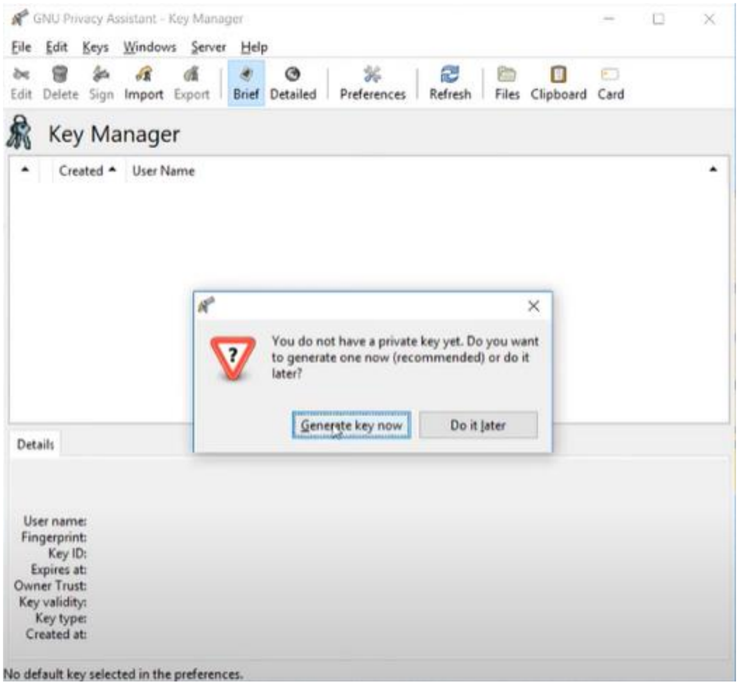
1) **Download GnuPG 3.0.2**



GnuPG 3.0.2

Full internal name:	org.gnupg.GnuPG
Project site:	https://www.gpg4win.org
Screen shots:	
	Download GnuPG 3.0.2
Download:	https://files.gpg4win.org/gpg4win-3.0.2.exe VirusTotal results
Change log:	n/a
Description:	secure solution for file and email encryption kw: OpenPGP, gpg4win, gpg
License:	GPLv2
Version:	3.0.2
SHA-1 or SHA-256:	a2dabaf0a65f3ef30c60e7522f3459c81120098e
Type:	one file
Dependencies:	
Tags:	
Text files:	Contents
Last modified:	Sat May 15 09:05:50 UTC 2021
Last modified by:	tim.lebedk...
Created:	Sat Dec 09 21:54:09 UTC 2017
Created by:	tim.lebedk...
Automated tests:	0 of 2 installations succeeded, 0 of 0 removals succeeded

2) **Install GnuPG, open the software and click ‘Generate key now’.**

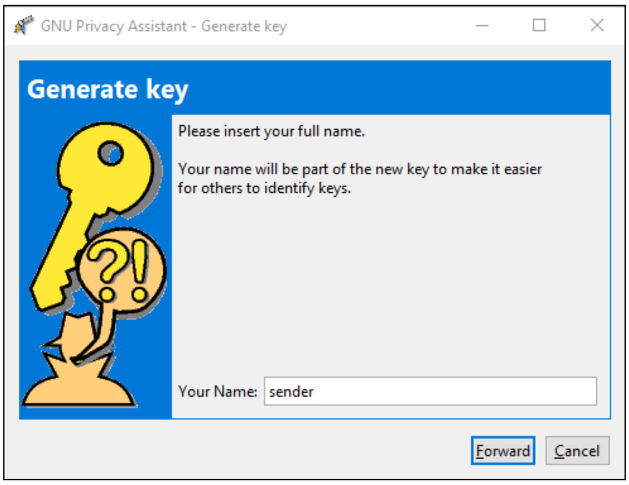

GNU Privacy Assistant - Key Manager
File Edit Keys Windows Server Help
Edit Delete Sign Import Export Brief Detailed Preferences Refresh Files Clipboard Card
Key Manager
Created User Name
Details
Generate key now
Do it later

You do not have a private key yet. Do you want to generate one now (recommended) or do it later?

User name:
Fingerprint:
Key ID:
Expires at:
Owner Trust:
Key validity:
Key type:
Created at:

No default key selected in the preferences.

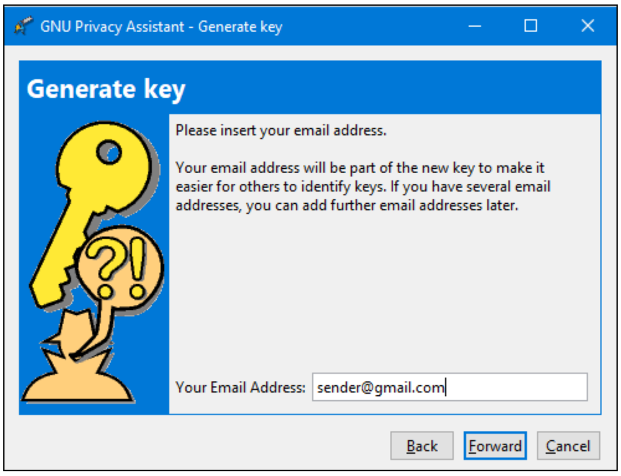
3) **Sender side.**



Please insert your full name.

Your name will be part of the new key to make it easier for others to identify keys.

Your Name: sender

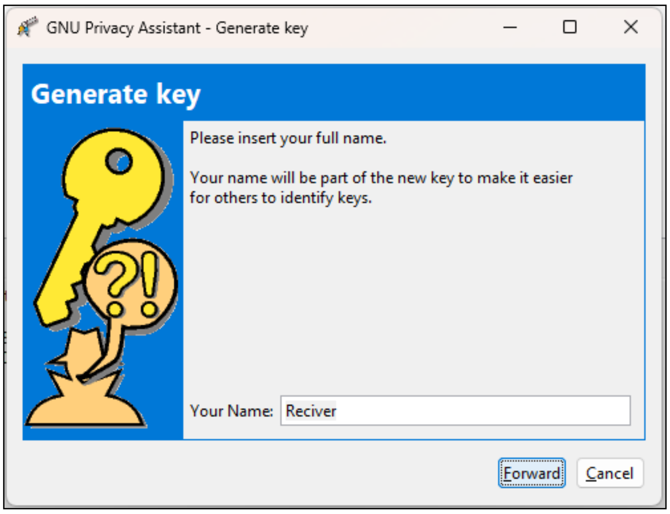


Please insert your email address.

Your email address will be part of the new key to make it easier for others to identify keys. If you have several email addresses, you can add further email addresses later.

Your Email Address: sender@gmail.com

4) Receiver side.

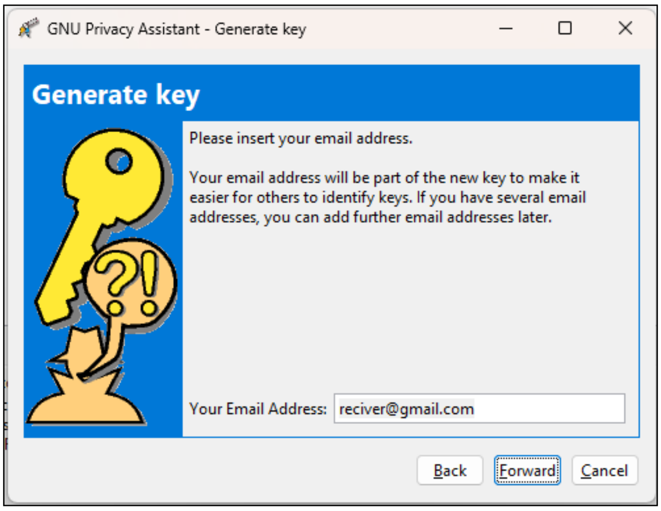

GNU Privacy Assistant - Generate key

Generate key

Please insert your full name.

Your name will be part of the new key to make it easier for others to identify keys.

Your Name: Reciver


GNU Privacy Assistant - Generate key

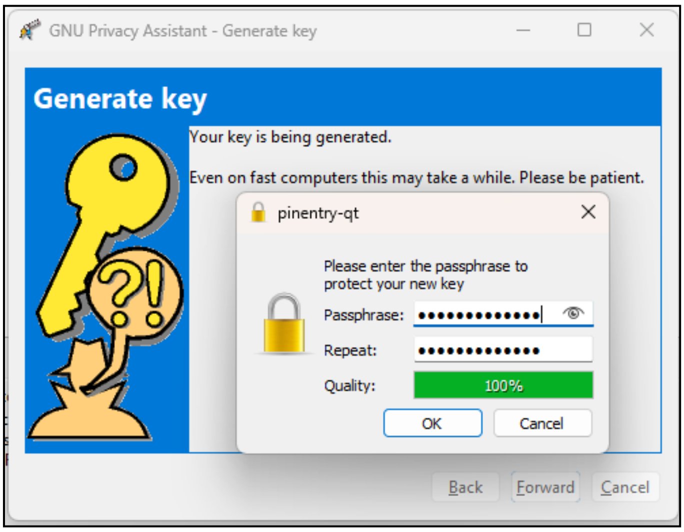
Generate key

Please insert your email address.

Your email address will be part of the new key to make it easier for others to identify keys. If you have several email addresses, you can add further email addresses later.

Your Email Address: reciver@gmail.com

5) Then create the receiver’s & sender’s public and private key respectively.


GNU Privacy Assistant - Generate key

Generate key

Your key is being generated.

Even on fast computers this may take a while. Please be patient.

pinentry-qt

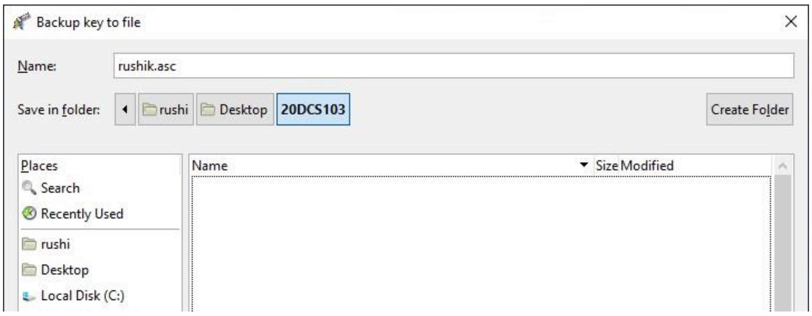
Please enter the passphrase to protect your new key

Passphrase: ••••••••••••••

Repeat: ••••••••••••••

Quality: 100%

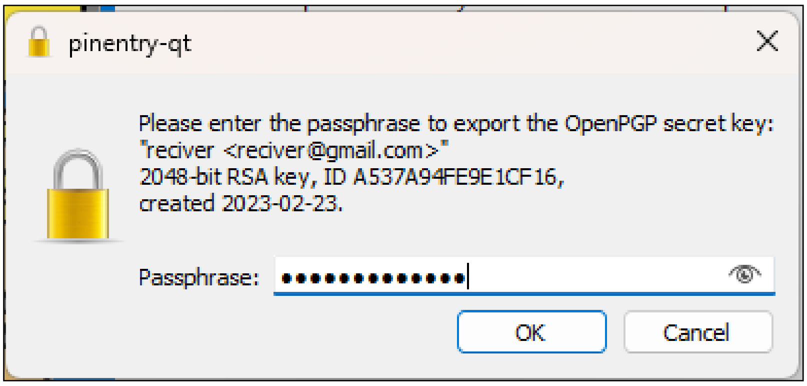
6) Create the backup file for your key on the desktop.


Backup key to file

Name: rushik.asc

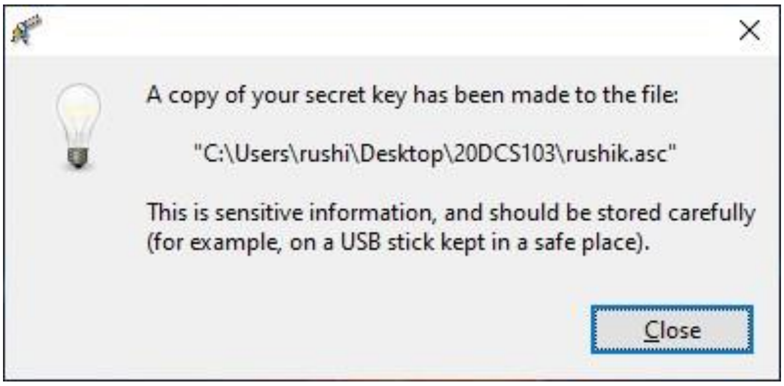
Save in folder: rushi / Desktop / 20DCS103

Enter Passphrase and click OK.


pinentry-qt

Please enter the passphrase to export the OpenPGP secret key:
"reciver <reciver@gmail.com>"
2048-bit RSA key, ID A537A94FE9E1CF16,
created 2023-02-23.

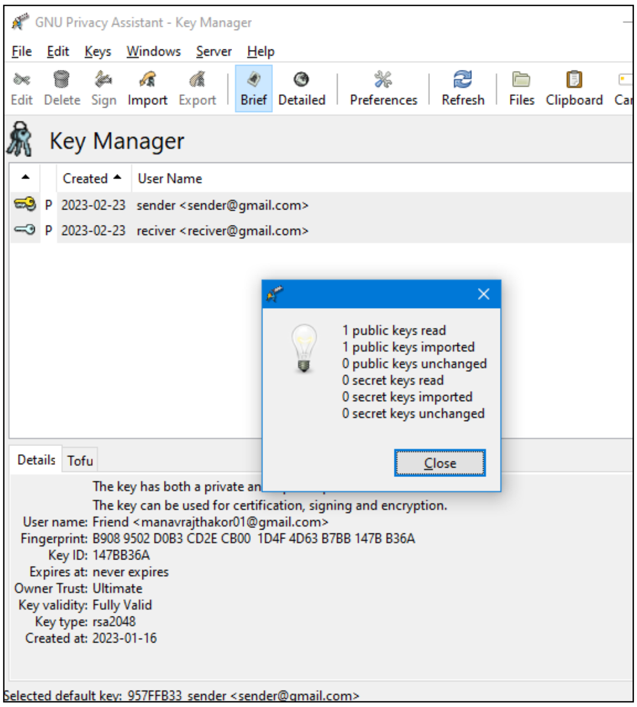
Passphrase: ●●●●●●●●●●●●●


A copy of your secret key has been made to the file:

"C:\Users\rushi\Desktop\20DCS103\rushik.asc"

This is sensitive information, and should be stored carefully (for example, on a USB stick kept in a safe place).

7) Then open the file where key is stored and edit out the private key of receiver and then send the remaining asc file (which contains public key of receiver) to the sender. Import the key of receiver on the sender's side to encrypt the message.

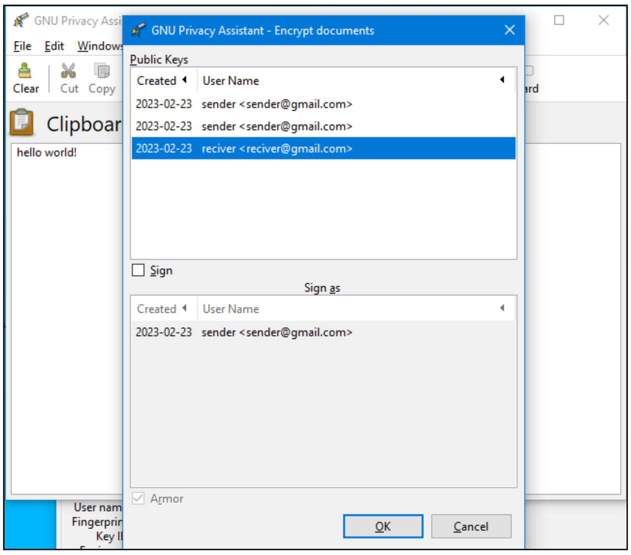

GNU Privacy Assistant - Key Manager
File Edit Keys Windows Server Help
Edit Delete Sign Import Export Brief Detailed Preferences Refresh Files Clipboard Ca
Key Manager
Details Tofu
Close

	Created	User Name
P	2023-02-23	sender <sender@gmail.com>
P	2023-02-23	reciver <reciver@gmail.com>

1 public keys read
1 public keys imported
0 public keys unchanged
0 secret keys read
0 secret keys imported
0 secret keys unchanged

The key has both a private an
The key can be used for certification, signing and encryption.
User name: Friend <manavrajthakor01@gmail.com>
Fingerprint: B908 9502 D0B3 CD2E CB00 1D4F 4D63 B7BB 147B B36A
Key ID: 147BB36A
Expires at: never expires
Owner Trust: Ultimate
Key validity: Fully Valid
Key type: rsa2048
Created at: 2023-01-16

Selected default key: 957FFB33 sender <sender@gmail.com>



hello world!

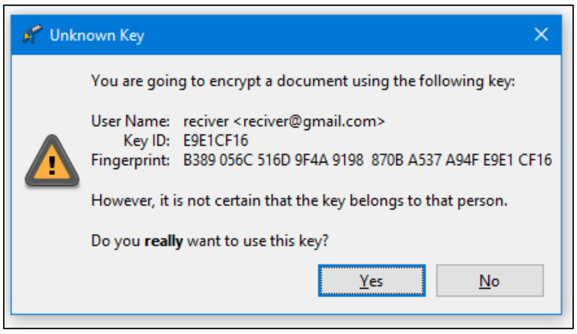
Created	User Name
2023-02-23	sender <sender@gmail.com>
2023-02-23	sender <sender@gmail.com>
2023-02-23	reciver <reciver@gmail.com>

□ Sign

Created	User Name
2023-02-23	sender <sender@gmail.com>

☑ Armor

User nam
Fingerprin
Key I



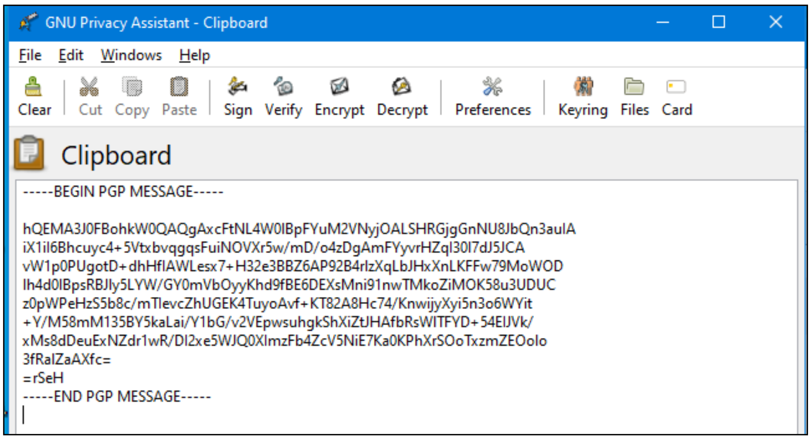
You are going to encrypt a document using the following key:

User Name: reciver <reciver@gmail.com>
Key ID: E9E1CF16
Fingerprint: B389 056C 516D 9F4A 9198 870B A537 A94F E9E1 CF16

However, it is not certain that the key belongs to that person.

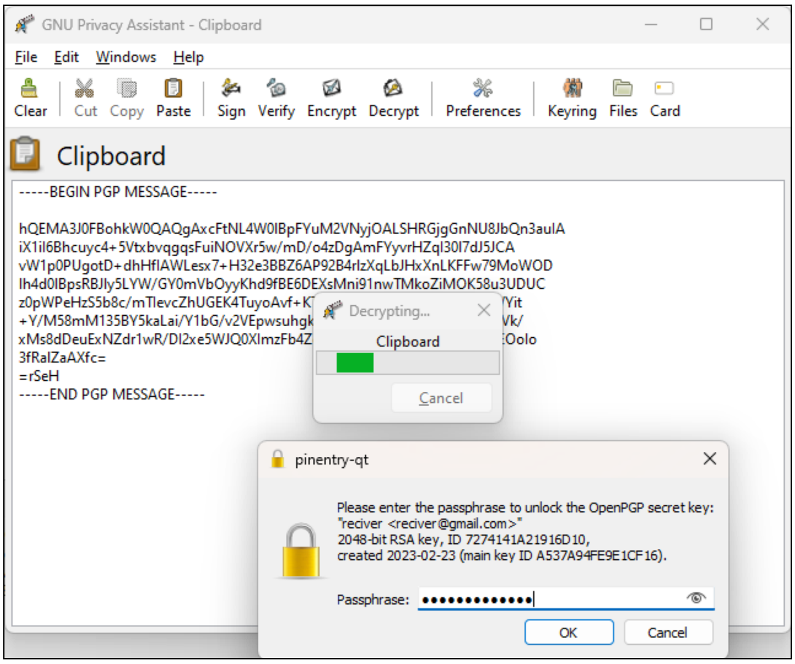
Do you **really** want to use this key?

8) Here the message is encrypted using the receiver's public key.


GNU Privacy Assistant - Clipboard
File Edit Windows Help
Clear Cut Copy Paste Sign Verify Encrypt Decrypt Preferences Keyring Files Card
Clipboard

```
-----BEGIN PGP MESSAGE-----

hQEMA3J0FBohkW0QAQgAxcFtNL4W0IBpFYuM2VNyjOALSHRGjgGnNU8JbQn3auIA
iX1il6Bhcuyc4+5VtxbvqgqsFuiNOVXr5w/mD/o4zDgAmFYyvrHZql30I7dJ5JCA
vW1p0PUgotD+dhHfIAWLesx7+H32e3BBZ6AP92B4rlzXqLbJHxXnLKFFw79MoWOD
Ih4d0IBpsRBJly5LYW/GY0mVbOyyKhd9fBE6DEXsMni91nwTMkoZiMOK58u3UDUC
z0pWPeHzS5b8c/mTlevcZhUGEK4TuyoAvf+KT82A8Hc74/KnwijyXyi5n3o6WYit
+Y/M58mM135BY5kaLai/Y1bG/v2VEpwsuhgkShXiZtJHAfbRsWITFYD+54EIJVk/
xMs8dDeuExNZdr1wR/DI2xe5WJQ0XImzFb4ZcV5NiE7Ka0KPhXrSOoTxzmZEOolo
3fRalZaAXfc=
=rSeH
-----END PGP MESSAGE-----
```

9) Decrypt the message using receiver's private and public keys on receiver's side.

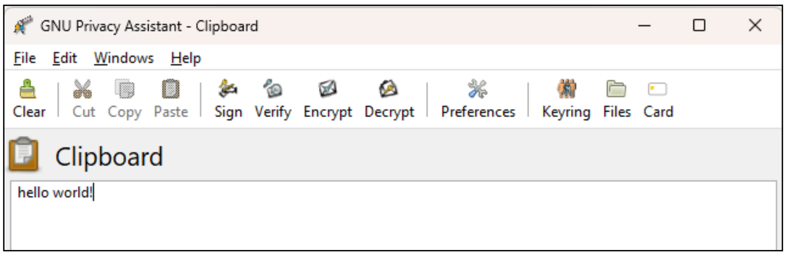

GNU Privacy Assistant - Clipboard
File Edit Windows Help
Clear Cut Copy Paste Sign Verify Encrypt Decrypt Preferences Keyring Files Card
Clipboard
Decrypting...
Clipboard
Cancel
pinentry-qt
OK
Cancel

```
-----BEGIN PGP MESSAGE-----

hQEMA3J0FBohkW0QAQgAxcFtNL4W0IBpFYuM2VNyjOALSHRGjgGnNU8JbQn3auIA
iX1il6Bhcuyc4+5VtxbvqgqsFuiNOVXr5w/mD/o4zDgAmFYyvrHZql30I7dJ5JCA
vW1p0PUgotD+dhHfIAWLesx7+H32e3BBZ6AP92B4rlzXqLbJHxXnLKFFw79MoWOD
Ih4d0IBpsRBJly5LYW/GY0mVbOyyKhd9fBE6DEXsMni91nwTMkoZiMOK58u3UDUC
z0pWPeHzS5b8c/mTlevcZhUGEK4TuyoAvf+K                         Yit
+Y/M58mM135BY5kaLai/Y1bG/v2VEpwsuhgk                         Vk/
xMs8dDeuExNZdr1wR/DI2xe5WJQ0XImzFb4Z                         Oolo
3fRalZaAXfc=
=rSeH
-----END PGP MESSAGE-----
```

Please enter the passphrase to unlock the OpenPGP secret key:
"reciver <reciver@gmail.com>"
2048-bit RSA key, ID 7274141A21916D10,
created 2023-02-23 (main key ID A537A94FE9E1CF16).

Passphrase: ••••••••••••••

10) Message is decrypted and original message is shown in the figure below.


GNU Privacy Assistant - Clipboard
File Edit Windows Help
Clear Cut Copy Paste Sign Verify Encrypt Decrypt Preferences Keyring Files Card
Clipboard

hello world!

Conclusion/Summary:		
Student Signature & Date	**Marks:**	**Evaluator Signature & Date**

Practical 6

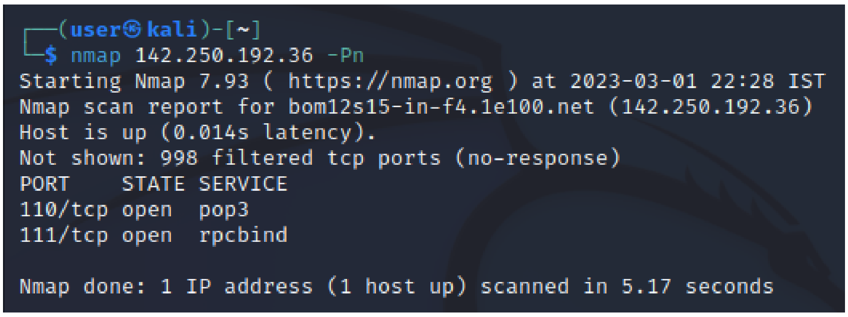
Date: / /2023

Aim: Configure Nmap (Network mapping tool) on Linux/Windows. Explore the various command for scanning your host/ips, ports and various services running on port. Prepare the document of at least 25 Nmap commands. Use the Nmap script and launch the DoS attack by flooding the packages in regular interval.

Solution:

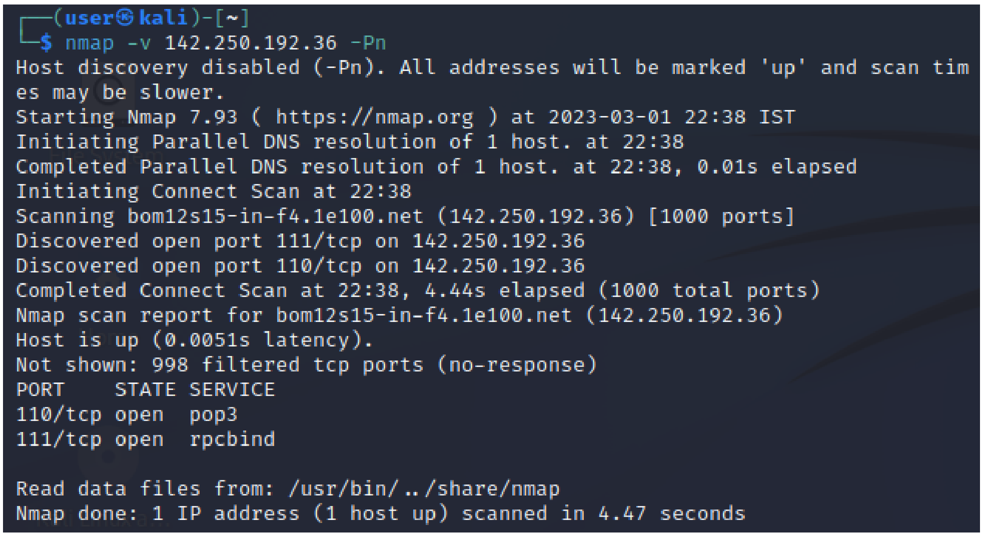
Steps:

1) **To scan a System with its IP address.**



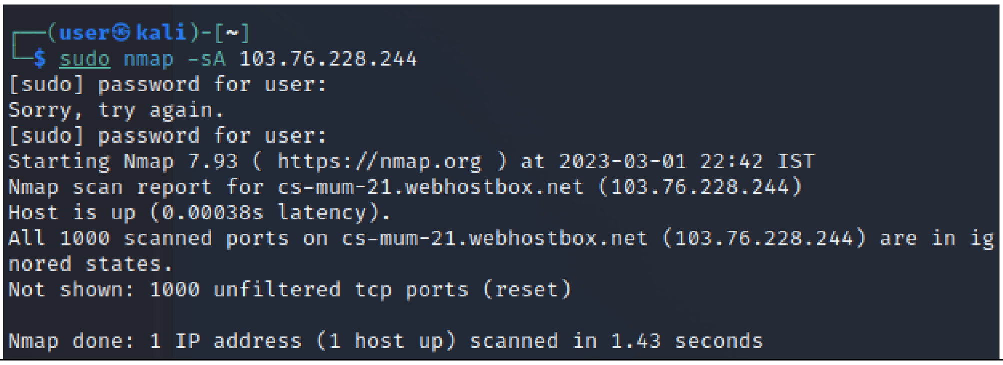
```
┌──(user㉿kali)-[~]
└─$ nmap 142.250.192.36 -Pn
Starting Nmap 7.93 ( https://nmap.org ) at 2023-03-01 22:28 IST
Nmap scan report for bom12s15-in-f4.1e100.net (142.250.192.36)
Host is up (0.014s latency).
Not shown: 998 filtered tcp ports (no-response)
PORT    STATE SERVICE
110/tcp open  pop3
111/tcp open  rpcbind

Nmap done: 1 IP address (1 host up) scanned in 5.17 seconds
```

2) **To scan using "-v" option.**



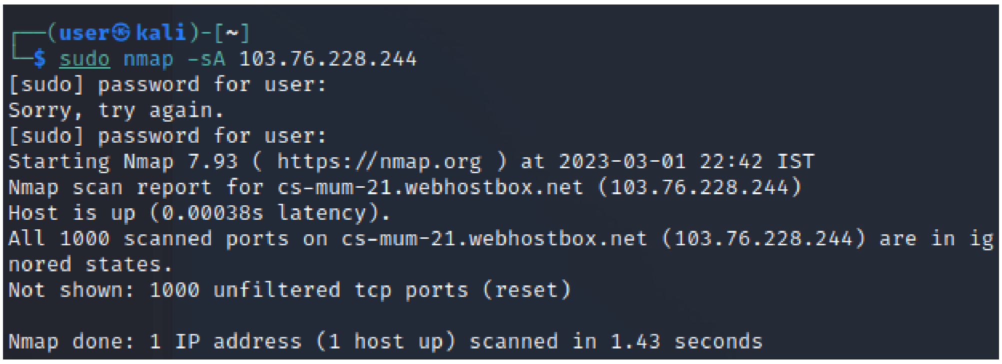
```
┌──(user㉿kali)-[~]
└─$ nmap -v 142.250.192.36 -Pn
Host discovery disabled (-Pn). All addresses will be marked 'up' and scan tim
es may be slower.
Starting Nmap 7.93 ( https://nmap.org ) at 2023-03-01 22:38 IST
Initiating Parallel DNS resolution of 1 host. at 22:38
Completed Parallel DNS resolution of 1 host. at 22:38, 0.01s elapsed
Initiating Connect Scan at 22:38
Scanning bom12s15-in-f4.1e100.net (142.250.192.36) [1000 ports]
Discovered open port 111/tcp on 142.250.192.36
Discovered open port 110/tcp on 142.250.192.36
Completed Connect Scan at 22:38, 4.44s elapsed (1000 total ports)
Nmap scan report for bom12s15-in-f4.1e100.net (142.250.192.36)
Host is up (0.0051s latency).
Not shown: 998 filtered tcp ports (no-response)
PORT    STATE SERVICE
110/tcp open  pop3
111/tcp open  rpcbind

Read data files from: /usr/bin/../share/nmap
Nmap done: 1 IP address (1 host up) scanned in 4.47 seconds
```

3) **To scan to detect firewall settings.**



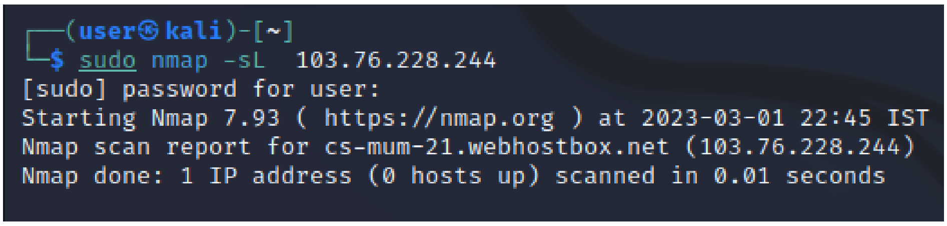
```
┌──(user㉿kali)-[~]
└─$ sudo nmap -sA 103.76.228.244
[sudo] password for user:
Sorry, try again.
[sudo] password for user:
Starting Nmap 7.93 ( https://nmap.org ) at 2023-03-01 22:42 IST
Nmap scan report for cs-mum-21.webhostbox.net (103.76.228.244)
Host is up (0.00038s latency).
All 1000 scanned ports on cs-mum-21.webhostbox.net (103.76.228.244) are in ig
nored states.
Not shown: 1000 unfiltered tcp ports (reset)

Nmap done: 1 IP address (1 host up) scanned in 1.43 seconds
```

4) To scan multiple hosts



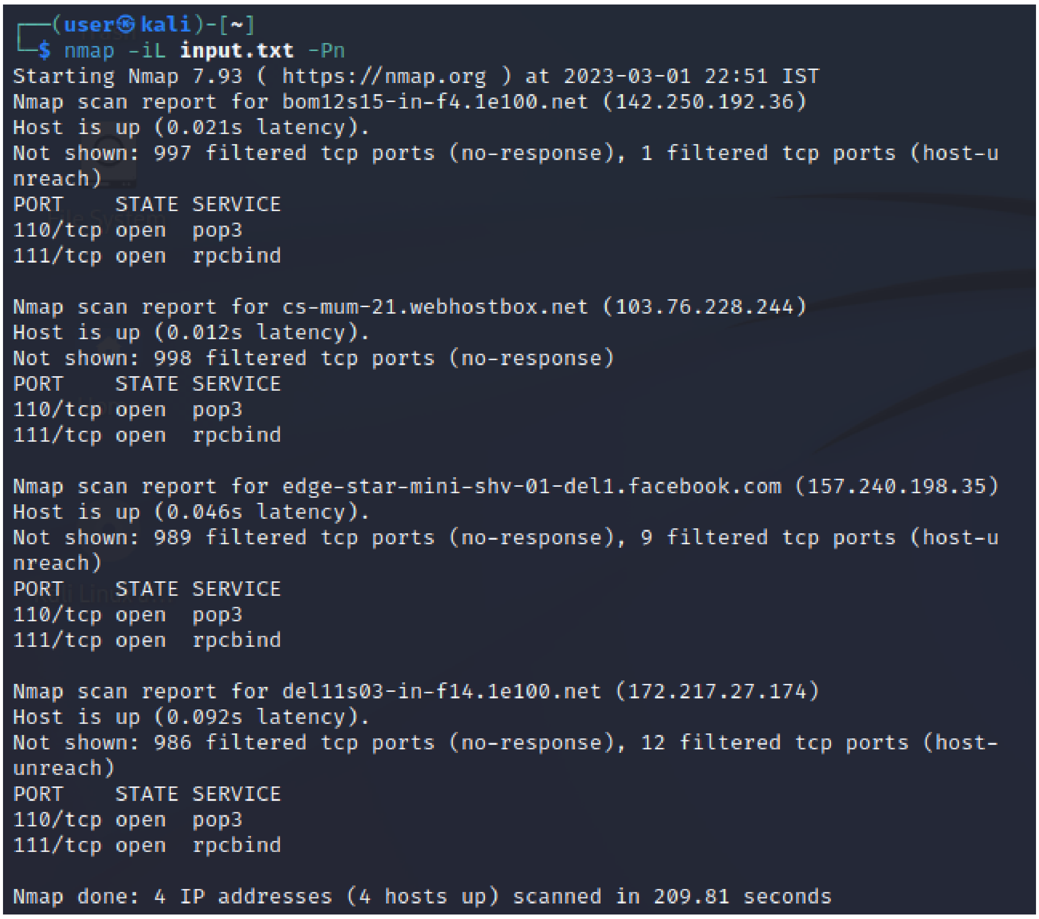
```
┌──(user㉿kali)-[~]
└─$ sudo nmap -sA 103.76.228.244
[sudo] password for user:
Sorry, try again.
[sudo] password for user:
Starting Nmap 7.93 ( https://nmap.org ) at 2023-03-01 22:42 IST
Nmap scan report for cs-mum-21.webhostbox.net (103.76.228.244)
Host is up (0.00038s latency).
All 1000 scanned ports on cs-mum-21.webhostbox.net (103.76.228.244) are in ig
nored states.
Not shown: 1000 unfiltered tcp ports (reset)

Nmap done: 1 IP address (1 host up) scanned in 1.43 seconds
```

5) To identify hostnames



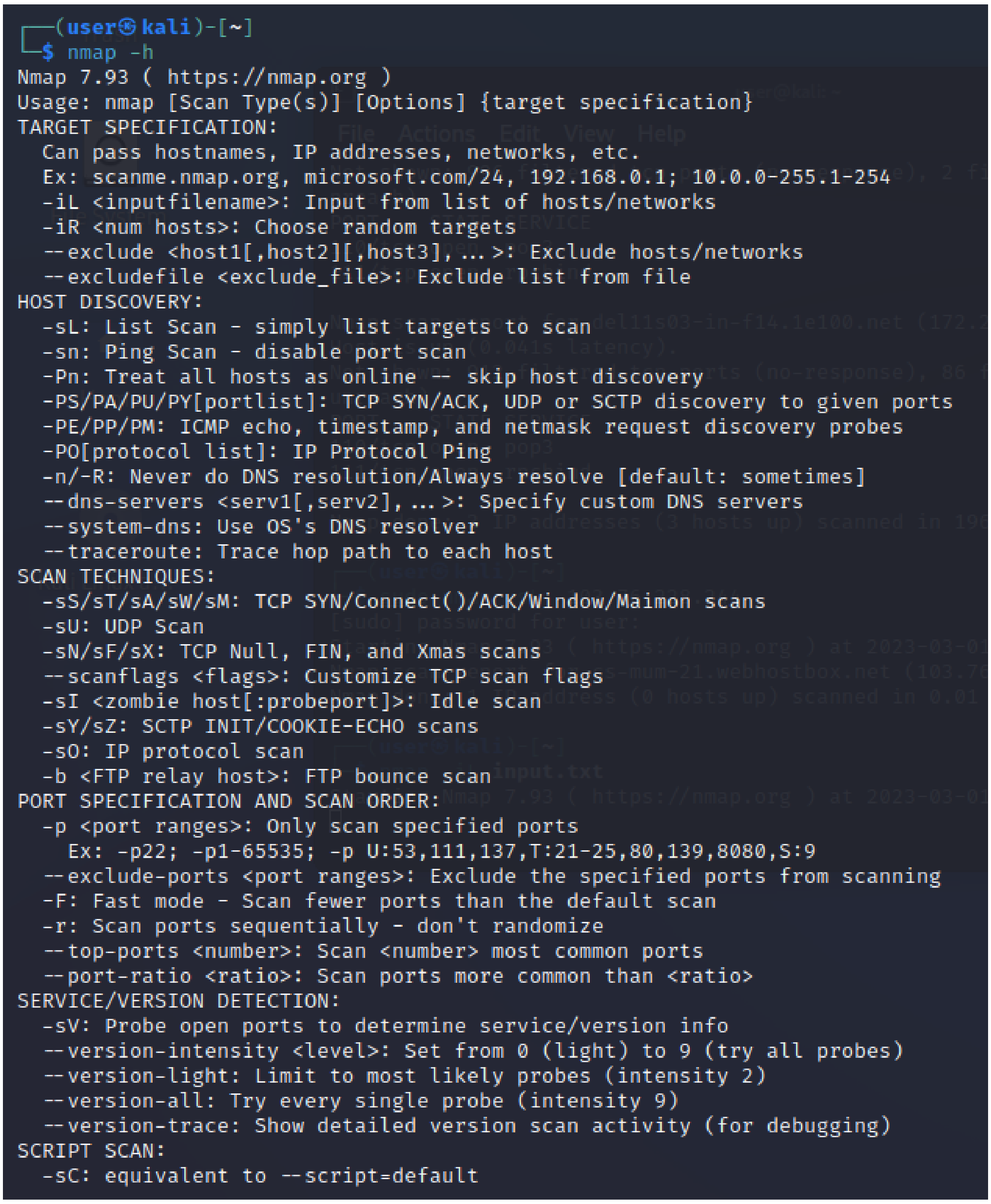
```
┌──(user㉿kali)-[~]
└─$ sudo nmap -sL  103.76.228.244
[sudo] password for user:
Starting Nmap 7.93 ( https://nmap.org ) at 2023-03-01 22:45 IST
Nmap scan report for cs-mum-21.webhostbox.net (103.76.228.244)
Nmap done: 1 IP address (0 hosts up) scanned in 0.01 seconds
```

6) To scan from a file



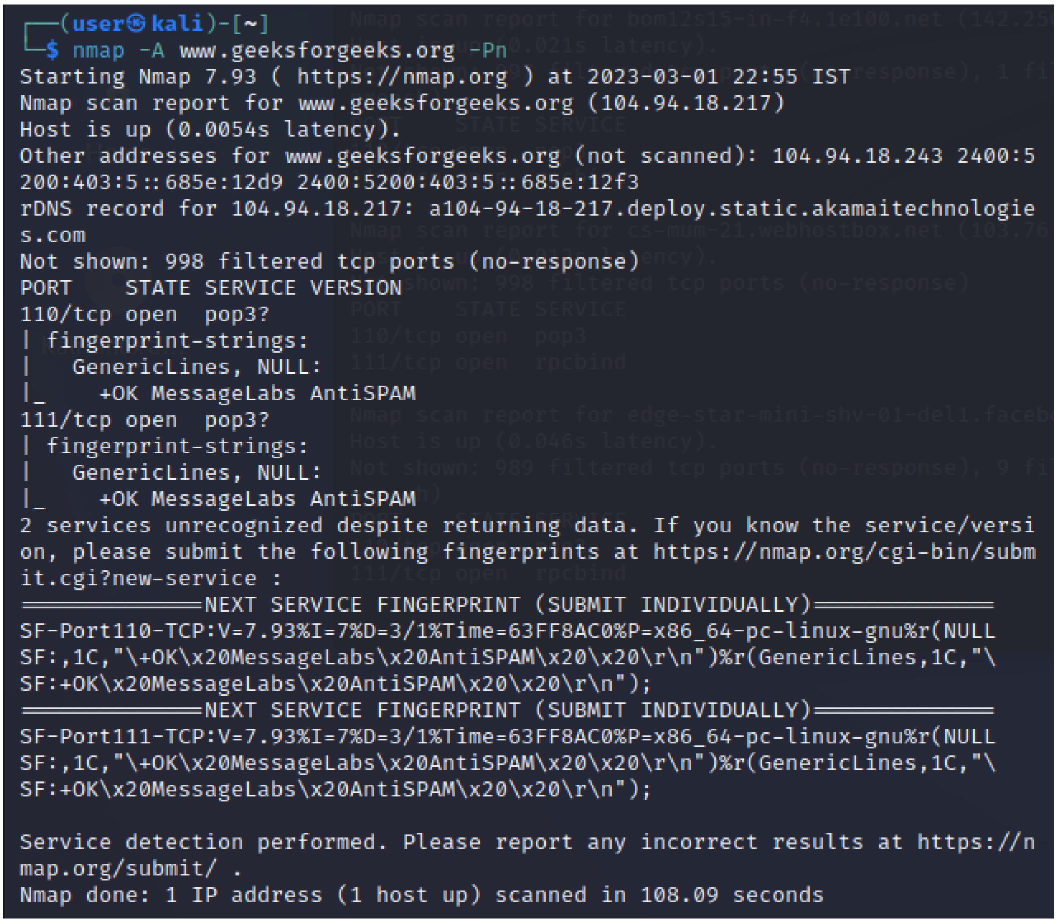
```
┌──(user㉿kali)-[~]
└─$ nmap -iL input.txt -Pn
Starting Nmap 7.93 ( https://nmap.org ) at 2023-03-01 22:51 IST
Nmap scan report for bom12s15-in-f4.1e100.net (142.250.192.36)
Host is up (0.021s latency).
Not shown: 997 filtered tcp ports (no-response), 1 filtered tcp ports (host-u
nreach)
PORT    STATE SERVICE
110/tcp open  pop3
111/tcp open  rpcbind

Nmap scan report for cs-mum-21.webhostbox.net (103.76.228.244)
Host is up (0.012s latency).
Not shown: 998 filtered tcp ports (no-response)
PORT    STATE SERVICE
110/tcp open  pop3
111/tcp open  rpcbind

Nmap scan report for edge-star-mini-shv-01-del1.facebook.com (157.240.198.35)
Host is up (0.046s latency).
Not shown: 989 filtered tcp ports (no-response), 9 filtered tcp ports (host-u
nreach)
PORT    STATE SERVICE
110/tcp open  pop3
111/tcp open  rpcbind

Nmap scan report for del11s03-in-f14.1e100.net (172.217.27.174)
Host is up (0.092s latency).
Not shown: 986 filtered tcp ports (no-response), 12 filtered tcp ports (host-
unreach)
PORT    STATE SERVICE
110/tcp open  pop3
111/tcp open  rpcbind

Nmap done: 4 IP addresses (4 hosts up) scanned in 209.81 seconds
```

7) To get some help.



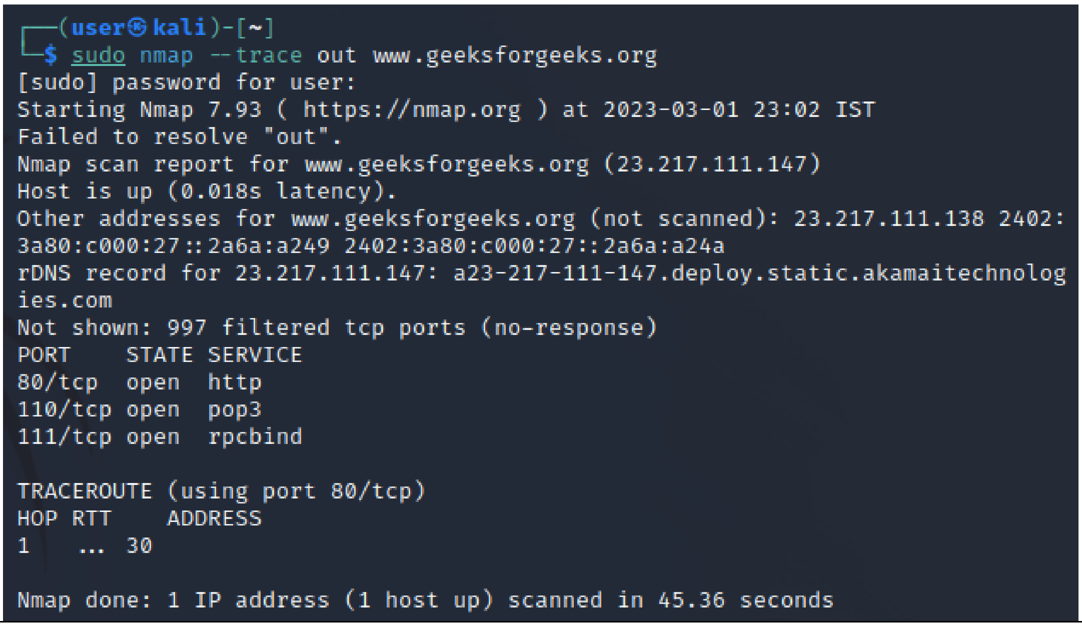
```
┌──(user㉿kali)-[~]
└─$ nmap -h
Nmap 7.93 ( https://nmap.org )
Usage: nmap [Scan Type(s)] [Options] {target specification}
TARGET SPECIFICATION:
  Can pass hostnames, IP addresses, networks, etc.
  Ex: scanme.nmap.org, microsoft.com/24, 192.168.0.1; 10.0.0-255.1-254
  -iL <inputfilename>: Input from list of hosts/networks
  -iR <num hosts>: Choose random targets
  --exclude <host1[,host2][,host3],...>: Exclude hosts/networks
  --excludefile <exclude_file>: Exclude list from file
HOST DISCOVERY:
  -sL: List Scan - simply list targets to scan
  -sn: Ping Scan - disable port scan
  -Pn: Treat all hosts as online -- skip host discovery
  -PS/PA/PU/PY[portlist]: TCP SYN/ACK, UDP or SCTP discovery to given ports
  -PE/PP/PM: ICMP echo, timestamp, and netmask request discovery probes
  -PO[protocol list]: IP Protocol Ping
  -n/-R: Never do DNS resolution/Always resolve [default: sometimes]
  --dns-servers <serv1[,serv2],...>: Specify custom DNS servers
  --system-dns: Use OS's DNS resolver
  --traceroute: Trace hop path to each host
SCAN TECHNIQUES:
  -sS/sT/sA/sW/sM: TCP SYN/Connect()/ACK/Window/Maimon scans
  -sU: UDP Scan
  -sN/sF/sX: TCP Null, FIN, and Xmas scans
  --scanflags <flags>: Customize TCP scan flags
  -sI <zombie host[:probeport]>: Idle scan
  -sY/sZ: SCTP INIT/COOKIE-ECHO scans
  -sO: IP protocol scan
  -b <FTP relay host>: FTP bounce scan
PORT SPECIFICATION AND SCAN ORDER:
  -p <port ranges>: Only scan specified ports
    Ex: -p22; -p1-65535; -p U:53,111,137,T:21-25,80,139,8080,S:9
  --exclude-ports <port ranges>: Exclude the specified ports from scanning
  -F: Fast mode - Scan fewer ports than the default scan
  -r: Scan ports sequentially - don't randomize
  --top-ports <number>: Scan <number> most common ports
  --port-ratio <ratio>: Scan ports more common than <ratio>
SERVICE/VERSION DETECTION:
  -sV: Probe open ports to determine service/version info
  --version-intensity <level>: Set from 0 (light) to 9 (try all probes)
  --version-light: Limit to most likely probes (intensity 2)
  --version-all: Try every single probe (intensity 9)
  --version-trace: Show detailed version scan activity (for debugging)
SCRIPT SCAN:
  -sC: equivalent to --script=default
```

8) **Here -A indicates aggressive, it will give us extra information, like OS detection (-O), version detection, script scanning (-sC), and traceroute (–traceroute). It even provides a lot of valuable information about the host.**



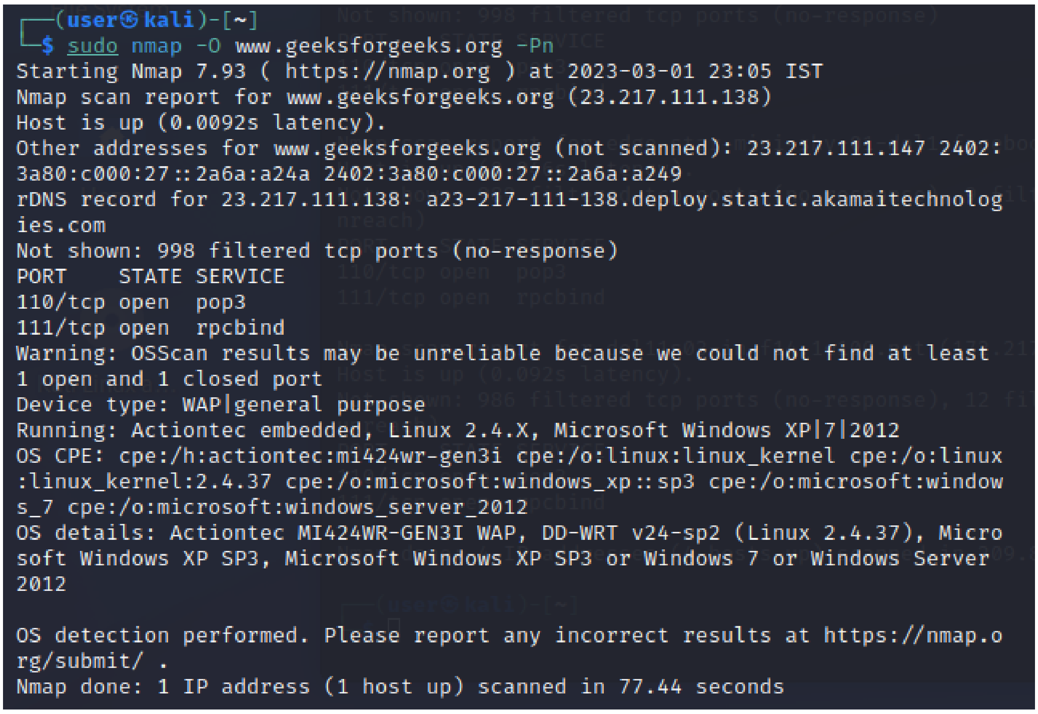
```
┌──(user㉿kali)-[~]
└─$ nmap -A www.geeksforgeeks.org -Pn
Starting Nmap 7.93 ( https://nmap.org ) at 2023-03-01 22:55 IST
Nmap scan report for www.geeksforgeeks.org (104.94.18.217)
Host is up (0.0054s latency).
Other addresses for www.geeksforgeeks.org (not scanned): 104.94.18.243 2400:5
200:403:5::685e:12d9 2400:5200:403:5::685e:12f3
rDNS record for 104.94.18.217: a104-94-18-217.deploy.static.akamaitechnologie
s.com
Not shown: 998 filtered tcp ports (no-response)
PORT    STATE SERVICE VERSION
110/tcp open  pop3?
| fingerprint-strings: 
|   GenericLines, NULL: 
|_    +OK MessageLabs AntiSPAM
111/tcp open  pop3?
| fingerprint-strings: 
|   GenericLines, NULL: 
|_    +OK MessageLabs AntiSPAM
2 services unrecognized despite returning data. If you know the service/versi
on, please submit the following fingerprints at https://nmap.org/cgi-bin/subm
it.cgi?new-service :
==============NEXT SERVICE FINGERPRINT (SUBMIT INDIVIDUALLY)==============
SF-Port110-TCP:V=7.93%I=7%D=3/1%Time=63FF8AC0%P=x86_64-pc-linux-gnu%r(NULL
SF:,1C,"\+OK\x20MessageLabs\x20AntiSPAM\x20\x20\r\n")%r(GenericLines,1C,"\
SF:+OK\x20MessageLabs\x20AntiSPAM\x20\x20\r\n");
==============NEXT SERVICE FINGERPRINT (SUBMIT INDIVIDUALLY)==============
SF-Port111-TCP:V=7.93%I=7%D=3/1%Time=63FF8AC0%P=x86_64-pc-linux-gnu%r(NULL
SF:,1C,"\+OK\x20MessageLabs\x20AntiSPAM\x20\x20\r\n")%r(GenericLines,1C,"\
SF:+OK\x20MessageLabs\x20AntiSPAM\x20\x20\r\n");

Service detection performed. Please report any incorrect results at https://n
map.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 108.09 seconds
```

9) **Using this command we can discover the target hosting service or identify additional targets according to our needs for quickly tracing the path.**



```
┌──(user㉿kali)-[~]
└─$ sudo nmap --trace out www.geeksforgeeks.org
[sudo] password for user: 
Starting Nmap 7.93 ( https://nmap.org ) at 2023-03-01 23:02 IST
Failed to resolve "out".
Nmap scan report for www.geeksforgeeks.org (23.217.111.147)
Host is up (0.018s latency).
Other addresses for www.geeksforgeeks.org (not scanned): 23.217.111.138 2402:
3a80:c000:27::2a6a:a249 2402:3a80:c000:27::2a6a:a24a
rDNS record for 23.217.111.147: a23-217-111-147.deploy.static.akamaitechnolog
ies.com
Not shown: 997 filtered tcp ports (no-response)
PORT    STATE SERVICE
80/tcp  open  http
110/tcp open  pop3
111/tcp open  rpcbind

TRACEROUTE (using port 80/tcp)
HOP RTT    ADDRESS
1   ... 30

Nmap done: 1 IP address (1 host up) scanned in 45.36 seconds
```

10) **Here it will display the operating system where the domain or ip address is running, but will not display the exact operating system available on the computer. It will display only the chance of operating system available in the computer. The command will just guess the running operating system (OS) on the host.**



```
┌──(user㉿kali)-[~]
└─$ sudo nmap -O www.geeksforgeeks.org -Pn
Starting Nmap 7.93 ( https://nmap.org ) at 2023-03-01 23:05 IST
Nmap scan report for www.geeksforgeeks.org (23.217.111.138)
Host is up (0.0092s latency).
Other addresses for www.geeksforgeeks.org (not scanned): 23.217.111.147 2402:
3a80:c000:27::2a6a:a24a 2402:3a80:c000:27::2a6a:a249
rDNS record for 23.217.111.138: a23-217-111-138.deploy.static.akamaitechnolog
ies.com
Not shown: 998 filtered tcp ports (no-response)
PORT    STATE SERVICE
110/tcp open  pop3
111/tcp open  rpcbind
Warning: OSScan results may be unreliable because we could not find at least
1 open and 1 closed port
Device type: WAP|general purpose
Running: Actiontec embedded, Linux 2.4.X, Microsoft Windows XP|7|2012
OS CPE: cpe:/h:actiontec:mi424wr-gen3i cpe:/o:linux:linux_kernel cpe:/o:linux
:linux_kernel:2.4.37 cpe:/o:microsoft:windows_xp::sp3 cpe:/o:microsoft:window
s_7 cpe:/o:microsoft:windows_server_2012
OS details: Actiontec MI424WR-GEN3I WAP, DD-WRT v24-sp2 (Linux 2.4.37), Micro
soft Windows XP SP3, Microsoft Windows XP SP3 or Windows 7 or Windows Server
2012

OS detection performed. Please report any incorrect results at https://nmap.o
rg/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 77.44 seconds
```

Conclusion/Summary:

Student Signature & Date	**Marks:**	**Evaluator Signature & Date**

Practical 7

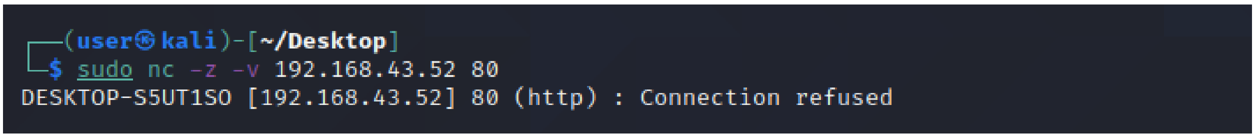
Date: / /2023

Aim: Perform Port Scanning, File Transfer, Client-server chat and Basic Webserver implementation using netcat. Find the service running on the particular port using netcat.

Solution:

Port Scanning:

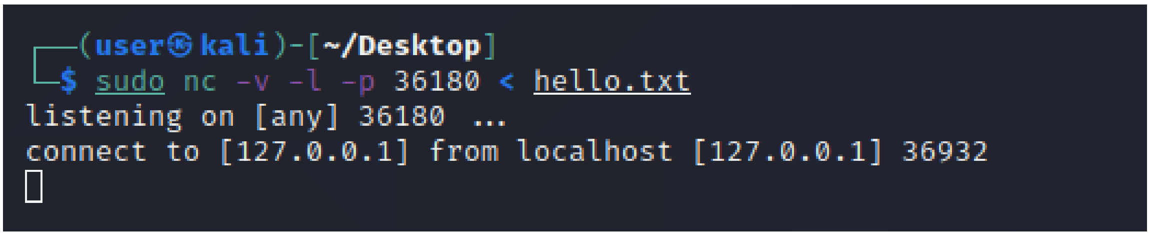
- This is useful to know which ports are open and running services on a target machine.
- The -z flag can be used to tell nc to report open ports, rather than initiate a connection.



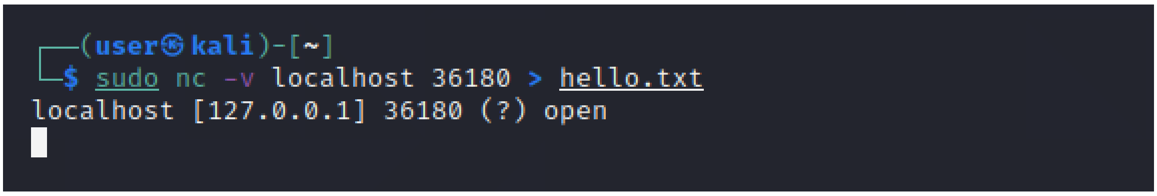
```
┌──(user㉿kali)-[~/Desktop]
└─$ sudo nc -z -v 192.168.43.52 80
DESKTOP-S5UT1SO [192.168.43.52] 80 (http) : Connection refused
```

File Transfer:

- The nc (netcat) command can be used to transfer arbitrary data over the network.
- It represents a quick way for Linux administrators to transfer data without the need for an additional data transfer services such as FTP, HTTP, SCP etc.



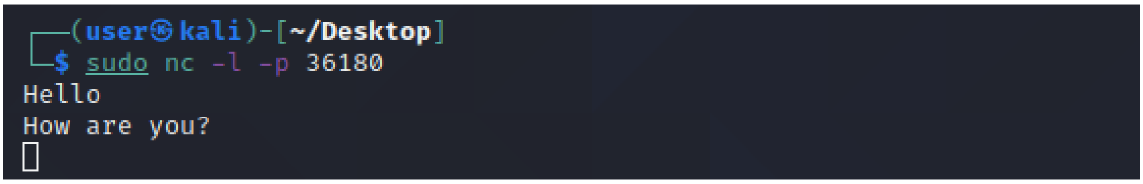
```
┌──(user㉿kali)-[~/Desktop]
└─$ sudo nc -v -l -p 36180 < hello.txt
listening on [any] 36180 ...
connect to [127.0.0.1] from localhost [127.0.0.1] 36932
```



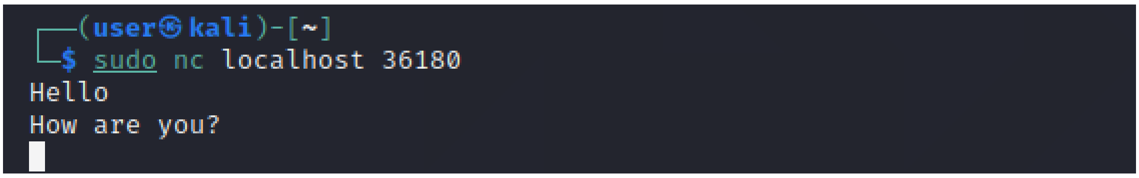
```
┌──(user㉿kali)-[~]
└─$ sudo nc -v localhost 36180 > hello.txt
localhost [127.0.0.1] 36180 (?) open
```

Client-Server Chat:

- To create a simple chat we need two instances of netcat, one to listen for incoming connections (the server) and another one to start the connection.



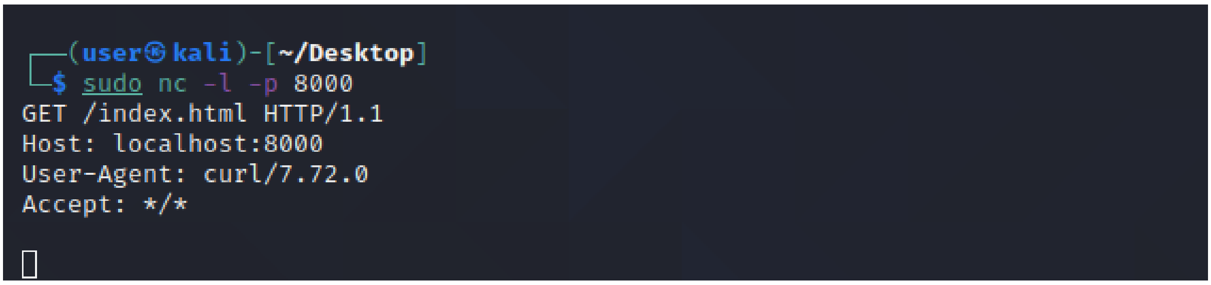
```
┌──(user㉿kali)-[~/Desktop]
└─$ sudo nc -l -p 36180
Hello
How are you?
```



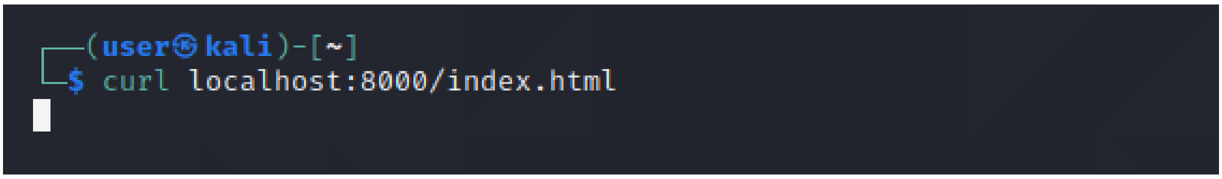
```
┌──(user㉿kali)-[~]
└─$ sudo nc localhost 36180
Hello
How are you?
```

Basic Webserver Implementation:

- The netcat tool nc can operate as a TCP client. Because HTTP works over TCP, nc can be used as an HTTP server!
- Because nc is a UNIX tool, we can use it to make custom web servers: servers which return any HTTP headers you want, servers which return the response very slowly, servers which return invalid HTTP, etc.



```
┌──(user㉿kali)-[~/Desktop]
└─$ sudo nc -l -p 8000
GET /index.html HTTP/1.1
Host: localhost:8000
User-Agent: curl/7.72.0
Accept: */*

```



```
┌──(user㉿kali)-[~]
└─$ curl localhost:8000/index.html
```

Conclusion/Summary:

Student Signature & Date	**Marks:**	**Evaluator Signature & Date**

Practical 8

Date: / /2023

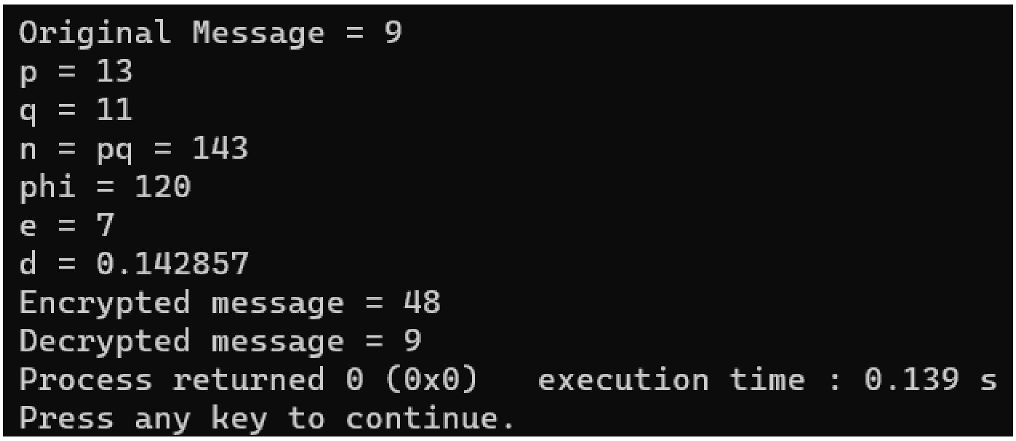
Aim: Bob is going to send his encrypted file using public key shared by Alice using Public-key infrastructure. Alice will decrypt the file by using her private key and ensure the confidentiality. Implement the given scenario using RSA algorithm.

Solution:

Code:

```
#include<iostream>
#include<math.h>
using namespace std;
int gcd(int a, int b) {
  int t;
  while(1) {
    t= a%b;
    if(t==0)
    return b;
    a = b;
    b = t;
  }
}

int main() {
  double p = 13;
  double q = 11;
  double n=p*q;
  double track;
  double phi= (p-1)*(q-1);
  double e=7;
  while(e<phi) {
    track = gcd(e,phi);
    if(track==1)
      break;
    else
      e++;
  }
  double d1=1/e;
  double d=fmod(d1,phi);
  double message = 9;
  double c = pow(message,e);
  double m = pow(c,d);
  c=fmod(c,n);
  m=fmod(m,n);
  cout<<"Original Message = "<<message;
  cout<<"\n"<<"p = "<<p;
  cout<<"\n"<<"q = "<<q;
  cout<<"\n"<<"n = pq = "<<n;
  cout<<"\n"<<"phi = "<<phi;
  cout<<"\n"<<"e = "<<e;
  cout<<"\n"<<"d = "<<d;
  cout<<"\n"<<"Encrypted message = "<<c;
  cout<<"\n"<<"Decrypted message = "<<m;

  return 0;
}
```

Output:



```
Original Message = 9
p = 13
q = 11
n = pq = 143
phi = 120
e = 7
d = 0.142857
Encrypted message = 48
Decrypted message = 9
Process returned 0 (0x0)   execution time : 0.139 s
Press any key to continue.
```

Conclusion/Summary:

Student Signature & Date	**Marks:**	**Evaluator Signature & Date**

Practical 9

Date: / /2023

Aim: In computers, Foot printing is the process of accumulating data regarding a specific network environment, usually for the purpose of finding ways to intrude into the environment. Foot printing can reveal system vulnerabilities and improve the ease with which they can be exploited.
Use the given approach to implement Footprinting: Gathering Target Information making use of following tools:

- Dmitry – Deepmagic
- Whatweb

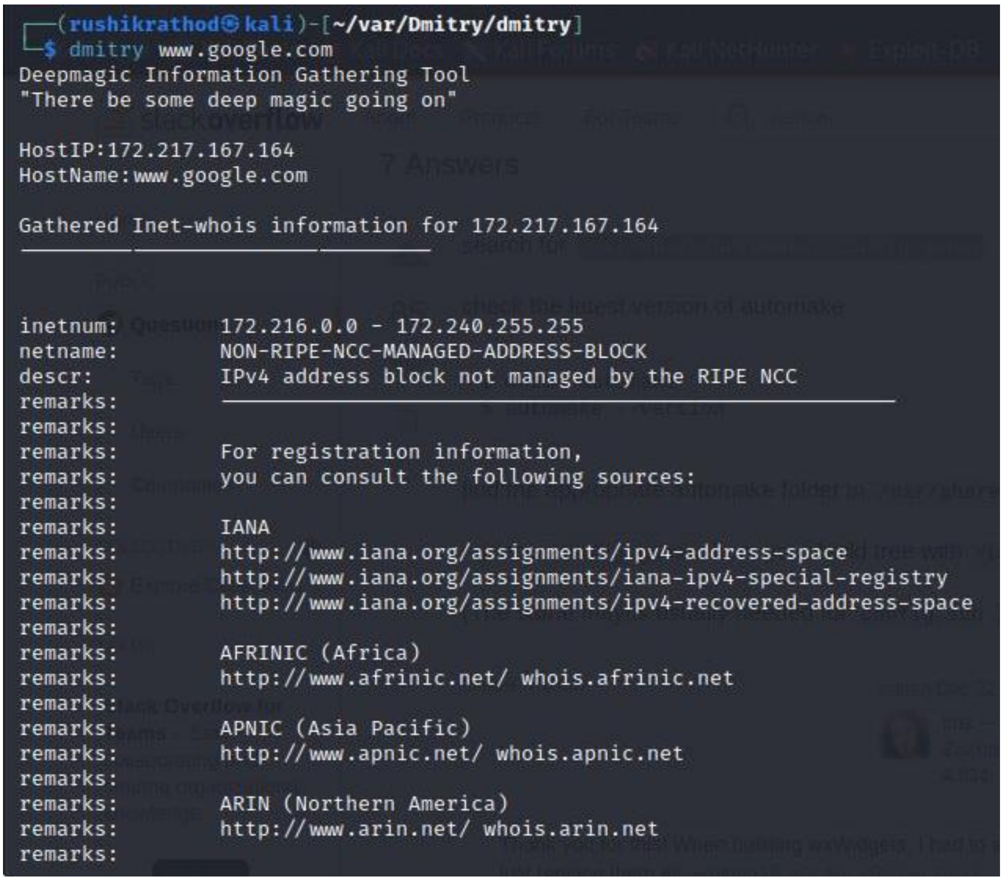
Solution:

Dmitry – Deepmagic:

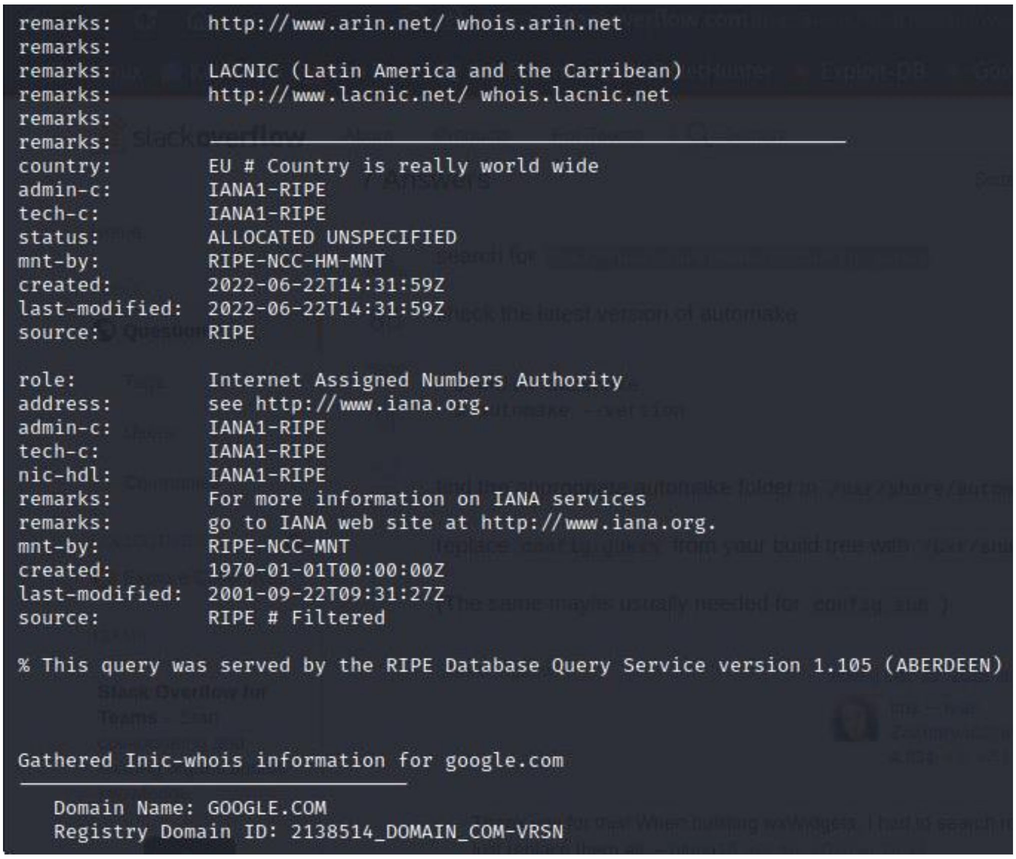
Dmitry is a free and open-source tool available on GitHub. The tool is used for information gathering. You can download the tool and install in your Kali Linux. Dmitry stands for DeepMagic Information Gathering Tool. It’s a command-line tool Using Dmitry tool You can collect information about the target, this information can be used for social engineering attacks. It can be used to gather a number of valuable pieces of information

Usages of Dmitry Tool :

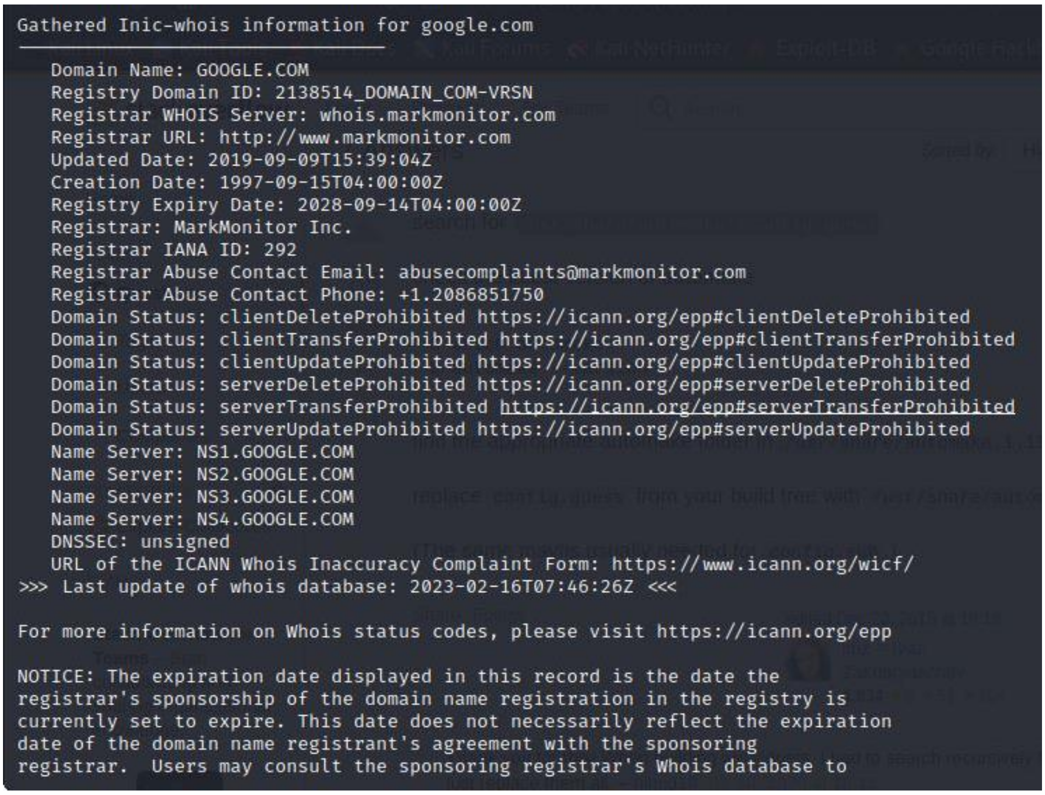
- Dmitry Tool can be used to search subdomains of the target.
- Dmitry Tool can be used to find open ports of the target system.
- Dmitry Tool can be used to perform TCP scan.
- Dmitry Tool can be used with netcraft service to get the target information such as operating system, web server details, web host details, hosting service details, etc.
- Dmitry Tool can be used with whois service to get the target information such as registered domain, name, address, the contact information of the person who registered it.
- Dmitry Tool can be used to get email addresses that are associated with the domain of the target.



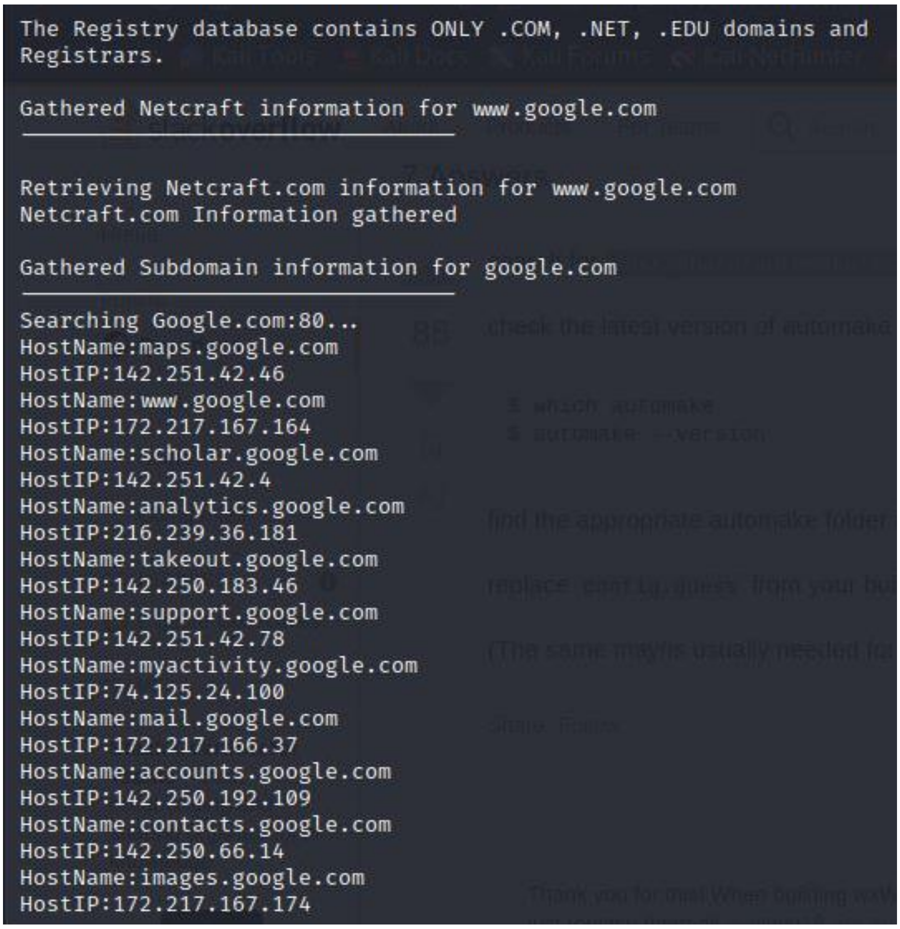
```
┌──(rushikrathod㉿kali)-[~/var/Dmitry/dmitry]
└─$ dmitry www.google.com
Deepmagic Information Gathering Tool
"There be some deep magic going on"

HostIP:172.217.167.164
HostName:www.google.com

Gathered Inet-whois information for 172.217.167.164
─────────────────────────────────


inetnum:        172.216.0.0 - 172.240.255.255
netname:        NON-RIPE-NCC-MANAGED-ADDRESS-BLOCK
descr:          IPv4 address block not managed by the RIPE NCC
remarks:        ──────────────────────────────────────
remarks:
remarks:        For registration information,
remarks:        you can consult the following sources:
remarks:
remarks:        IANA
remarks:        http://www.iana.org/assignments/ipv4-address-space
remarks:        http://www.iana.org/assignments/iana-ipv4-special-registry
remarks:        http://www.iana.org/assignments/ipv4-recovered-address-space
remarks:
remarks:        AFRINIC (Africa)
remarks:        http://www.afrinic.net/ whois.afrinic.net
remarks:
remarks:        APNIC (Asia Pacific)
remarks:        http://www.apnic.net/ whois.apnic.net
remarks:
remarks:        ARIN (Northern America)
remarks:        http://www.arin.net/ whois.arin.net
remarks:
```



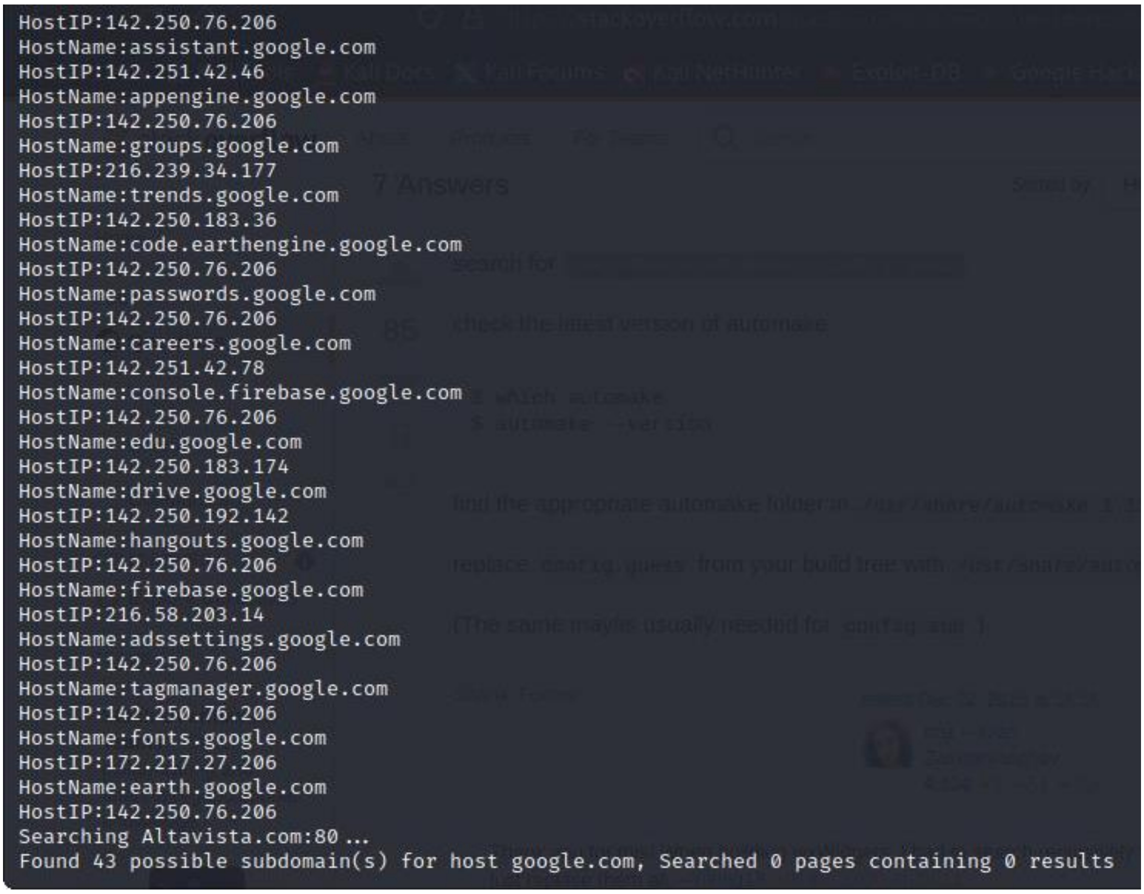
```
remarks:        http://www.arin.net/ whois.arin.net
remarks:
remarks:        LACNIC (Latin America and the Carribean)
remarks:        http://www.lacnic.net/ whois.lacnic.net
remarks:
remarks:        ──────────────────────────────────────
country:        EU # Country is really world wide
admin-c:        IANA1-RIPE
tech-c:         IANA1-RIPE
status:         ALLOCATED UNSPECIFIED
mnt-by:         RIPE-NCC-HM-MNT
created:        2022-06-22T14:31:59Z
last-modified:  2022-06-22T14:31:59Z
source:         RIPE

role:           Internet Assigned Numbers Authority
address:        see http://www.iana.org.
admin-c:        IANA1-RIPE
tech-c:         IANA1-RIPE
nic-hdl:        IANA1-RIPE
remarks:        For more information on IANA services
remarks:        go to IANA web site at http://www.iana.org.
mnt-by:         RIPE-NCC-MNT
created:        1970-01-01T00:00:00Z
last-modified:  2001-09-22T09:31:27Z
source:         RIPE # Filtered

% This query was served by the RIPE Database Query Service version 1.105 (ABERDEEN)


Gathered Inic-whois information for google.com
────────────────────────────────
   Domain Name: GOOGLE.COM
   Registry Domain ID: 2138514_DOMAIN_COM-VRSN
```



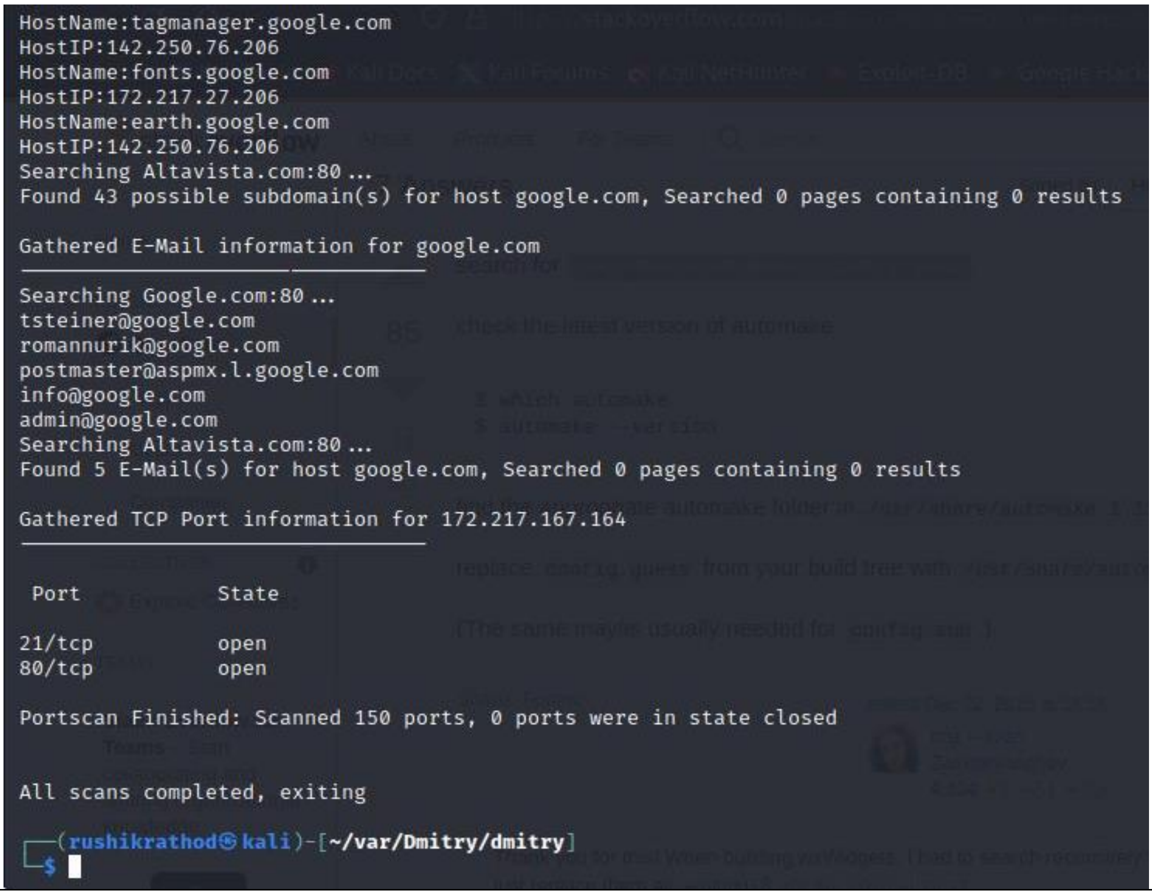
```
Gathered Inic-whois information for google.com
---------------------------------------------

   Domain Name: GOOGLE.COM
   Registry Domain ID: 2138514_DOMAIN_COM-VRSN
   Registrar WHOIS Server: whois.markmonitor.com
   Registrar URL: http://www.markmonitor.com
   Updated Date: 2019-09-09T15:39:04Z
   Creation Date: 1997-09-15T04:00:00Z
   Registry Expiry Date: 2028-09-14T04:00:00Z
   Registrar: MarkMonitor Inc.
   Registrar IANA ID: 292
   Registrar Abuse Contact Email: abusecomplaints@markmonitor.com
   Registrar Abuse Contact Phone: +1.2086851750
   Domain Status: clientDeleteProhibited https://icann.org/epp#clientDeleteProhibited
   Domain Status: clientTransferProhibited https://icann.org/epp#clientTransferProhibited
   Domain Status: clientUpdateProhibited https://icann.org/epp#clientUpdateProhibited
   Domain Status: serverDeleteProhibited https://icann.org/epp#serverDeleteProhibited
   Domain Status: serverTransferProhibited https://icann.org/epp#serverTransferProhibited
   Domain Status: serverUpdateProhibited https://icann.org/epp#serverUpdateProhibited
   Name Server: NS1.GOOGLE.COM
   Name Server: NS2.GOOGLE.COM
   Name Server: NS3.GOOGLE.COM
   Name Server: NS4.GOOGLE.COM
   DNSSEC: unsigned
   URL of the ICANN Whois Inaccuracy Complaint Form: https://www.icann.org/wicf/
>>> Last update of whois database: 2023-02-16T07:46:26Z <<<

For more information on Whois status codes, please visit https://icann.org/epp

NOTICE: The expiration date displayed in this record is the date the
registrar's sponsorship of the domain name registration in the registry is
currently set to expire. This date does not necessarily reflect the expiration
date of the domain name registrant's agreement with the sponsoring
registrar.  Users may consult the sponsoring registrar's Whois database to
```



```
The Registry database contains ONLY .COM, .NET, .EDU domains and
Registrars.

Gathered Netcraft information for www.google.com
------------------------------------------------

Retrieving Netcraft.com information for www.google.com
Netcraft.com Information gathered

Gathered Subdomain information for google.com
---------------------------------------------
Searching Google.com:80...
HostName:maps.google.com
HostIP:142.251.42.46
HostName:www.google.com
HostIP:172.217.167.164
HostName:scholar.google.com
HostIP:142.251.42.4
HostName:analytics.google.com
HostIP:216.239.36.181
HostName:takeout.google.com
HostIP:142.250.183.46
HostName:support.google.com
HostIP:142.251.42.78
HostName:myactivity.google.com
HostIP:74.125.24.100
HostName:mail.google.com
HostIP:172.217.166.37
HostName:accounts.google.com
HostIP:142.250.192.109
HostName:contacts.google.com
HostIP:142.250.66.14
HostName:images.google.com
HostIP:172.217.167.174
```



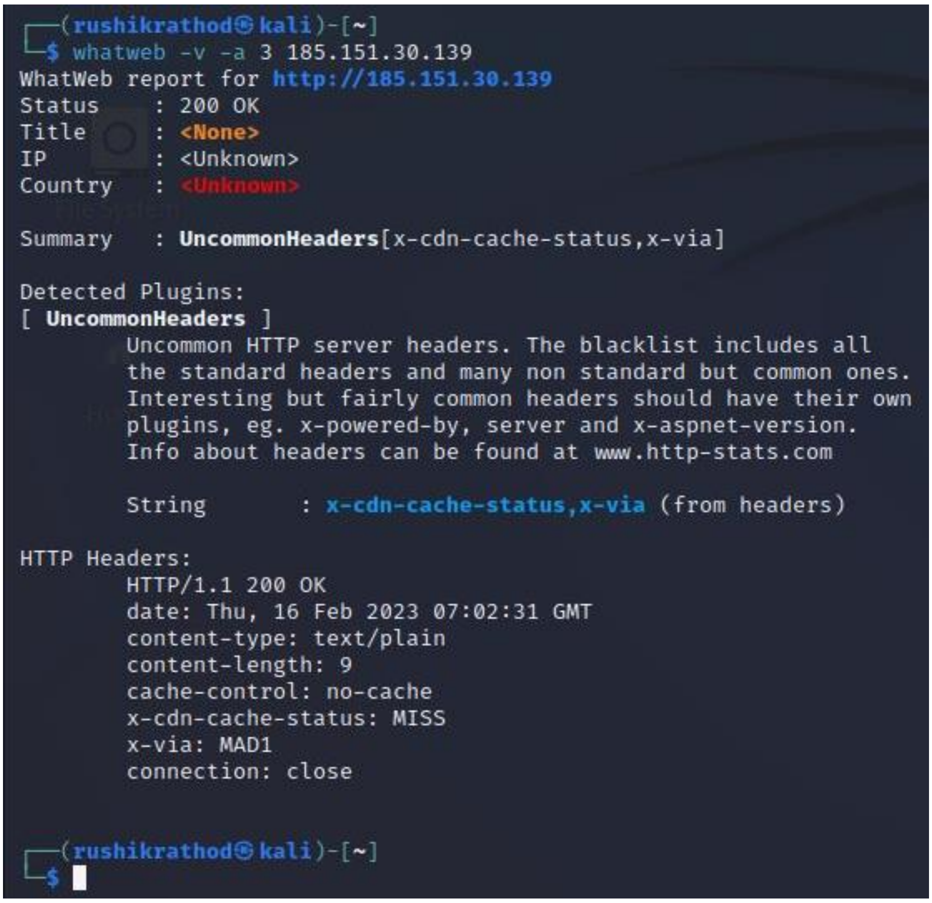
```
HostIP:142.250.76.206
HostName:assistant.google.com
HostIP:142.251.42.46
HostName:appengine.google.com
HostIP:142.250.76.206
HostName:groups.google.com
HostIP:216.239.34.177
HostName:trends.google.com
HostIP:142.250.183.36
HostName:code.earthengine.google.com
HostIP:142.250.76.206
HostName:passwords.google.com
HostIP:142.250.76.206
HostName:careers.google.com
HostIP:142.251.42.78
HostName:console.firebase.google.com
HostIP:142.250.76.206
HostName:edu.google.com
HostIP:142.250.183.174
HostName:drive.google.com
HostIP:142.250.192.142
HostName:hangouts.google.com
HostIP:142.250.76.206
HostName:firebase.google.com
HostIP:216.58.203.14
HostName:adssettings.google.com
HostIP:142.250.76.206
HostName:tagmanager.google.com
HostIP:142.250.76.206
HostName:fonts.google.com
HostIP:172.217.27.206
HostName:earth.google.com
HostIP:142.250.76.206
Searching Altavista.com:80 ...
Found 43 possible subdomain(s) for host google.com, Searched 0 pages containing 0 results
```



```
HostName:tagmanager.google.com
HostIP:142.250.76.206
HostName:fonts.google.com
HostIP:172.217.27.206
HostName:earth.google.com
HostIP:142.250.76.206
Searching Altavista.com:80 ...
Found 43 possible subdomain(s) for host google.com, Searched 0 pages containing 0 results

Gathered E-Mail information for google.com
---------------------------------

Searching Google.com:80 ...
tsteiner@google.com
romannurik@google.com
postmaster@aspmx.l.google.com
info@google.com
admin@google.com
Searching Altavista.com:80 ...
Found 5 E-Mail(s) for host google.com, Searched 0 pages containing 0 results

Gathered TCP Port information for 172.217.167.164
---------------------------------

 Port           State

21/tcp          open
80/tcp          open

Portscan Finished: Scanned 150 ports, 0 ports were in state closed


All scans completed, exiting

┌──(rushikrathod㉿kali)-[~/var/Dmitry/dmitry]
└─$ ▌
```

Whatweb:

Whatweb is a free and open-source tool available on GitHub. Whatweb is a scanner written in the Ruby language. This tool can identify and recognize all the web technologies available on the target website. This tool can identify technologies used by websites such as blogging, content management system, all JavaScript libraries. Whatweb contains more than 180 modules. each module is responsible for grabbing particular information from the target website. Whatweb works as an information-gathering tool and can identify all the email addresses, SQL errors, technology used in the website.



```
┌──(rushikrathod㉿kali)-[~]
└─$ whatweb -v -a 3 185.151.30.139
WhatWeb report for http://185.151.30.139
Status    : 200 OK
Title     : <None>
IP        : <Unknown>
Country   : <Unknown>

Summary   : UncommonHeaders[x-cdn-cache-status,x-via]

Detected Plugins:
[ UncommonHeaders ]
        Uncommon HTTP server headers. The blacklist includes all
        the standard headers and many non standard but common ones.
        Interesting but fairly common headers should have their own
        plugins, eg. x-powered-by, server and x-aspnet-version.
        Info about headers can be found at www.http-stats.com

        String       : x-cdn-cache-status,x-via (from headers)

HTTP Headers:
        HTTP/1.1 200 OK
        date: Thu, 16 Feb 2023 07:02:31 GMT
        content-type: text/plain
        content-length: 9
        cache-control: no-cache
        x-cdn-cache-status: MISS
        x-via: MAD1
        connection: close


┌──(rushikrathod㉿kali)-[~]
└─$ ▮
```

Conclusion/Summary:

Student Signature & Date	**Marks:**	**Evaluator Signature & Date**

Practical 10

Date: / /2023

Aim: Find out Web Application Vulnerability using OWASP-ZAP tool.

Solution:

What is ZAP?

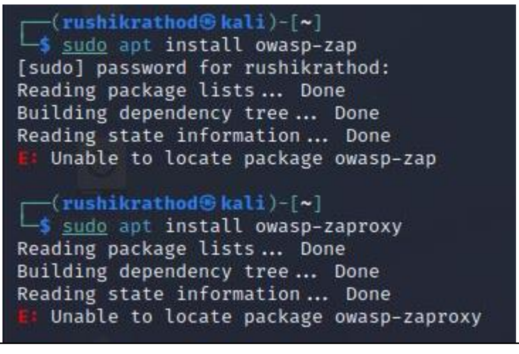
OWASP ZAP is an open-source web application security scanner. It is intended to be used by both those new to application security as well as professional penetration testers. It is one of the most active Open Web Application Security Project projects and has been given Flagship status.

- OWASP stands for "Open Web Application Security Project".
- It is an open, online community that creates methodologies, tools, technologies and guidance on how to deliver secure web applications.
- OWASP ZAP (ZAP) is one of the world's most popular free security tools and is actively maintained by hundreds of international volunteers. It can help to find security vulnerabilities in web applications. It's also a great tool for experienced pen testers and beginners.
- ZAP is what is known as a "man-in-the-middle proxy." It stands between the browser and the web application. While you navigate through all the features of the website, it captures all actions. Then it attacks the website with known techniques to find security vulnerabilities.
- It is one of the most active Open Web Application Security Project (OWASP) projects and has been given Flagship status.
- When used as a proxy server it allows the user to manipulate all of the traffic that passes through it, including traffic using https.
- It can also run in a daemon mode which is then controlled via a REST API.
- ZAP was added to the ThoughtWorks Technology Radar in May 2015 in the Trial ring.
- ZAP was originally forked from Paros, another pentesting proxy. Simon Bennetts, the project lead, stated in 2014 that only 20% of ZAP's source code was still from Paros.

Steps:

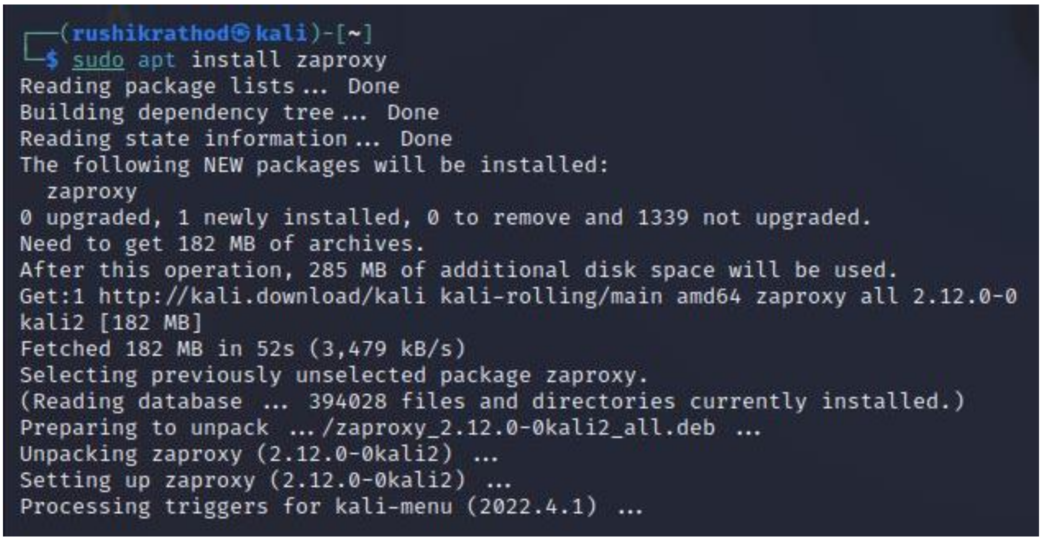
1) **Install owasp-zap by typing – sudo apt install owasp-zap**

If it shows error try – sudo apt install owasp-zaproxy or sudo apt install zaproxy
If this also shows error means we need to update kali for owasp package.
Type – sudo apt-get update



```
┌──(rushikrathod㉿kali)-[~]
└─$ sudo apt install owasp-zap
[sudo] password for rushikrathod:
Reading package lists ... Done
Building dependency tree ... Done
Reading state information ... Done
E: Unable to locate package owasp-zap

┌──(rushikrathod㉿kali)-[~]
└─$ sudo apt install owasp-zaproxy
Reading package lists ... Done
Building dependency tree ... Done
Reading state information ... Done
E: Unable to locate package owasp-zaproxy
```

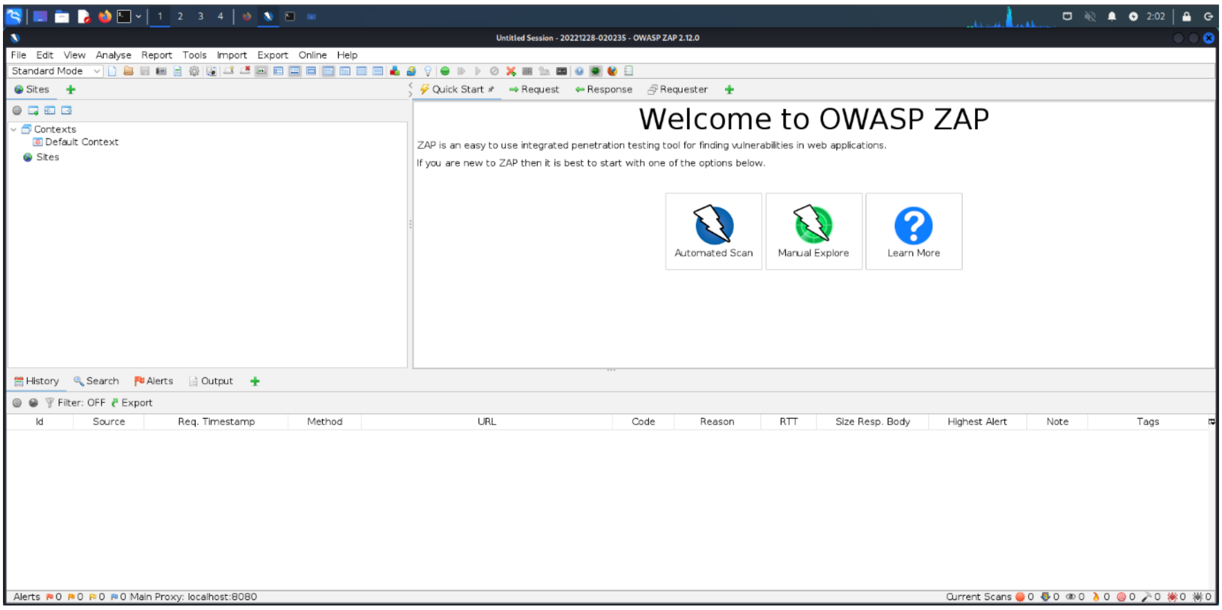
2) Install zaproxy – sudo apt install zaproxy



```
┌──(rushikrathod㉿kali)-[~]
└─$ sudo apt install zaproxy
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following NEW packages will be installed:
  zaproxy
0 upgraded, 1 newly installed, 0 to remove and 1339 not upgraded.
Need to get 182 MB of archives.
After this operation, 285 MB of additional disk space will be used.
Get:1 http://kali.download/kali kali-rolling/main amd64 zaproxy all 2.12.0-0
kali2 [182 MB]
Fetched 182 MB in 52s (3,479 kB/s)
Selecting previously unselected package zaproxy.
(Reading database ... 394028 files and directories currently installed.)
Preparing to unpack .../zaproxy_2.12.0-0kali2_all.deb ...
Unpacking zaproxy (2.12.0-0kali2) ...
Setting up zaproxy (2.12.0-0kali2) ...
Processing triggers for kali-menu (2022.4.1) ...
```

3) Go to Application menu and type zap and start that application. Now it will take some time for start the main interface.

- Spidering the web application
- Spidering a web application means crawling all the links and getting the structure of the application. ZAP provides two spiders for crawling web applications;
- The traditional ZAP spider discovers links by examining the HTML in responses from the web application. This spider is fast, but it is not always effective when exploring an AJAX web application.
- This is more likely to be effective for AJAX applications. This spider explores the web application by invoking browsers which then follow the links that have been generated. The AJAX spider is slower than the traditional spider.



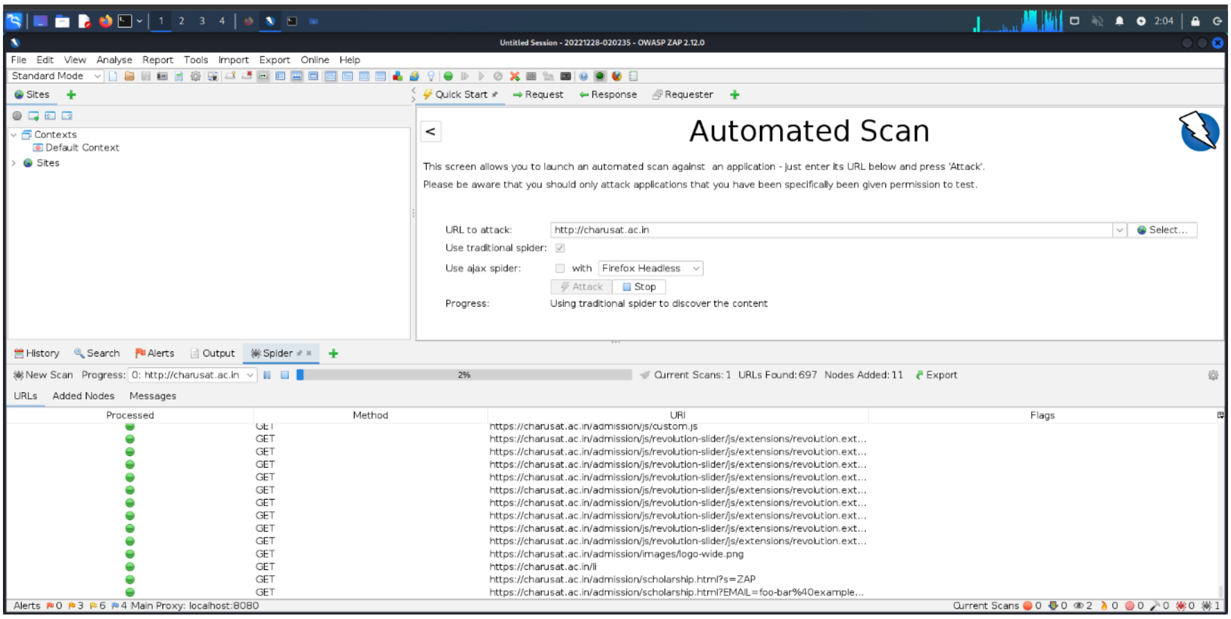
Untitled Session - 20221228-020235 - OWASP ZAP 2.12.0

Welcome to OWASP ZAP

ZAP is an easy to use integrated penetration testing tool for finding vulnerabilities in web applications.

If you are new to ZAP then it is best to start with one of the options below.

4) Click Automated Scan – Enter url in URL section to attack: http://charusat.ac.in and then click Attack.



Untitled Session - 20221228-020235 - OWASP ZAP 2.12.0

Automated Scan

This screen allows you to launch an automated scan against an application - just enter its URL below and press 'Attack'.

Please be aware that you should only attack applications that you have been specifically been given permission to test.

URL to attack: http://charusat.ac.in

Use traditional spider: ☑

Use ajax spider: □ with Firefox Headless

Attack Stop

Progress: Using traditional spider to discover the content

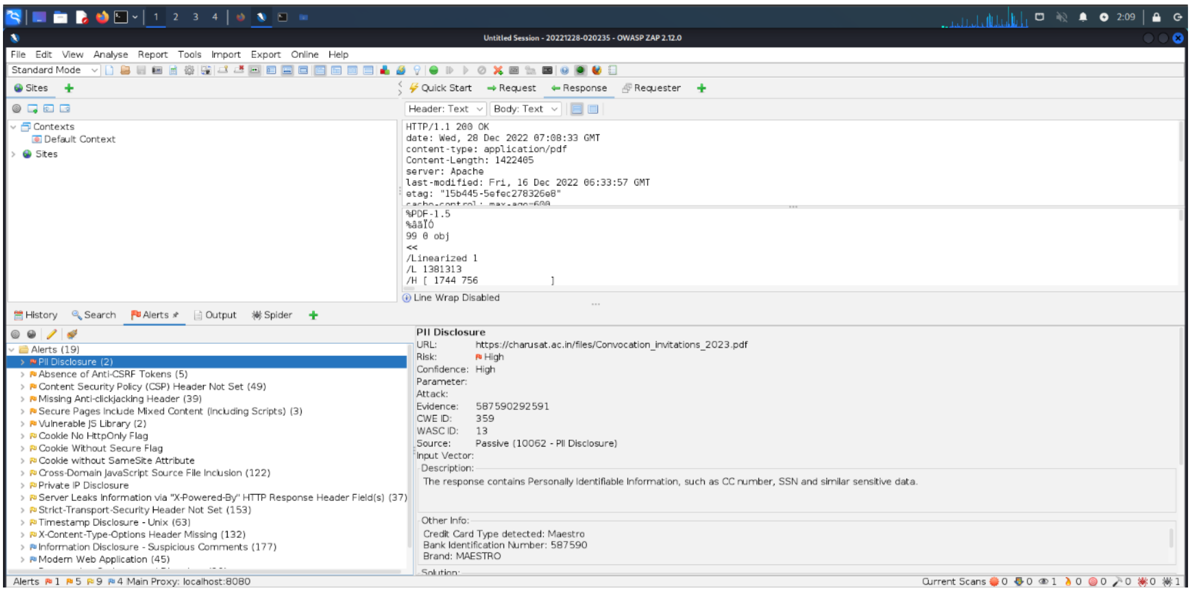
New Scan Progress: 0: http://charusat.ac.in 2% Current Scans: 1 URLs Found: 697 Nodes Added: 11 Export

URLs Added Nodes Messages

Processed	Method	URI	Flags
●	GET	https://charusat.ac.in/admission/js/custom.js	
●	GET	https://charusat.ac.in/admission/js/revolution-slider/js/extensions/revolution.ext...	
●	GET	https://charusat.ac.in/admission/js/revolution-slider/js/extensions/revolution.ext...	
●	GET	https://charusat.ac.in/admission/js/revolution-slider/js/extensions/revolution.ext...	
●	GET	https://charusat.ac.in/admission/js/revolution-slider/js/extensions/revolution.ext...	
●	GET	https://charusat.ac.in/admission/js/revolution-slider/js/extensions/revolution.ext...	
●	GET	https://charusat.ac.in/admission/js/revolution-slider/js/extensions/revolution.ext...	
●	GET	https://charusat.ac.in/admission/js/revolution-slider/js/extensions/revolution.ext...	
●	GET	https://charusat.ac.in/admission/js/revolution-slider/js/extensions/revolution.ext...	
●	GET	https://charusat.ac.in/admission/js/revolution-slider/js/extensions/revolution.ext...	
●	GET	https://charusat.ac.in/admission/images/logo-wide.png	
●	GET	https://charusat.ac.in/li	
●	GET	https://charusat.ac.in/admission/scholarship.html?s=ZAP	
●	GET	https://charusat.ac.in/admission/scholarship.html?EMAIL=foo-bar%40example...	

Alerts 0 3 6 4 Main Proxy: localhost:8080 Current Scans 0 0 2 0 0 0 0 1

5) We can check for the alerts that are present on the website by clicking on Alerts and see the risk of that particular whether it is High, Medium, or Low.



Untitled Session - 20221228-020235 - OWASP ZAP 2.12.0

```
HTTP/1.1 200 OK
date: Wed, 28 Dec 2022 07:08:33 GMT
content-type: application/pdf
Content-Length: 1422405
server: Apache
last-modified: Fri, 16 Dec 2022 06:33:57 GMT
etag: "15b445-5efec278326e8"
cache-control: max-age=600
```

```
%PDF-1.5
%âãÏÓ
99 0 obj
<<
/Linearized 1
/L 1381313
/H [ 1744 756              ]
```

Line Wrap Disabled

- Alerts (19)
  - PII Disclosure (2)
  - Absence of Anti-CSRF Tokens (5)
  - Content Security Policy (CSP) Header Not Set (49)
  - Missing Anti-clickjacking Header (39)
  - Secure Pages Include Mixed Content (Including Scripts) (3)
  - Vulnerable JS Library (2)
  - Cookie No HttpOnly Flag
  - Cookie Without Secure Flag
  - Cookie without SameSite Attribute
  - Cross-Domain JavaScript Source File Inclusion (122)
  - Private IP Disclosure
  - Server Leaks Information via "X-Powered-By" HTTP Response Header Field(s) (37)
  - Strict-Transport-Security Header Not Set (153)
  - Timestamp Disclosure - Unix (63)
  - X-Content-Type-Options Header Missing (132)
  - Information Disclosure - Suspicious Comments (177)
  - Modern Web Application (45)

PII Disclosure

URL: https://charusat.ac.in/files/Convocation_invitations_2023.pdf
Risk: High
Confidence: High
Parameter:
Attack:
Evidence: 587590292591
CWE ID: 359
WASC ID: 13
Source: Passive (10062 - PII Disclosure)
Input Vector:

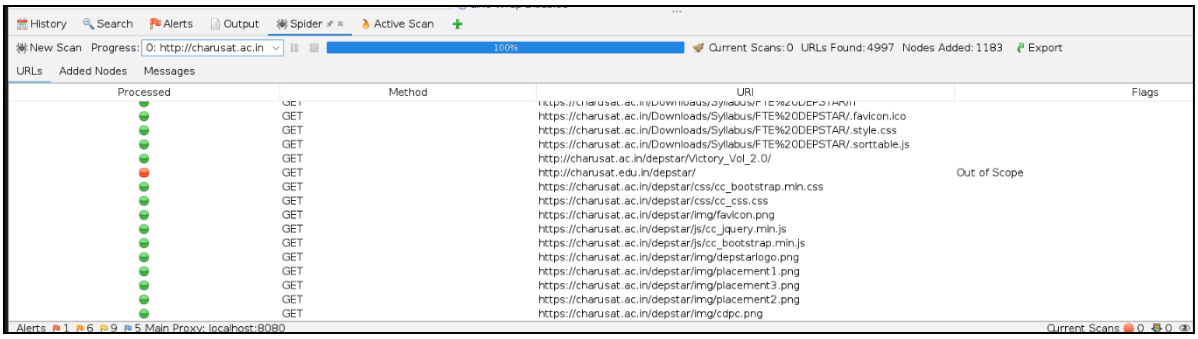
Description:
The response contains Personally Identifiable Information, such as CC number, SSN and similar sensitive data.

Other Info:
Credit Card Type detected: Maestro
Bank Identification Number: 587590
Brand: MAESTRO

Solution:

Alerts 1 5 9 4 Main Proxy: localhost:8080 Current Scans 0 0 1 0 0 0 0 1

6) **Let the scan get complete and we can get all the vulnerabilities present in the website and all the alerts so that we can work on those to make it more secure.**



Progress: 0: http://charusat.ac.in

Current Scans: 0 URLs Found: 4997 Nodes Added: 1183

Processed	Method	URI	Flags
●	GET	https://charusat.ac.in/Downloads/Syllabus/FTE%20DEPSTAR/n	
●	GET	https://charusat.ac.in/Downloads/Syllabus/FTE%20DEPSTAR/.favicon.ico	
●	GET	https://charusat.ac.in/Downloads/Syllabus/FTE%20DEPSTAR/.style.css	
●	GET	https://charusat.ac.in/Downloads/Syllabus/FTE%20DEPSTAR/.sorttable.js	
●	GET	http://charusat.ac.in/depstar/Victory_Vol_2.0/	
●	GET	http://charusat.edu.in/depstar/	Out of Scope
●	GET	https://charusat.ac.in/depstar/css/cc_bootstrap.min.css	
●	GET	https://charusat.ac.in/depstar/css/cc_css.css	
●	GET	https://charusat.ac.in/depstar/img/favicon.png	
●	GET	https://charusat.ac.in/depstar/js/cc_jquery.min.js	
●	GET	https://charusat.ac.in/depstar/js/cc_bootstrap.min.js	
●	GET	https://charusat.ac.in/depstar/img/depstarlogo.png	
●	GET	https://charusat.ac.in/depstar/img/placement1.png	
●	GET	https://charusat.ac.in/depstar/img/placement3.png	
●	GET	https://charusat.ac.in/depstar/img/placement2.png	
●	GET	https://charusat.ac.in/depstar/img/cdpc.png	

Alerts 1 6 9 5 Main Proxy: localhost:8080

Current Scans 0 0 0

Conclusion/Summary:

Student Signature & Date	**Marks:**	**Evaluator Signature & Date**

Practical 11

Date: / /2023

Aim: Implement hash function for the given Scenario.

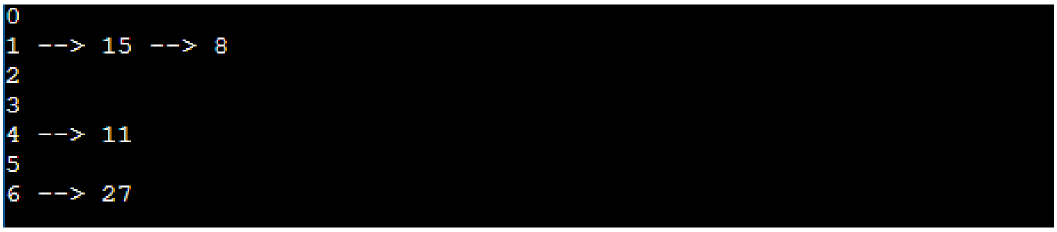
Solution:

A Hash Function is a function that converts a given numeric or alphanumeric key to a small practical integer value. The mapped integer value is used as an index in the hash table. In simple terms, a hash function maps a significant number or string to a small integer that can be used as the index in the hash table.

Code:

```
#include<bits/stdc++.h>
using namespace std;

class Hash{
	int BUCKET;
	list<int> *table;
public:
	Hash(int V);
	void insertItem(int x);
	void deleteItem(int key);
	int hashFunction(int x) {
		return (x % BUCKET);
	}

	void displayHash();
};

Hash::Hash(int b){
	this->BUCKET = b;
	table = new list<int>[BUCKET];
}

void Hash::insertItem(int key){
	int index = hashFunction(key);
	table[index].push_back(key);
}

void Hash::deleteItem(int key){
int index = hashFunction(key);
list <int> :: iterator i;
for (i = table[index].begin();
		i != table[index].end(); i++) {
	if (*i == key)
	break;
}
if (i != table[index].end())
        table[index].erase(i);
}

void Hash::displayHash() {
for (int i = 0; i< BUCKET; i++) {
        cout<<i;
        for (auto x : table[i])
        cout<< " --> " << x;
        cout<<endl;
}
}

int main(){
int a[] = {15, 11, 27, 8, 12};
int n = sizeof(a)/sizeof(a[0]);
Hash h(7);
for (int i = 0; i< n; i++)
        h.insertItem(a[i]);
h.deleteItem(12);
h.displayHash();

return 0;
}
```

Output:



```
0
1 --> 15 --> 8
2
3
4 --> 11
5
6 --> 27
```

Conclusion/Summary:

Student Signature & Date	**Marks:**	**Evaluator Signature & Date**

Practical 12

Date: / /2023

Aim: Implement DES/AES algorithm for the given plain text.

Solution:

DES:

Code:

```
#include <bits/stdc++.h>
using namespace std;
string hex2bin(string s)
{
	// hexadecimal to binary conversion
	unordered_map<char, string>mp;
	mp['0'] = "0000";
	mp['1'] = "0001";
	mp['2'] = "0010";
	mp['3'] = "0011";
	mp['4'] = "0100";
	mp['5'] = "0101";
	mp['6'] = "0110";
	mp['7'] = "0111";
	mp['8'] = "1000";
	mp['9'] = "1001";
	mp['A'] = "1010";
	mp['B'] = "1011";
	mp['C'] = "1100";
	mp['D'] = "1101";
	mp['E'] = "1110";
	mp['F'] = "1111";
	string bin = "";
	for (int i = 0; i<s.size(); i++) {
		bin += mp[s[i]];
	}
	return bin;
}
string bin2hex(string s)
{
	// binary to hexadecimal conversion
	unordered_map<string, string>mp;
	mp["0000"] = "0";
	mp["0001"] = "1";
	mp["0010"] = "2";
	mp["0011"] = "3";
	mp["0100"] = "4";
	mp["0101"] = "5";
	mp["0110"] = "6";
```

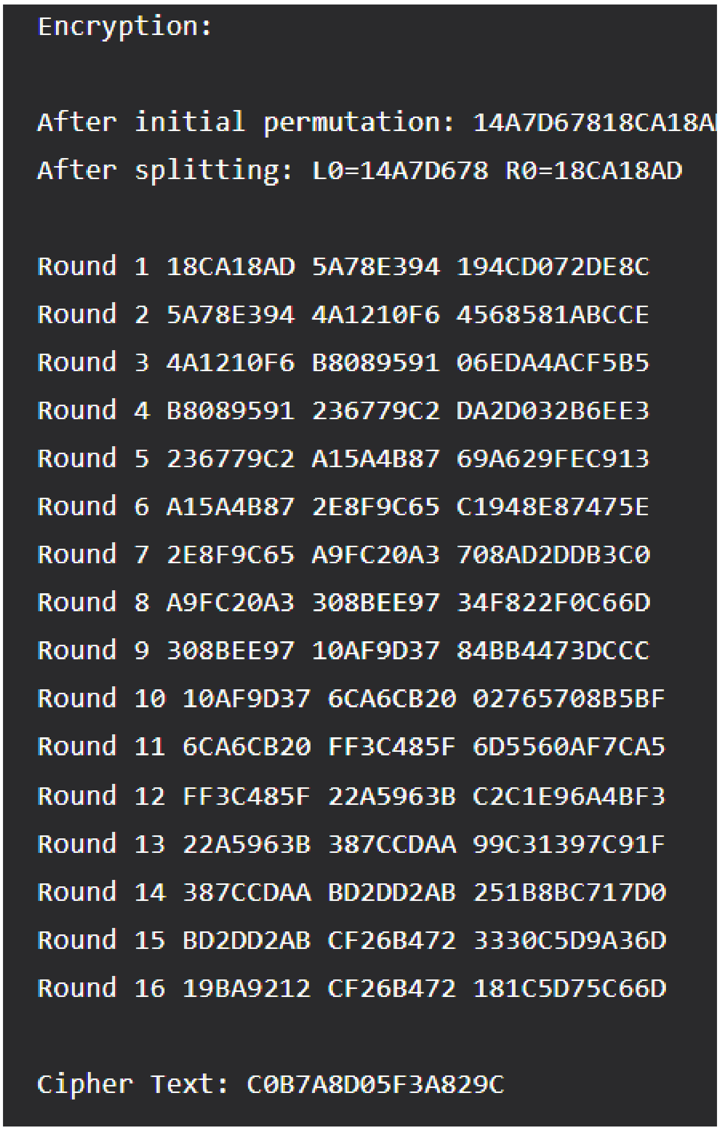
```
    mp["0111"] = "7";
    mp["1000"] = "8";
    mp["1001"] = "9";
    mp["1010"] = "A";
    mp["1011"] = "B";
    mp["1100"] = "C";
    mp["1101"] = "D";
    mp["1110"] = "E";
    mp["1111"] = "F";
    string hex = "";
    for (int i = 0; i<s.length(); i += 4) {
        string ch = "";
        ch += s[i];
        ch += s[i + 1];
        ch += s[i + 2];
        ch += s[i + 3];
        hex += mp[ch];
    }
    return hex;
}

string permute(string k, int* arr, int n)
{
    string per = "";
    for (int i = 0; i< n; i++) {
        per += k[arr[i] - 1];
    }
    return per;
}

string shift_left(string k, int shifts)
{
    string s = "";
    for (int i = 0; i< shifts; i++) {
        for (int j = 1; j < 28; j++) {
            s += k[j];
        }
        s += k[0];
        k = s;
        s = "";
    }
    return k;
}

string xor_(string a, string b)
{
    string ans = "";
    for (int i = 0; i<a.size(); i++) {
        if (a[i] == b[i]) {
            ans += "0";
```

```
                    }
                    else {
                              ans += "1";
                    }
          }
          return ans;
}

string encrypt(string pt, vector<string>rkb,
                              vector<string>rk)
{
          // Hexadecimal to binary
          pt = hex2bin(pt);

          // Initial Permutation Table
          int initial_perm[64]
                    = { 58, 50, 42, 34, 26, 18, 10, 2, 60, 52, 44,
                              36, 28, 20, 12, 4, 62, 54, 46, 38, 30, 22,
                              14, 6, 64, 56, 48, 40, 32, 24, 16, 8, 57,
                              49, 41, 33, 25, 17, 9, 1, 59, 51, 43, 35,
                              27, 19, 11, 3, 61, 53, 45, 37, 29, 21, 13,
                              5, 63, 55, 47, 39, 31, 23, 15, 7 };
          // Initial Permutation
          pt = permute(pt, initial_perm, 64);
          cout<< "After initial permutation: " << bin2hex(pt)
                    <<endl;

          // Splitting
          string left = pt.substr(0, 32);
          string right = pt.substr(32, 32);
          cout<< "After splitting: L0=" << bin2hex(left)
                    << " R0=" << bin2hex(right) <<endl;

          // Expansion D-box Table
          int exp_d[48]
                    = { 32, 1, 2, 3, 4, 5, 4, 5, 6, 7, 8, 9,
                              8, 9, 10, 11, 12, 13, 12, 13, 14, 15, 16, 17,
                              16, 17, 18, 19, 20, 21, 20, 21, 22, 23, 24, 25,
                              24, 25, 26, 27, 28, 29, 28, 29, 30, 31, 32, 1 };

          // S-box Table
          int s[8][4][16] = {
                    { 14, 4, 13, 1, 2, 15, 11, 8, 3, 10, 6, 12, 5,9, 0, 7, 0, 15, 7, 4, 14, 2, 13, 1, 10, 6,
                    12, 11, 9, 5, 3, 8, 4, 1, 14, 8, 13, 6, 2,11, 15, 12, 9, 7, 3, 10, 5, 0, 15, 12, 8, 2,
                    4, 9, 1, 7, 5, 11, 3, 14, 10, 0, 6, 13 },
                    { 15, 1, 8, 14, 6, 11, 3, 4, 9, 7, 2, 13, 12,0, 5, 10, 3, 13, 4, 7, 15, 2, 8, 14, 12, 0,
                    1, 10, 6, 9, 11, 5, 0, 14, 7, 11, 10, 4, 13,1, 5, 8, 12, 6, 9, 3, 2, 15, 13, 8, 10, 1,
                    3, 15, 4, 2, 11, 6, 7, 12, 0, 5, 14, 9 },

                    { 10, 0, 9, 14, 6, 3, 15, 5, 1, 13, 12,7, 11, 4, 2, 8, 13, 7, 0, 9, 3, 4,
        6, 10, 2, 8, 5, 14, 12, 11, 15, 1, 13,6, 4, 9, 8, 15, 3, 0, 11, 1, 2, 12,
        5, 10, 14, 7, 1, 10, 13, 0, 6, 9, 8,7, 4, 15, 14, 3, 11, 5, 2, 12 },
        { 7, 13, 14, 3, 0, 6, 9, 10, 1, 2, 8, 5, 11,12, 4, 15, 13, 8, 11, 5, 6, 15, 0, 3, 4, 7,
        2, 12, 1, 10, 14, 9, 10, 6, 9, 0, 12, 11, 7,13, 15, 1, 3, 14, 5, 2, 8, 4, 3, 15, 0, 6,
        10, 1, 13, 8, 9, 4, 5, 11, 12, 7, 2, 14 },
        { 2, 12, 4, 1, 7, 10, 11, 6, 8, 5, 3, 15, 13,0, 14, 9, 14, 11, 2, 12, 4, 7, 13, 1, 5, 0,
        15, 10, 3, 9, 8, 6, 4, 2, 1, 11, 10, 13, 7,8, 15, 9, 12, 5, 6, 3, 0, 14, 11, 8, 12, 7,
        1, 14, 2, 13, 6, 15, 0, 9, 10, 4, 5, 3 },
        { 12, 1, 10, 15, 9, 2, 6, 8, 0, 13, 3, 4, 14,7, 5, 11, 10, 15, 4, 2, 7, 12, 9, 5, 6, 1,
        13, 14, 0, 11, 3, 8, 9, 14, 15, 5, 2, 8, 12,3, 7, 0, 4, 10, 1, 13, 11, 6, 4, 3, 2, 12,
        9, 5, 15, 10, 11, 14, 1, 7, 6, 0, 8, 13 },
        { 4, 11, 2, 14, 15, 0, 8, 13, 3, 12, 9, 7, 5,10, 6, 1, 13, 0, 11, 7, 4, 9, 1, 10, 14, 3,
        5, 12, 2, 15, 8, 6, 1, 4, 11, 13, 12, 3, 7,14, 10, 15, 6, 8, 0, 5, 9, 2, 6, 11, 13, 8,
        1, 4, 10, 7, 9, 5, 0, 15, 14, 2, 3, 12 },
        { 13, 2, 8, 4, 6, 15, 11, 1, 10, 9, 3, 14, 5,0, 12, 7, 1, 15, 13, 8, 10, 3, 7, 4, 12, 5,
        6, 11, 0, 14, 9, 2, 7, 11, 4, 1, 9, 12, 14,2, 0, 6, 10, 13, 15, 3, 5, 8, 2, 1, 14, 7,
        4, 10, 8, 13, 15, 12, 9, 0, 3, 5, 6, 11 }
    };
    int per[32]
        = { 16, 7, 20, 21, 29, 12, 28, 17, 1, 15, 23,
            26, 5, 18, 31, 10, 2, 8, 24, 14, 32, 27,
            3, 9, 19, 13, 30, 6, 22, 11, 4, 25 };

    cout<<endl;
    for (int i = 0; i< 16; i++) {
        string right_expanded = permute(right, exp_d, 48);
        string x = xor_(rkb[i], right_expanded);
        string op = "";
        for (int i = 0; i< 8; i++) {
            int row = 2 * int(x[i * 6] - '0')
                    + int(x[i * 6 + 5] - '0');
            int col = 8 * int(x[i * 6 + 1] - '0')
                    + 4 * int(x[i * 6 + 2] - '0')
                    + 2 * int(x[i * 6 + 3] - '0')
                    + int(x[i * 6 + 4] - '0');
            int val = s[i][row][col];
            op += char(val / 8 + '0');
            val = val % 8;
            op += char(val / 4 + '0');
            val = val % 4;
            op += char(val / 2 + '0');
            val = val % 2;
            op += char(val + '0');
        }
        op = permute(op, per, 32);
        x = xor_(op, left);
        left = x;
        if (i != 15) {
            swap(left, right);
        }
```

```
                cout<< "Round " <<i + 1 << " " << bin2hex(left)
                        << " " << bin2hex(right) << " " <<rk[i]
                        <<endl;
        }

        string combine = left + right;

        int final_perm[64]
                = { 40, 8, 48, 16, 56, 24, 64, 32, 39, 7, 47,15, 55, 23, 63, 31, 38, 6, 46, 14, 54, 22,
                        62, 30, 37, 5, 45, 13, 53, 21, 61, 29, 36,4, 44, 12, 52, 20, 60, 28, 35, 3, 43, 11,
                        51, 19, 59, 27, 34, 2, 42, 10, 50, 18, 58,26, 33, 1, 41, 9, 49, 17, 57, 25 };

        string cipher
                = bin2hex(permute(combine, final_perm, 64));
        return cipher;
}

int main()
{
        string pt, key;
        /*cout<<"Enter plain text(in hexadecimal): ";
        cin>>pt;
        cout<<"Enter key(in hexadecimal): ";
        cin>>key;*/

        pt = "123456ABCD132536";
        key = "AABB09182736CCDD";

        key = hex2bin(key);

        int keyp[56]
                = { 57, 49, 41, 33, 25, 17, 9, 1, 58, 50, 42, 34,26, 18, 10, 2, 59, 51, 43, 35, 27, 19, 11,
3,60, 52, 44, 36, 63, 55, 47, 39, 31, 23, 15, 7,62, 54, 46, 38, 30, 22, 14, 6, 61, 53, 45, 37,29, 21, 13,
5, 28, 20, 12, 4 };

        key = permute(key, keyp, 56);

        int shift_table[16] = { 1, 1, 2, 2, 2, 2, 2, 2,
                                                        1, 2, 2, 2, 2, 2, 2, 1 };

        int key_comp[48] = { 14, 17, 11, 24, 1, 5, 3, 28,15, 6, 21, 10, 23, 19, 12, 4,26, 8, 16, 7, 27,
20, 13, 2,41, 52, 31, 37, 47, 55, 30, 40,51, 45, 33, 48, 44, 49, 39, 56,34, 53, 46, 42, 50, 36, 29, 32 };

        string left = key.substr(0, 28);
        string right = key.substr(28, 28);
        vector<string>rkb; // rkb for RoundKeys in binary
        vector<string>rk; // rk for RoundKeys in hexadecimal
        for (int i = 0; i< 16; i++) {
                // Shifting
                left = shift_left(left, shift_table[i]);
```

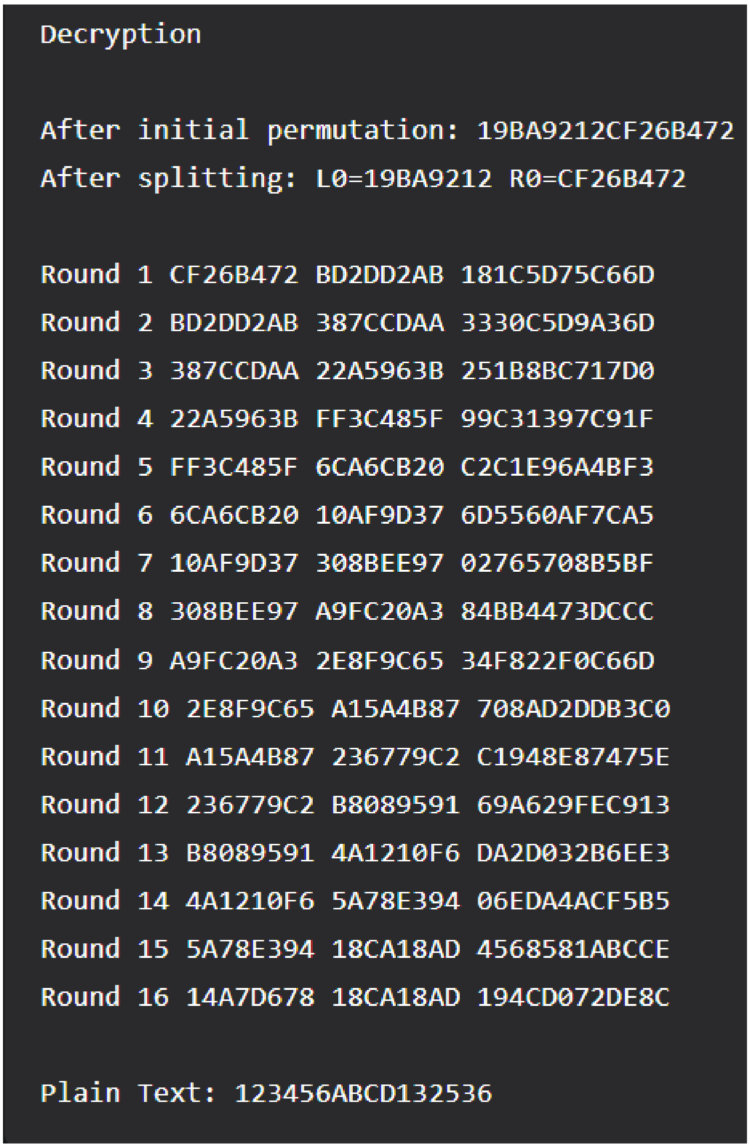
```
                    right = shift_left(right, shift_table[i]);
                    string combine = left + right;
                    string RoundKey = permute(combine, key_comp, 48);
                    rkb.push_back(RoundKey);
                    rk.push_back(bin2hex(RoundKey));
          }
          cout<< "\nEncryption:\n\n";
          string cipher = encrypt(pt, rkb, rk);
          cout<< "\nCipher Text: " << cipher <<endl;
          cout<< "\nDecryption\n\n";
          reverse(rkb.begin(), rkb.end());
          reverse(rk.begin(), rk.end());
          string text = encrypt(cipher, rkb, rk);
          cout<< "\nPlain Text: " << text <<endl;
}
```

Encryption:



```
Encryption:


After initial permutation: 14A7D67818CA18A
After splitting: L0=14A7D678 R0=18CA18AD


Round 1 18CA18AD 5A78E394 194CD072DE8C
Round 2 5A78E394 4A1210F6 4568581ABCCE
Round 3 4A1210F6 B8089591 06EDA4ACF5B5
Round 4 B8089591 236779C2 DA2D032B6EE3
Round 5 236779C2 A15A4B87 69A629FEC913
Round 6 A15A4B87 2E8F9C65 C1948E87475E
Round 7 2E8F9C65 A9FC20A3 708AD2DDB3C0
Round 8 A9FC20A3 308BEE97 34F822F0C66D
Round 9 308BEE97 10AF9D37 84BB4473DCCC
Round 10 10AF9D37 6CA6CB20 02765708B5BF
Round 11 6CA6CB20 FF3C485F 6D5560AF7CA5
Round 12 FF3C485F 22A5963B C2C1E96A4BF3
Round 13 22A5963B 387CCDAA 99C31397C91F
Round 14 387CCDAA BD2DD2AB 251B8BC717D0
Round 15 BD2DD2AB CF26B472 3330C5D9A36D
Round 16 19BA9212 CF26B472 181C5D75C66D


Cipher Text: C0B7A8D05F3A829C
```

Decryption:



```
Decryption

After initial permutation: 19BA9212CF26B472
After splitting: L0=19BA9212 R0=CF26B472

Round 1 CF26B472 BD2DD2AB 181C5D75C66D
Round 2 BD2DD2AB 387CCDAA 3330C5D9A36D
Round 3 387CCDAA 22A5963B 251B8BC717D0
Round 4 22A5963B FF3C485F 99C31397C91F
Round 5 FF3C485F 6CA6CB20 C2C1E96A4BF3
Round 6 6CA6CB20 10AF9D37 6D5560AF7CA5
Round 7 10AF9D37 308BEE97 02765708B5BF
Round 8 308BEE97 A9FC20A3 84BB4473DCCC
Round 9 A9FC20A3 2E8F9C65 34F822F0C66D
Round 10 2E8F9C65 A15A4B87 708AD2DDB3C0
Round 11 A15A4B87 236779C2 C1948E87475E
Round 12 236779C2 B8089591 69A629FEC913
Round 13 B8089591 4A1210F6 DA2D032B6EE3
Round 14 4A1210F6 5A78E394 06EDA4ACF5B5
Round 15 5A78E394 18CA18AD 4568581ABCCE
Round 16 14A7D678 18CA18AD 194CD072DE8C

Plain Text: 123456ABCD132536
```

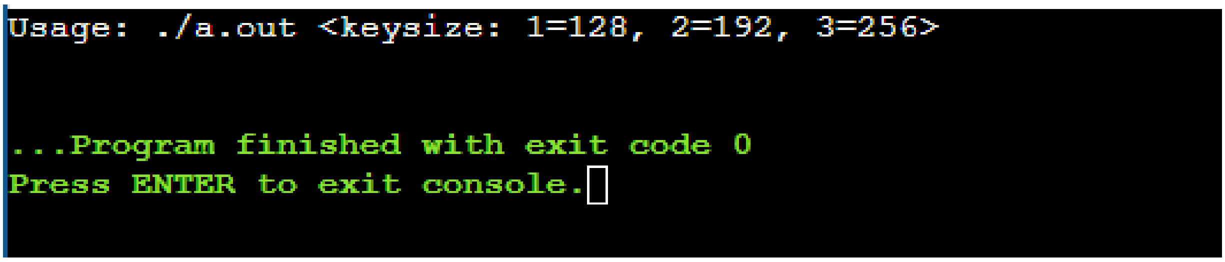
AES:

Code:

```
#include <stdio.h>
#include <iostream>
#include <stdlib.h>
#include <string.h>
using namespace std;
#define Nb 4

int Nr=0;
int Nk=0;.
unsigned char in[1024], out[1024], state[4][Nb];
unsigned char RoundKey[240];
unsigned char Key[32];

int getSBoxValue(int num) {
  int sbox[256] = {
    0x63, 0x7c, 0x77, 0x7b, 0xf2, 0x6b, 0x6f, 0xc5,
    0x30, 0x01, 0x67, 0x2b, 0xfe, 0xd7, 0xab, 0x76, //0
    0xca, 0x82, 0xc9, 0x7d, 0xfa, 0x59, 0x47, 0xf0,
    0xad, 0xd4, 0xa2, 0xaf, 0x9c, 0xa4, 0x72, 0xc0, //1
    0xb7, 0xfd, 0x93, 0x26, 0x36, 0x3f, 0xf7, 0xcc,
    0x34, 0xa5, 0xe5, 0xf1, 0x71, 0xd8, 0x31, 0x15, //2
    0x04, 0xc7, 0x23, 0xc3, 0x18, 0x96, 0x05, 0x9a,
    0x07, 0x12, 0x80, 0xe2, 0xeb, 0x27, 0xb2, 0x75, //3
    0x09, 0x83, 0x2c, 0x1a, 0x1b, 0x6e, 0x5a, 0xa0,
    0x52, 0x3b, 0xd6, 0xb3, 0x29, 0xe3, 0x2f, 0x84, //4
    0x53, 0xd1, 0x00, 0xed, 0x20, 0xfc, 0xb1, 0x5b,
    0x6a, 0xcb, 0xbe, 0x39, 0x4a, 0x4c, 0x58, 0xcf, //5
    0xd0, 0xef, 0xaa, 0xfb, 0x43, 0x4d, 0x33, 0x85,
    0x45, 0xf9, 0x02, 0x7f, 0x50, 0x3c, 0x9f, 0xa8, //6
    0x51, 0xa3, 0x40, 0x8f, 0x92, 0x9d, 0x38, 0xf5,
    0xbc, 0xb6, 0xda, 0x21, 0x10, 0xff, 0xf3, 0xd2, //7
    0xcd, 0x0c, 0x13, 0xec, 0x5f, 0x97, 0x44, 0x17,
    0xc4, 0xa7, 0x7e, 0x3d, 0x64, 0x5d, 0x19, 0x73, //8
    0x60, 0x81, 0x4f, 0xdc, 0x22, 0x2a, 0x90, 0x88,
    0x46, 0xee, 0xb8, 0x14, 0xde, 0x5e, 0x0b, 0xdb, //9
    0xe0, 0x32, 0x3a, 0x0a, 0x49, 0x06, 0x24, 0x5c,
    0xc2, 0xd3, 0xac, 0x62, 0x91, 0x95, 0xe4, 0x79, //A
    0xe7, 0xc8, 0x37, 0x6d, 0x8d, 0xd5, 0x4e, 0xa9,
    0x6c, 0x56, 0xf4, 0xea, 0x65, 0x7a, 0xae, 0x08, //B
    0xba, 0x78, 0x25, 0x2e, 0x1c, 0xa6, 0xb4, 0xc6,
    0xe8, 0xdd, 0x74, 0x1f, 0x4b, 0xbd, 0x8b, 0x8a, //C
    0x70, 0x3e, 0xb5, 0x66, 0x48, 0x03, 0xf6, 0x0e,
    0x61, 0x35, 0x57, 0xb9, 0x86, 0xc1, 0x1d, 0x9e, //D
    0xe1, 0xf8, 0x98, 0x11, 0x69, 0xd9, 0x8e, 0x94,
    0x9b, 0x1e, 0x87, 0xe9, 0xce, 0x55, 0x28, 0xdf, //E
    0x8c, 0xa1, 0x89, 0x0d, 0xbf, 0xe6, 0x42, 0x68,
    0x41, 0x99, 0x2d, 0x0f, 0xb0, 0x54, 0xbb, 0x16 }; //F
  return sbox[num];}

int Rcon[255] = {
      0x8d, 0x01, 0x02, 0x04, 0x08, 0x10, 0x20,
  0x40, 0x80, 0x1b, 0x36, 0x6c, 0xd8, 0xab, 0x4d,
  0x9a, 0x2f, 0x5e, 0xbc, 0x63, 0xc6, 0x97, 0x35,
  0x6a, 0xd4, 0xb3, 0x7d, 0xfa, 0xef, 0xc5, 0x91,
  0x39, 0x72, 0xe4, 0xd3, 0xbd, 0x61, 0xc2, 0x9f,
  0x25, 0x4a, 0x94, 0x33, 0x66, 0xcc, 0x83, 0x1d,
  0x3a, 0x74, 0xe8, 0xcb, 0x8d, 0x01, 0x02, 0x04,
  0x08, 0x10, 0x20, 0x40, 0x80, 0x1b, 0x36, 0x6c,
  0xd8, 0xab, 0x4d, 0x9a, 0x2f, 0x5e, 0xbc, 0x63,
  0xc6, 0x97, 0x35, 0x6a, 0xd4, 0xb3, 0x7d, 0xfa,
  0xef, 0xc5, 0x91, 0x39, 0x72, 0xe4, 0xd3, 0xbd,
  0x61, 0xc2, 0x9f, 0x25, 0x4a, 0x94, 0x33, 0x66,
  0xcc, 0x83, 0x1d, 0x3a, 0x74, 0xe8, 0xcb, 0x8d,
  0x01, 0x02, 0x04, 0x08, 0x10, 0x20, 0x40, 0x80,
  0x1b, 0x36, 0x6c, 0xd8, 0xab, 0x4d, 0x9a, 0x2f,
  0x5e, 0xbc, 0x63, 0xc6, 0x97, 0x35, 0x6a, 0xd4,
  0xb3, 0x7d, 0xfa, 0xef, 0xc5, 0x91, 0x39, 0x72,
  0xe4, 0xd3, 0xbd, 0x61, 0xc2, 0x9f, 0x25, 0x4a,
  0x94, 0x33, 0x66, 0xcc, 0x83, 0x1d, 0x3a, 0x74,
  0xe8, 0xcb, 0x8d, 0x01, 0x02, 0x04, 0x08, 0x10,
  0x20, 0x40, 0x80, 0x1b, 0x36, 0x6c, 0xd8, 0xab,
  0x4d, 0x9a, 0x2f, 0x5e, 0xbc, 0x63, 0xc6, 0x97,
  0x35, 0x6a, 0xd4, 0xb3, 0x7d, 0xfa, 0xef, 0xc5,
  0x91, 0x39, 0x72, 0xe4, 0xd3, 0xbd, 0x61, 0xc2,
  0x9f, 0x25, 0x4a, 0x94, 0x33, 0x66, 0xcc, 0x83,
  0x1d, 0x3a, 0x74, 0xe8, 0xcb, 0x8d, 0x01, 0x02,
  0x04, 0x08, 0x10, 0x20, 0x40, 0x80, 0x1b, 0x36,
  0x6c, 0xd8, 0xab, 0x4d, 0x9a, 0x2f, 0x5e, 0xbc,
  0x63, 0xc6, 0x97, 0x35, 0x6a, 0xd4, 0xb3, 0x7d,
  0xfa, 0xef, 0xc5, 0x91, 0x39, 0x72, 0xe4, 0xd3,
  0xbd, 0x61, 0xc2, 0x9f, 0x25, 0x4a, 0x94, 0x33,
  0x66, 0xcc, 0x83, 0x1d, 0x3a, 0x74, 0xe8, 0xcb  };

void KeyExpansion() {
  int i,j;
  unsigned char temp[4],k;
  for (i=0 ; i<Nk ; i++) {
RoundKey[i*4] = Key[i*4];
RoundKey[i*4+1] = Key[i*4+1];
RoundKey[i*4+2] = Key[i*4+2];
RoundKey[i*4+3] = Key[i*4+3];
  }
  while (i< (Nb * (Nr+1))) {
    for (j=0 ; j < 4 ; j++) {
         temp[j] = RoundKey[(i-1) * 4 + j];
    }
    if (i % Nk == 0) {
         k = temp[0];
         temp[0] = temp[1];
         temp[1] = temp[2];
         temp[2] = temp[3];
         temp[3] = k;
         temp[0] = getSBoxValue(temp[0]);
         temp[1] = getSBoxValue(temp[1]);
         temp[2] = getSBoxValue(temp[2]);
         temp[3] = getSBoxValue(temp[3]);
         temp[0] =  temp[0] ^ Rcon[i/Nk];
    }
    else if (Nk> 6 &&i % Nk == 4) {
         temp[0] = getSBoxValue(temp[0]);
         temp[1] = getSBoxValue(temp[1]);
```

```
           temp[2] = getSBoxValue(temp[2]);
           temp[3] = getSBoxValue(temp[3]);
      }
RoundKey[i*4+0] = RoundKey[(i-Nk)*4+0] ^ temp[0];
RoundKey[i*4+1] = RoundKey[(i-Nk)*4+1] ^ temp[1];
RoundKey[i*4+2] = RoundKey[(i-Nk)*4+2] ^ temp[2];
RoundKey[i*4+3] = RoundKey[(i-Nk)*4+3] ^ temp[3];
i++;
  }}

void AddRoundKey(int round) {
  int i,j;
  for (i=0 ; i< Nb ; i++) {
    for (j=0 ; j < 4 ; j++) {
         state[j][i] ^= RoundKey[round * Nb * 4 + i * Nb + j];
    }
  }
}

void SubBytes() {
  int i,j;
  for (i=0 ; i< 4 ; i++) {
    for (j=0 ; j < Nb ; j++) {
         state[i][j] = getSBoxValue(state[i][j]);
    }
  }
}

void ShiftRows() {
  unsigned char temp;
  temp = state[1][0];
  state[1][0] = state[1][1];
  state[1][1] = state[1][2];
  state[1][2] = state[1][3];
  state[1][3] = temp;
  temp = state[2][0];
  state[2][0] = state[2][2];
  state[2][2] = temp;
  temp = state[2][1];
  state[2][1] = state[2][3];
  state[2][3] = temp;
  temp = state[3][0];
  state[3][0] = state[3][3];
  state[3][3] = state[3][2];
  state[3][2] = state[3][1];
  state[3][1] = temp;
}

void MixColumns() {
  int i;
  unsigned char Tmp,Tm,t;
  for (i=0 ; i< Nb ; i++) {
    t = state[0][i];
Tmp = state[0][i] ^ state[1][i] ^ state[2][i] ^ state[3][i] ;
    Tm = state[0][i] ^ state[1][i] ;
    Tm = xtime(Tm);
    state[0][i] ^= Tm ^ Tmp ;

    Tm = state[1][i] ^ state[2][i] ;
    Tm = xtime(Tm);
    state[1][i] ^= Tm ^ Tmp ;

    Tm = state[2][i] ^ state[3][i] ;
    Tm = xtime(Tm);
    state[2][i] ^= Tm ^ Tmp ;

    Tm = state[3][i] ^ t ;
    Tm = xtime(Tm);
    state[3][i] ^= Tm ^ Tmp ;
  }
}

void Cipher() {
  int i,j,round=0;
  for (i=0 ; i< Nb ; i++) {
    for (j=0 ; j < 4 ; j++) {
         state[j][i] = in[i*4 + j]; }
  }
AddRoundKey(0);
  for (round=1 ; round < Nr ; round++) {
SubBytes();
ShiftRows();
MixColumns();
AddRoundKey(round);
  }
SubBytes();
ShiftRows();
AddRoundKey(Nr);
  for (i=0 ; i< Nb ; i++) {
    for (j=0 ; j < 4 ; j++) {
         out[i*4+j]=state[j][i];
    }}}
int fillBlock (int sz, char *str, unsigned char *in) {
  int j=0;
  while (sz<strlen(str)) {
    if (j >= Nb*4) break;
    in[j++] = (unsigned char)str[sz];
sz++;}
  // Pad the block with 0s, if necessary
  if (sz>= strlen(str)) for ( ; j < Nb*4 ; j++) in[j] = 0;
  return sz;
}
int main(int argc, char **argv) {
  int i;

  if (argc != 2) {
cerr<< "Usage: " <<argv[0] << " <keysize: 1=128, 2=192, 3=256>\n";
    exit(0); }
  switch (atoi(argv[1])) {
  case 1: Nk = 4; break;
  case 2: Nk = 6; break;
  case 3: Nk = 8; break;
  default: Nk = 4; break; }
  Nr = Nk + 6;
  Key[0]  = 0x2b;  Key[1]  = 0x7e;  Key[2]  = 0x15;  Key[3]  = 0x16;
  Key[4]  = 0x28;  Key[5]  = 0xae;  Key[6]  = 0xd2;  Key[7]  = 0xa6;
  Key[8]  = 0xab;  Key[9]  = 0xf7;  Key[10] = 0x15;  Key[11] = 0x88;
  Key[12] = 0x09;  Key[13] = 0xcf;  Key[14] = 0x4f;  Key[15] = 0x3c;
  char str[1024];
fgets(str, 1024, stdin);
KeyExpansion();
  int sz=0;
  while (sz<strlen(str)) {
sz = fillBlock (sz, str, in);
    Cipher();
    for (i=0 ; i< Nb*4 ; i++) cout<< (int)out[i] << " ";
  }
printf("\n\n");
}
```

Output:



```
Usage: ./a.out <keysize: 1=128, 2=192, 3=256>


...Program finished with exit code 0
Press ENTER to exit console.
```

Conclusion/Summary:

Student Signature & Date	**Marks:**	**Evaluator Signature & Date**