KARPAGAM ACADEMY OF HIGHER EDUCATION
(Deemed University Established Under Section 3 of UGC Act 1956)
Coimbatore - 641021.
(For the candidates admitted from 2017 onwards)
DEPARTMENT OF COMPUTER SCIECE

Semester – V

17CSU511A INFORMATION SECURITY - PRACTICAL 4H – 2C

Instruction Hours / week: L: 0 T: 0 P: 4 Marks: Int : **40** Ext : **60** Total: **100**

1. Demonstrate the use of Network tools: ping, ipconfig, ifconfig, tracert, arp, netstat, whois
2. Use of Password cracking tools : John the Ripper, Ophcrack. Verify the trength of passwords using these tools.
3. Perform encryption and decryption of Caesar cipher. Write a script for performing these operations.
4. Perform encryption and decryption of a Rail fence cipher. Write a script for performing these operations.
5. Use nmap/zenmap to analyse a remote machine.
6. Use Burp proxy to capture and modify the message.
7. Demonstrate sending of a protected word document.
8. Demonstrate sending of a digitally signed document.
9. Demonstrate sending of a protected worksheet.
10. Demonstrate use of steganography tools.
11. Demonstrate use of gpg utility for signing and encrypting purposes.

KARPAGAM ACADEMY OF HIGHER EDUCATION

(Deemed to be University)

(Established Under Section 3 of UGC Act 1956)

Eachanari Post, Coimbatore – 641021, INDIA

DEPARTMENT OF CS, CA & IT

III -B.Sc. (COMPUTER SCIENCE)

INFORMATION SECURITY PRACTICAL

(17CSU511A)

SEMESTER: V

(2017-2020 Batch)

Name : _____

Reg.No : _____

KARPAGAM ACADEMY OF HIGHER EDUCATION

(Deemed to be University)

(Established Under Section 3 of UGC Act 1956)

Eachanari Post, Coimbatore – 641021, INDIA

DEPARTMENT OF CS, CA & IT

CERTIFICATE

This is to certify that this is a bonafide record work done by _____of **III-BSc. Computer Science** during the period **June to November-2019** for the **"Information Security Practical".**

Examination held on _____

Reg.No:	**Subject Code: 17CSU511A**

_____ _____

Staff-in-charge **Head of the Department**

_____ _____

(Internal Examiner) **(External Examiner)**

CONTENTS

Ex.No:01	Network Tools
Date:	

Aim:

Demonstrate the use of network tools:ping, ipconfig,ipconfig/all, truncate,arp, netstat.

Algorithm:

Step 1:Open cmd (command prompt).

Step 2:PING IPaddress used to sends packets of data to a specific IP address on a network, and then lets us to know how long it took to transmit that data and get a response.

Step 3: TRACERTuse to trace the path that an Internet Protocol (IP) packet takes to its destination from a source.

Step 4:IPCONFIG tool used to find out the configured network adapters information, such as IP address, subnet mask and gateway.

Step 5: IPCONFIG /ALL - shows detailed information of network adapter that includes IP address, subnet mask, gateway, DNS, DHCP, MAC address, etc.

Step 6: ARP maps the physical MAC address with the IP

Step 7: NETSTAT command isused to show detailed network status information.

Program:

C:\Users\User.TERMINAL90>ping 172.16.25.1

Pinging 172.16.25.1 with 32 bytes of data:
Reply from 172.16.25.1: bytes=32 time=3ms TTL=127
Reply from 172.16.25.1: bytes=32 time=1ms TTL=127
Reply from 172.16.25.1: bytes=32 time=1ms TTL=127
Reply from 172.16.25.1: bytes=32 time=1ms TTL=127
Ping statistics for 172.16.25.1:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 1ms, Maximum = 3ms, Average = 1ms

C:\Users\User.TERMINAL90>ping 172.16.4.90

Pinging 172.16.4.90 with 32 bytes of data:
Reply from 172.16.4.90: bytes=32 time=5ms TTL=128
Reply from 172.16.4.90: bytes=32 time=1ms TTL=128
Reply from 172.16.4.90: bytes=32 time=1ms TTL=128
Reply from 172.16.4.90: bytes=32 time=1ms TTL=128
Ping statistics for 172.16.4.90:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 1ms, Maximum = 5ms, Average = 2ms

C:\Users\User.TERMINAL90>ping 172.16.4.100

Pinging 172.16.4.100 with 32 bytes of data:
Reply from 172.16.4.100: bytes=32 time=3ms TTL=128
Reply from 172.16.4.100: bytes=32 time=1ms TTL=128
Reply from 172.16.4.100: bytes=32 time=1ms TTL=128
Reply from 172.16.4.100: bytes=32 time=1ms TTL=128
Ping statistics for 172.16.4.100:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:

Minimum = 1ms, Maximum = 3ms, Average = 1ms

C:\Users\User.TERMINAL90>ipconfig

Windows IP Configuration

Ethernet adapter Local Area Connection:

Connection-specific DNS Suffix . :

Link-local IPv6 Address . . . . . : fe80::a55c:379f:a2dc:2b58%11

IPv4 Address. . . . . : 172.16.4.81

Subnet Mask . . . . . : 255.255.254.0

Default Gateway . . . . . : 172.16.4.254

Tunnel adapter isatap.{B49FA87C-984C-42A3-A715-EA0B95DA60C4}:

Media State . . . . . : Media disconnected

Connection-specific DNS Suffix . :

C:\Users\User.TERMINAL90>ipconfig/all

Windows IP Configuration

Host Name . . . . . : Terminal81

Primary Dns Suffix . . . . . :

Node Type . . . . . : Hybrid

IP Routing Enabled. . . . . : No

WINS Proxy Enabled. . . . . : No

Ethernet adapter Local Area Connection:

Connection-specific DNS Suffix . :

Description . . . . . : RealtekPCIe GBE Family Controller

Physical Address. . . . . : 60-45-CB-87-3E-F8

DHCP Enabled. . . . . : No

Autoconfiguration Enabled . . . . : Yes

Link-local IPv6 Address . . . . . : fe80::a55c:379f:a2dc:2b58%11(Preferred)

IPv4 Address. . . . . : 172.16.4.81(Preferred)

Subnet Mask . . . . . : 255.255.254.0

Default Gateway . . . . . : 172.16.4.254

DHCPv6 IAID . . . . . : 241190347

DHCPv6 Client DUID. . . . . : 00-01-00-01-21-6A-A4-B5-60-45-CB-87-3E-F8

DNS Servers . . . . . : 172.16.25.8

8.8.8.8

NetBIOS over Tcpip. . . . . : Enabled

Tunnel adapter isatap.{B49FA87C-984C-42A3-A715-EA0B95DA60C4}:

Media State . . . . . : Media disconnected

Connection-specific DNS Suffix . :

Description . . . . . : Microsoft ISATAP Adapter

Physical Address. . . . . : 00-00-00-00-00-00-00-E0

DHCP Enabled. . . . . : No

Autoconfiguration Enabled . . . . : Yes

C:\Users\User.TERMINAL90>tracert 172.16.4.100

Tracing route to Termina100 [172.16.4.100]

over a maximum of 30 hops:

1 4 ms 1 ms 1 ms Termina100 [172.16.4.100]

Trace complete.

C:\Users\User.TERMINAL90>arp -a

Interface: 172.16.4.81 --- 0xb

Internet Address	Physical Address	Type
172.16.4.67	b0-6e-bf-2c-1f-25	dynamic
172.16.4.75	b0-6e-bf-2c-21-a4	dynamic
172.16.4.83	b0-6e-bf-2c-1f-5b	dynamic
172.16.4.84	b0-6e-bf-2c-22-7c	dynamic
172.16.4.90	60-45-cb-86-aa-01	dynamic
172.16.4.91	b0-6e-bf-2c-25-ea	dynamic
172.16.4.94	60-45-cb-87-3f-37	dynamic
172.16.4.95	b0-6e-bf-2c-25-da	dynamic
172.16.4.96	60-45-cb-87-3e-31	dynamic
172.16.4.97	60-45-cb-87-42-ba	dynamic
172.16.4.98	60-45-cb-87-3f-b5	dynamic
172.16.4.100	b0-6e-bf-2c-22-98	dynamic
172.16.4.101	b0-6e-bf-2c-24-f0	dynamic
172.16.4.102	60-45-cb-87-3f-ba	dynamic
172.16.4.103	60-45-cb-87-49-9a	dynamic

172.16.4.104	60-45-cb-87-47-5f	dynamic
172.16.4.119	b0-6e-bf-d2-e4-2e	dynamic
172.16.4.130	b0-6e-bf-2c-1f-2a	dynamic
172.16.4.137	b0-6e-bf-2c-1f-92	dynamic
172.16.4.140	60-45-cb-87-4a-82	dynamic
172.16.4.141	b0-6e-bf-2c-1f-5e	dynamic
172.16.4.254	64-64-9b-1f-1d-41	dynamic
172.16.5.255	ff-ff-ff-ff-ff-ff	static
224.0.0.22	01-00-5e-00-00-16	static
224.0.0.252	01-00-5e-00-00-fc	static
239.255.255.250	01-00-5e-7f-ff-fa	static

C:\Users\User.TERMINAL90>netstat

Active Connections

Proto	Local Address	Foreign Address	State
Tcp	172.16.4.90:49223	maa05s04-in-f3: http	SYS-SENT

C:\Users\User.TERMINAL90>netstat -s

IPv4 Statistics

Packets Received	= 62188
Received Header Errors	= 0
Received Address Errors	= 303
Datagrams Forwarded	= 0
Unknown Protocols Received	= 0
Received Packets Discarded	= 2195
Received Packets Delivered	= 62211
Output Requests	= 10301
Routing Discards	= 0
Discarded Output Packets	= 0
Output Packet No Route	= 0
Reassembly Required	= 0
Reassembly Successful	= 0
Reassembly Failures	= 0
Datagrams Successfully Fragmented	= 0

Datagrams Failing Fragmentation = 0
Fragments Created = 0
IPv6 Statistics
Packets Received = 5201
Received Header Errors = 0
Received Address Errors = 11
Datagrams Forwarded = 0
Unknown Protocols Received = 0
Received Packets Discarded = 28
Received Packets Delivered = 5244
Output Requests = 187
Routing Discards = 0
Discarded Output Packets = 0
Output Packet No Route = 2
Reassembly Required = 0
Reassembly Successful = 0
Reassembly Failures = 0
Datagrams Successfully Fragmented = 0
Datagrams Failing Fragmentation = 0
Fragments Created = 0
ICMPv4 Statistics

	Received	Sent
Messages	81	81
Errors	0	0
Destination Unreachable	0	0
Time Exceeded	0	0
Parameter Problems	0	0
Source Quenches	0	0
Redirects	0	0
Echo Replies	37	44
Echos	44	37
Timestamps	0	0
Timestamp Replies	0	0
Address Masks	0	0

Address Mask Replies	0	0
Router Solicitations	0	0
Router Advertisements	0	0

ICMPv6 Statistics

	Received	Sent
Messages	83	20
Errors	0	0
Destination Unreachable	0	0
Packet Too Big	0	0
Time Exceeded	0	0
Parameter Problems	0	0
Echos	0	0
Echo Replies	0	0
MLD Queries	0	0
MLD Reports	0	0
MLD Dones	0	0
Router Solicitations	0	6
Router Advertisements	0	0
Neighbor Solicitations	5	7
Neighbor Advertisements	78	7
Redirects	0	0
Router Renumberings	0	0

TCP Statistics for IPv4

Active Opens	= 41
Passive Opens	= 776
Failed Connection Attempts	= 886
Reset Connections	= 28
Current Connections	= 0
Segments Received	= 43949
Segments Sent	= 8067
Segments Retransmitted	= 1772

TCP Statistics for IPv6

Active Opens	= 0
Passive Opens	= 0

Failed Connection Attempts = 0
Reset Connections = 0
Current Connections = 0
Segments Received = 0
Segments Sent = 0
Segments Retransmitted = 0

UDP Statistics for IPv4

Datagrams Received = 15918
No Ports = 2195
Receive Errors = 0
Datagrams Sent = 371

UDP Statistics for IPv6

Datagrams Received = 5079
No Ports = 28
Receive Errors = 0
Datagrams Sent = 150

C:\Users\User.TERMINAL90>netstat -e

Interface Statistics

	Received	Sent
Bytes	176974128	2534916
Unicast packets	184740	40400
Non-unicast packets	137916	1404
Discards	0	0
Errors	0	0
Unknown protocols	0	

Result:

The above program has been executed successfully and the output is verified.

Ex.No:02	Password Cracking Tool
Date:	

Aim:

To use of password cracking tool: Join then Ripper, ophcrack. Verify the strength Of password using these tools.

Algorithm:

Step 1:Start the process.

Step 2:Open the administrator user account id.

Step 3:In command prompt type - net user administrator *.

Step 4:Type the password and retype the password. We get the command completed successfully and log off.

Step 5:open command prompt type net user administrator * and change the administrator password.

Step 6:Display the result.

Program:

Step1:

C:\users\Administrator>net user

User accounts for \\TERMINAL81

Administrator Guest test

The command completed successfully.

Step2:

C:\users\Adiministrator>net user administrator *

Type a password for the user: ########

Retype the password to confirm:########

The command completed successfully.

Result:

The above program has been executed successfully and the output is verified.

Ex.No:03	Implementation of Encryption and Decryption of a Caesar Cipher
Date:	

Aim:

To perform encryption and decryption of a Caesar cipher. Write a script to perform the operations.

Algorithm:

Step 1: Read the plain text from the user.

Step 2: Read the key value from the user.

Step 3: If the key is positive then encrypt the text by adding character in the plain text.

Step 4: Else subtract the key from the plain text.

Program:

```
#include<stdio.h>
#include<string.h>
#include<conio.h>
#include<ctype.h>
void main()
{
char plain[10],cipher[10];
intkey,i,length;
int result;
clrscr();
printf("\n enter the plain text:");
scanf("%s",plain);
printf("\n enter the key value:");
scanf("%d",&key);
printf("\n \n \t PLAIN TEXT: %s",plain);
printf("\n \n \t ENCRYPED TEXT:");
for(i=0,length=strlen(plain);i<length;i++)
{
cipher[i]=plain[i]+key;
if(isupper(plain[i]) && (cipher[i]>'Z'))
cipher[i]=cipher[i]-26;
if(islower(plain[i]) && (cipher[i]>'z'))
```

```
cipher[i]=cipher[i]-26;
printf("%c",cipher[i]);
}
printf("\n \n \t AFTER DECRYPTION:");
for(i=0;i<length;i++)
{
plain[i]=cipher[i]-key;
if(isupper(cipher[i]) && (plain[i]<'A'))
plain[i]=plain[i]+26;
if(islower(cipher[i]) && (plain[i]<'a'))
plain[i]=plain[i]+26;
printf("%c",plain[i]);
}
getch();
}
```

Output:

Enter the plain text: security

Enter the key value: 3

PLAIN TEXT: security

ENCRYPTED TEXT: vhfxulwb

AFTER DECRYPTION: security

Result:

The above program has been executed successfully and the output is verified.

Ex.No:04	Implementation of Encryption and Decryption of
Date:	Rail Fence Cipher

Aim:

To write a script to implement Encryption and Decryption of rail fence cipher.

Algorithm:

Step 1:Read the plain text.

Step 2: Arrange the plain text in row columnar matrix format.

Step 3:Now read the keyboard depending on the number of Columns of the plain text.

Step 4:Arrange the characters of the keyword in second order and the corresponding columns of the plain text.

Step 5: Read the characters of row wise or column wise in the former order to get the cipher text.

Program:

```
#include<stdio.h>
#include<conio.h>
#include<string.h>
void main()
{
inti,j,k,l;
char a[20],c[20],d[20];
clrscr();
printf("\n\t\t RAIL FENCE TECHNIQUE");
printf("\n\nEnter the input string:");
gets(a);
 l=strlen(a);
/*ciphering*/
for(i=0,j=0;i<l;i++)
{
if(i%2==0)
c[j++]=a[i];
}
for(i=0;i<l;i++)
{
if(i%2==1)
c[j++]=a[i];
```

```
}
c[j]='\0';
printf("\n Cipher text after applying rail fence:");
printf("\n%s",c);
/*Deciphering*/
if(l%2==0)
k=l/2;
else
 k=(l/2)+1;
for(i=0,j=0;i<k;i++)
{
d[j]=c[i];
 j=j+2;
}
for(i=k,j=1;i<l;i++){
d[j]=c[i];
 j=j+2;
 }
d[l]='\0';
printf("\nText after decryption:");
printf("%s",d);
getch();
}
```

Output:

RAIL FENCE TECHNIQUE

Enter the input string: hackers

Cipher text after applying rail fence: hcesakr

Text after decryption: hackers

Result:

The above program has been executed successfully and the output is verified.

Ex.No:05	Nmap/Zenmap
Date:	

Aim:

Use nmap/zen map to analyse a remote mechanism.

Algorithm:

Step 1: Start the process.

Step 2: Open nmap/zenmap tool.

Step 3:In target address bar type the IP address 172.16.4.66 - 80 and Select ping scan→ scan.

Step 4: The nmap output for the ping scan is displayed on the screen.

Step 5: The ports/hosts shows the version state services for the ping Scan.

Step 6: click the Topology option,the diagram shows the connected Terminals IP address.

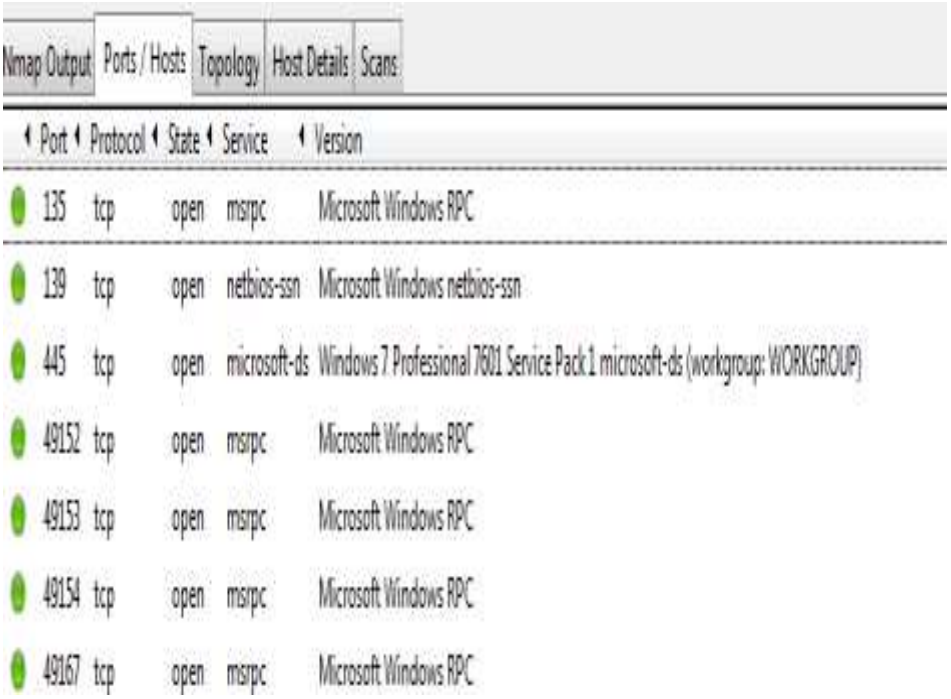
Step 7: using fisheye zoom option the diagram will get clearly.

Step 8: Stop the process.

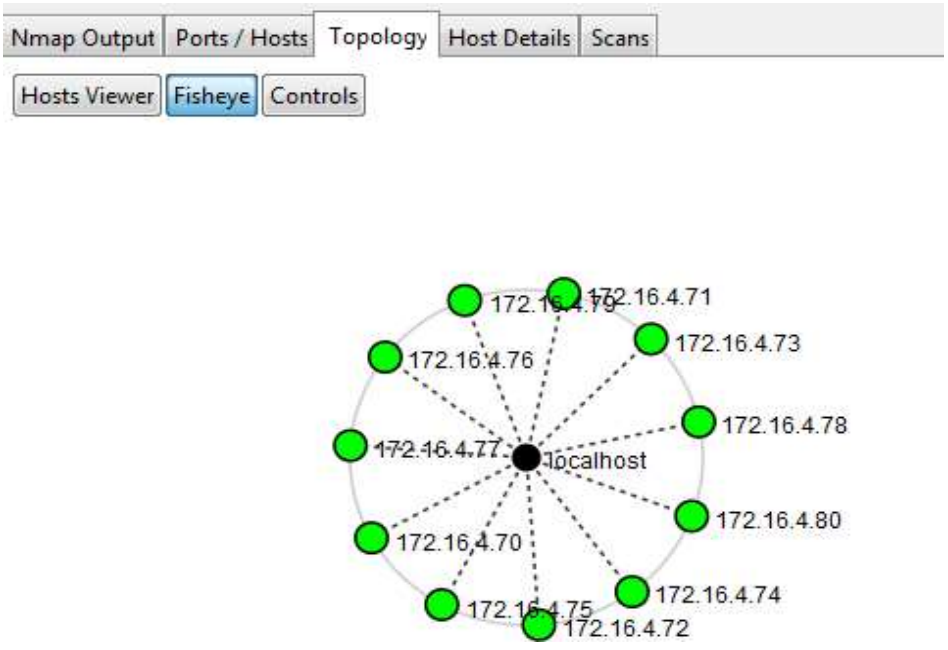
Program:

Nmap output:

Service:



Topology:



Result:

The above program has been executed successfully and the output is verified.

Ex.No:06	Burp Proxy Tool
Date:	

Aim:

To use burp proxy to capture and modify the message.

Algorithm:

Step1: start the process.

Step2: Open the Burp proxy tool.

Step3: First, ensure that Burp is correctly configured with your browser.
In the Burp Proxy "Intercept" tab ensure "Intercept is off".

Step4**:** To visit the web application you are testing in your browser.
Access the log in page of the web application.

Step5: Return to Burp. In the Proxy Intercept tab, ensure "Intercept is on".

Step6: Enter login details in to the login form and submit the request.
by clicking “Login”.

Step7: Return to Burp. The raw request details should now be displayed
In the Proxy “Intercept" tab. Right click on the request to bring up the context menu and click “Do an active scan.”

Step8: The results of the scan are displayed in the Target “Site map” tab.
The Scanner has detected that the application has an issue; “Clear text submission of password”.

Step9: By clicking on an individual issue we can view a description of the vulnerability and suggested remediation in the “Advisory tab”. The full request and response are also shown.

Step10: Burp Scanner checks for a variety of types of data exposure, SSH keys, credit card numbers and email addresses, etc.

Program:

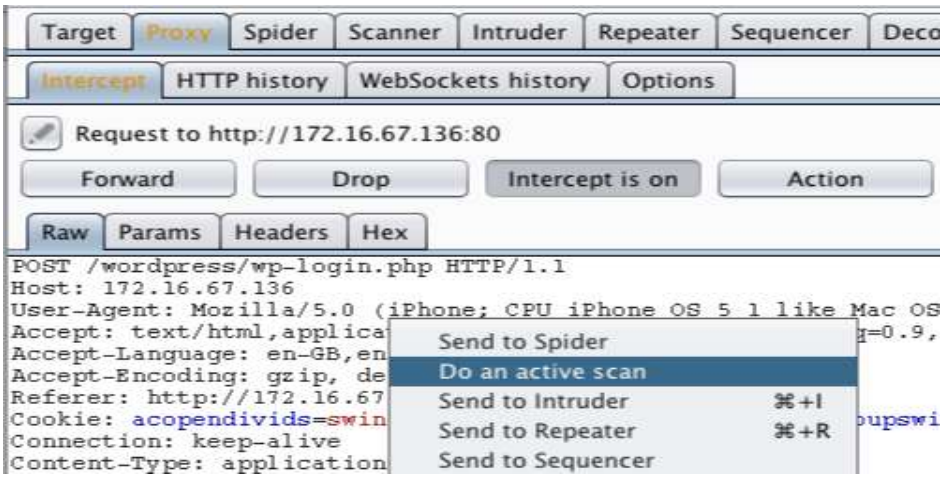
Step1:

Step2:

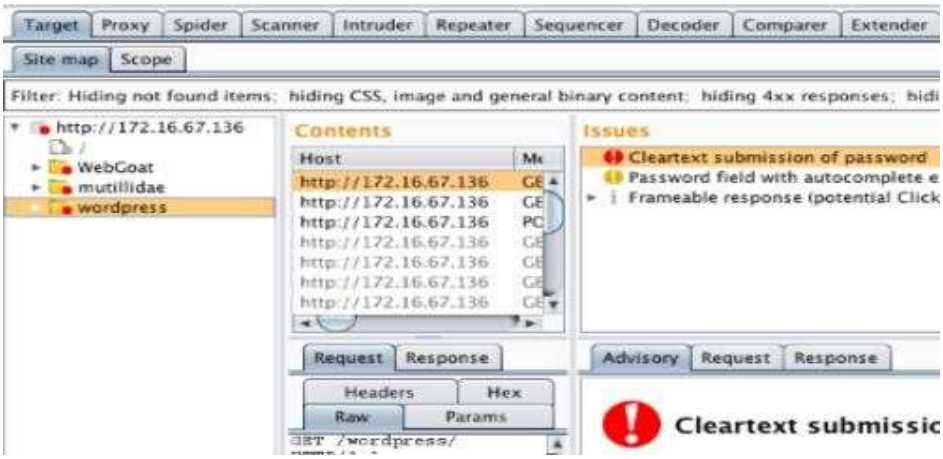
Step3:

Step4

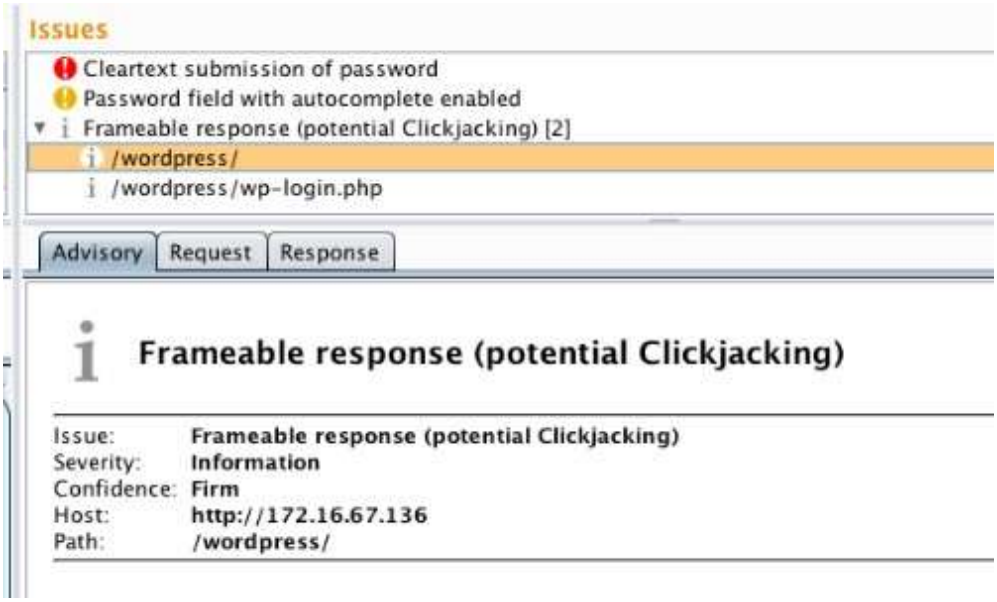
Step5:



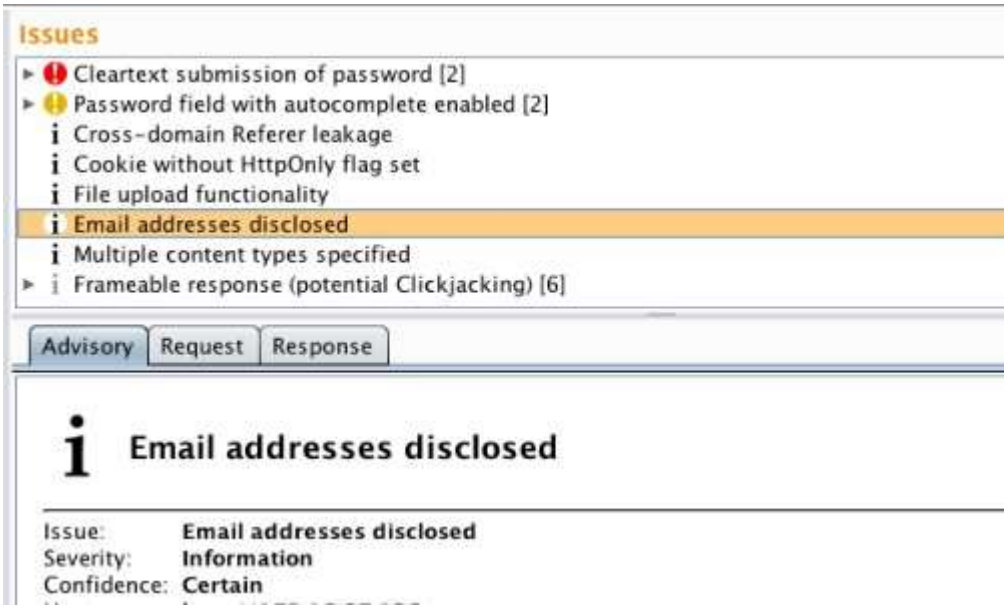
Step6:



Step7:



Step8:



Result:

The above program has been executed successfully and output is verified.

Ex.No:07	Protected Word Document
Date:	

Aim:

To demonstrate sending of protected word document.

Algorithm:

Step 1: Open the new Microsoft word document.

Step 2: click the office button → save as → word document→

Tools→general options.

Step 3: In general options → type password to open and modify a

Document and re-enter the password to open and modify the document.

Step 4: save the document and close the file.

Step 5: The protected word document is attached to mail-id.

Step 6: In separate mail send the password without mentioning the

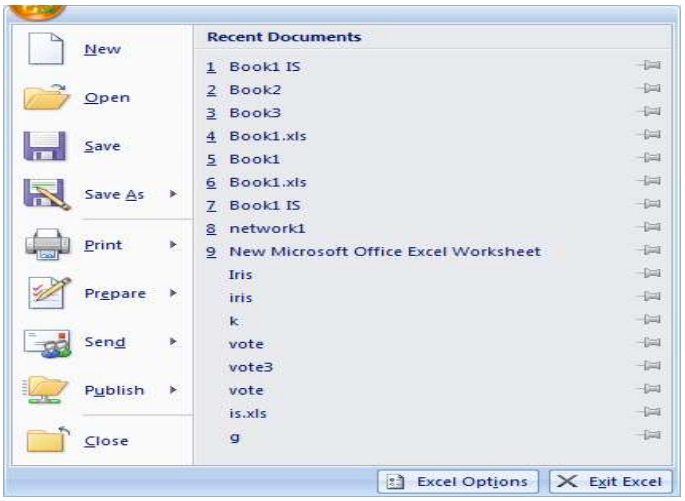
Subject.

Step 7: Stop the process.

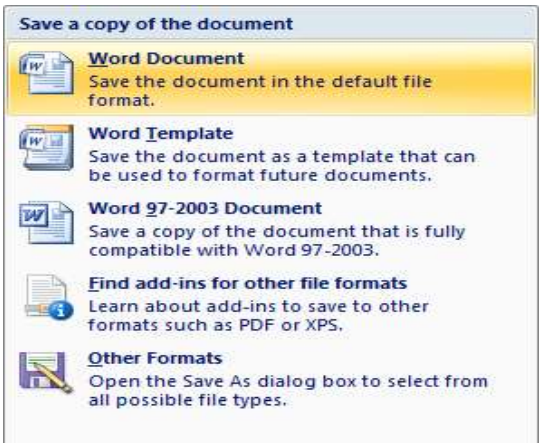
Program:

Step1:

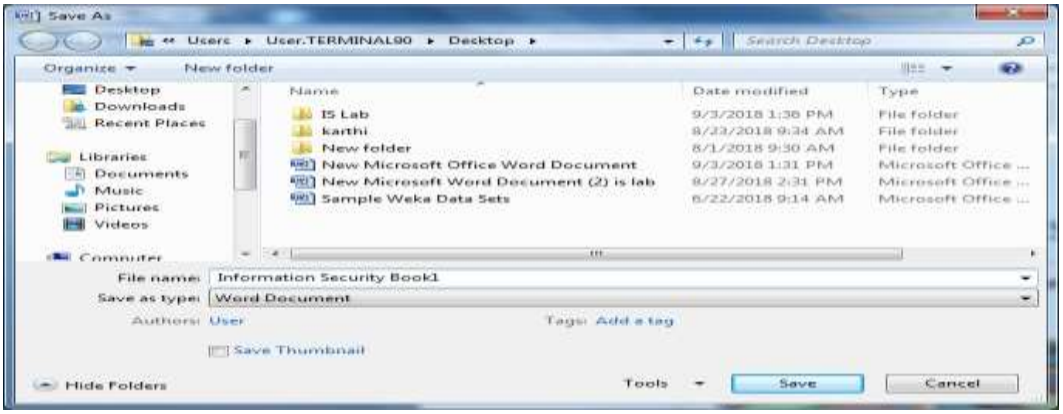
Step2:



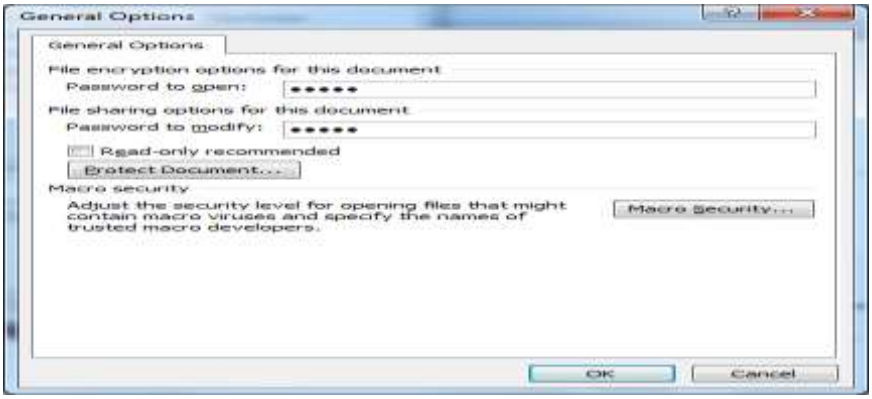
Step3:



Step4:



Step5:



Step6:

Step7:

Result: The above program has been executed successfully and the output is verified.

Ex.No:08	Digitally Signed Document
Date:	

Aim:

To demonstrate sending or a digitally signed document.

Algorithm:

Step 1: Open a word document.

Step 2: Type a content in word document with letter format.

Step 3: place the curser to insert the digitally signed signature.

Step 4: Click insert menu →signature line→signature set up dialog Box type the suggested signer, etc.

Step 5: The signature line is created in the word document and right click → select sign option→select image.

Step 6: Finally the digitally signed document is created with date line

Leave Letter

From

yyyyyyyy,

III B.SC (cs),

Department of computer science,

Karpagam Academy of Heigher Education,

Eachanari,

Coimbatore-641021.

To

The Head of the department,

Department of computer science,

Karpagam Academy of Heigher Education,

Eachanari,

Coimbatore-641021.

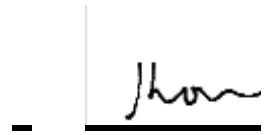
Respected Madam,

I am suffering from fever. I am unable to attend the class So, leave for grant me two days with your permission (03/09/2018 to 04/09/2018).

Thanking You,

Your's faithfully,

9/10/2018



yyyyyyyy

Result:

The above program has been executed successfully. And output is verified.

Ex.No:09	Protected Excel Document
Date:	

Aim:

To demonstrate sending of a protected worksheet.

Algorithm:

Step 1: Open the excel worksheet.

Step 2: click the office button→ prepare → Encryption document.

Step 3: enter the password and re-enter the password in encryption Document.

Step 4: Enter the data in worksheet and close the file.

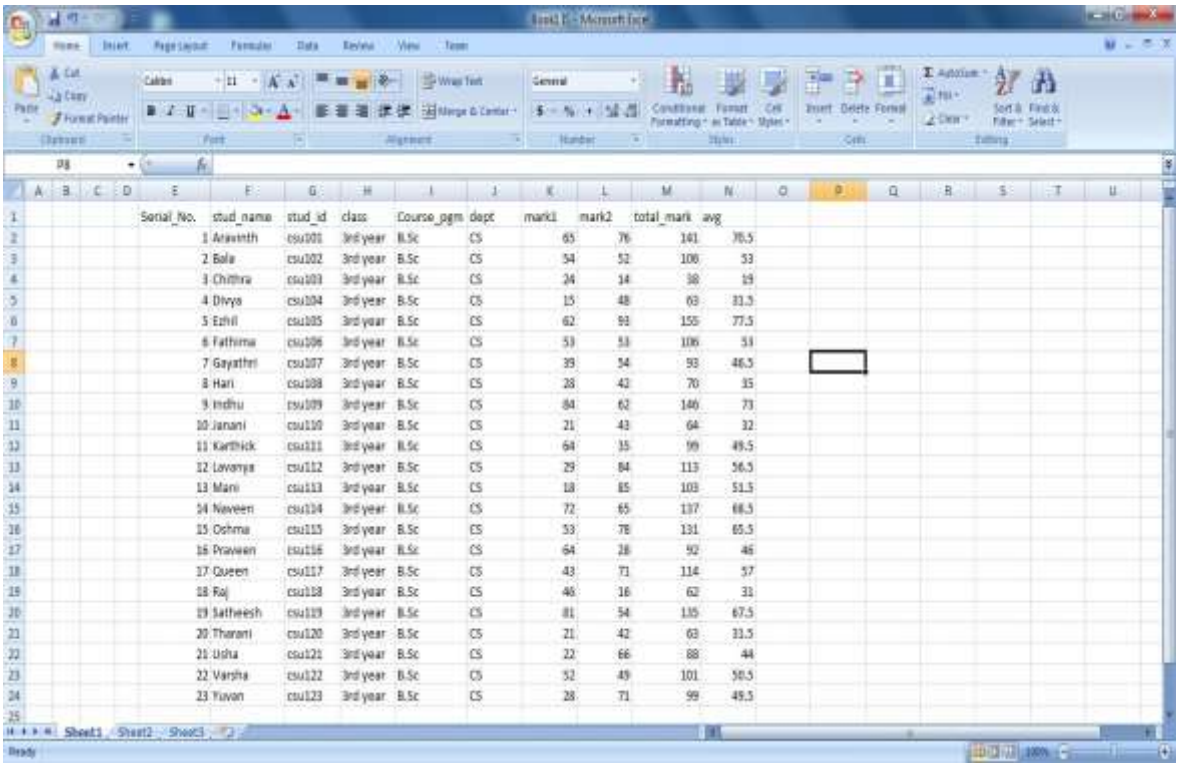
Step 5: The protected excel document is attached to mail-id.

Step 6: In separate mail send the password without mentioning the Subject.

Step 7: Stop the process.

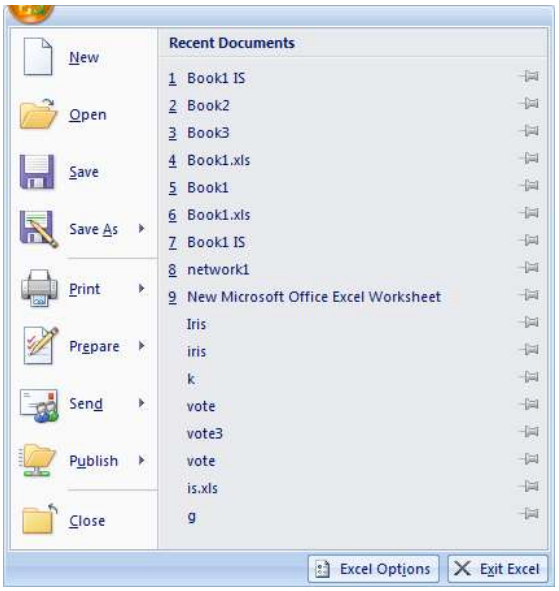
Program:

Step1:



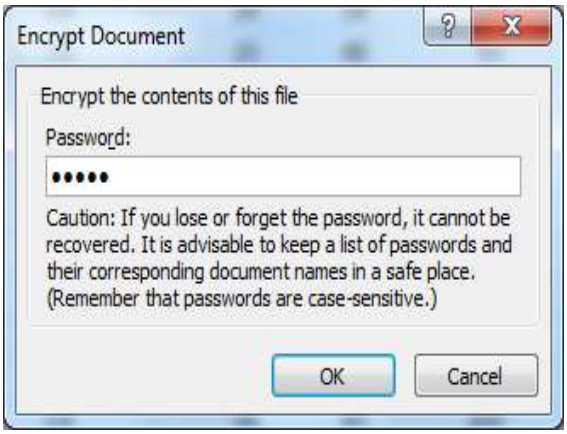
Serial_No.	stud_name	stud_id	class	Course_pgm	dept	mark1	mark2	total_mark	avg
1	Aravinth	csu101	3rd year	B.Sc	CS	65	76	141	70.5
2	Bala	csu102	3rd year	B.Sc	CS	54	52	106	53
3	Chithra	csu103	3rd year	B.Sc	CS	24	14	38	19
4	Divya	csu104	3rd year	B.Sc	CS	15	48	63	31.5
5	Ezhil	csu105	3rd year	B.Sc	CS	62	93	155	77.5
6	Fathima	csu106	3rd year	B.Sc	CS	53	53	106	53
7	Gayathri	csu107	3rd year	B.Sc	CS	39	54	93	46.5
8	Hari	csu108	3rd year	B.Sc	CS	28	42	70	35
9	Indhu	csu109	3rd year	B.Sc	CS	84	62	146	73
10	Janani	csu110	3rd year	B.Sc	CS	21	43	64	32
11	Karthick	csu111	3rd year	B.Sc	CS	64	35	99	49.5
12	Lavanya	csu112	3rd year	B.Sc	CS	29	84	113	56.5
13	Mani	csu113	3rd year	B.Sc	CS	18	85	103	51.5
14	Naveen	csu114	3rd year	B.Sc	CS	72	65	137	68.5
15	Oshma	csu115	3rd year	B.Sc	CS	53	78	131	65.5
16	Praveen	csu116	3rd year	B.Sc	CS	64	28	92	46
17	Queen	csu117	3rd year	B.Sc	CS	43	71	114	57
18	Raj	csu118	3rd year	B.Sc	CS	46	16	62	31
19	Satheesh	csu119	3rd year	B.Sc	CS	81	54	135	67.5
20	Tharani	csu120	3rd year	B.Sc	CS	21	42	63	31.5
21	Usha	csu121	3rd year	B.Sc	CS	22	66	88	44
22	Varsha	csu122	3rd year	B.Sc	CS	52	49	101	50.5
23	Yuvan	csu123	3rd year	B.Sc	CS	28	71	99	49.5

Step2:



Step3:

Step4:



Step5:

Result:

The above program has been executed successfully and the outputis verified.

Ex.No:10	Steganography Tool
Date:	

Aim:

To demonstrate use of stenography tool.

Algorithm:

Step 1: Start →Xiao Steganography 2.6.1→ add file → load target file.

Step 2: The image is loaded in the target file.

Step 3: create a worksheet and save the file with .xls extension.

Step 4: click next option button →add file→browse the worksheet.

Step 5: click next option button →choose encryption and hashing

Algorithms and type password in given textbox.

Step 6: click next option button → save the file → save as type →bmp

File type →save→finish.

Step 7: click target file→select source file→next option button→

Enter the password to unlock the file.

Step 8: click the extract file option button and save the file with

.xls extension→save.

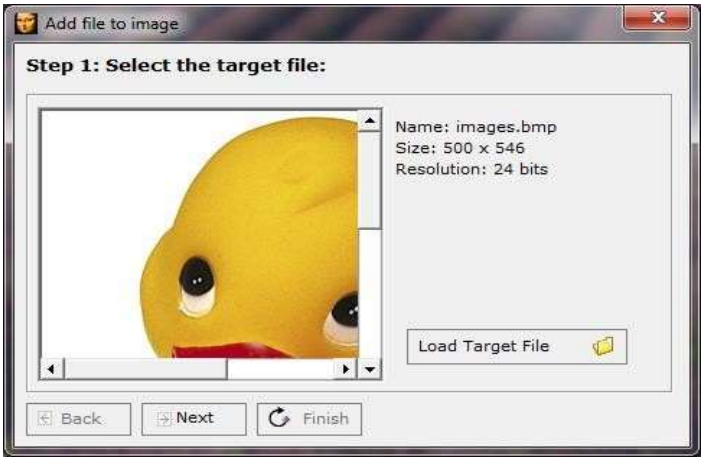
Step 9: we get, file extract was successful! Message.

Step 10: Open the source file in worksheet.

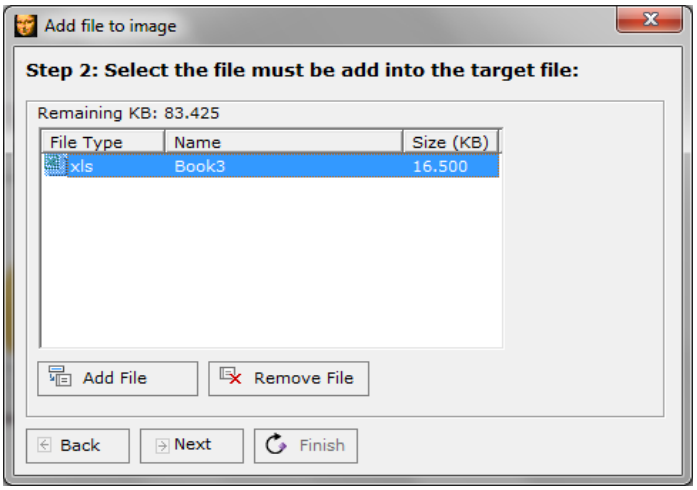
Program:

Step1:

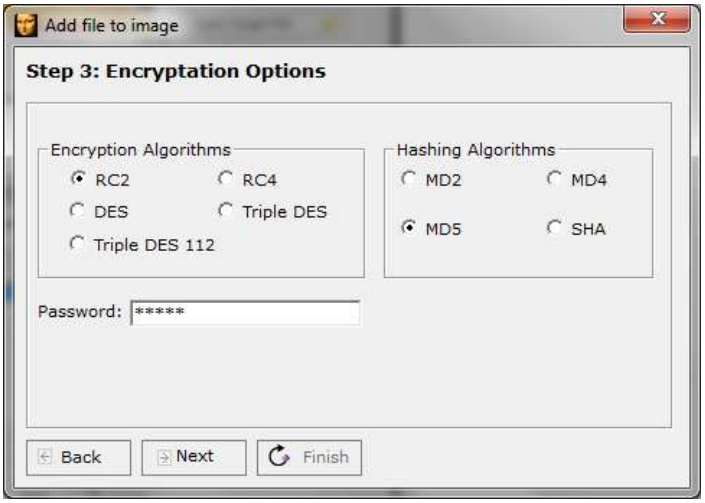
Step2:



Step3:

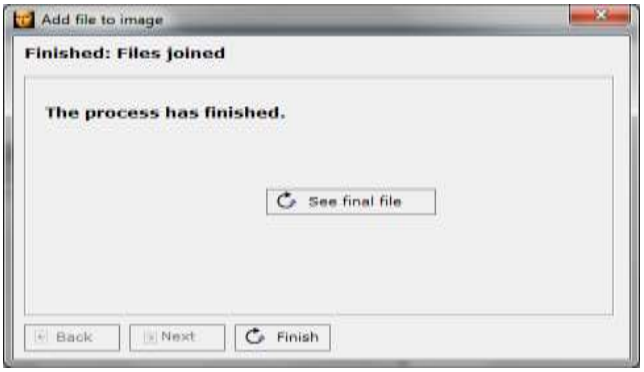


Step4:

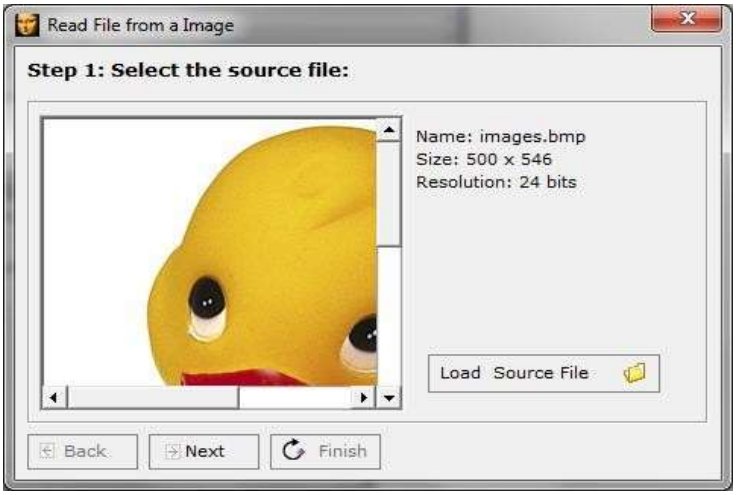


Step5:

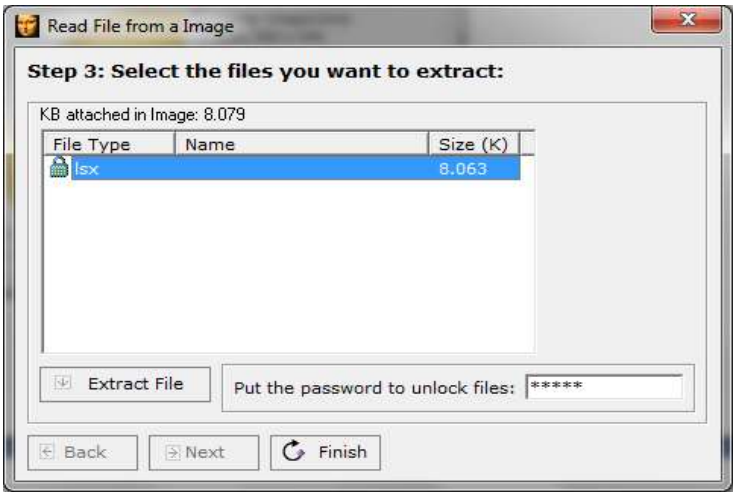
Step6:



Step7:



Step8:



Step9:

Result:

The above program has been executed successfully and the output is verified.

Ex.No:11	GPG Utility
Date:	

Aim:

To demonstrate use of GPG utility for singing and encrypting purposes.

Algorithm:

Step1: start the process.

Step2: Open the GPG command line tool

Step3**:** Open Git Bash.

Step4: Generate a GPG key pair. Since there are multiple versions of GPG,to

Needed a consult the relevant *man page* to find the appropriate key generation

command.

Step5: Paste the text below to generate a GPG key pair.

Step6: The gpg --full-generate-key command doesn't work. Paste the text below

Step7:At the prompt, specify the kind of key we want, or press Enter to accept

The default RSA and RSA Algorithm.

Step8:Enter the desired key size. The maximum key size of 4096.

Step9:Enter the length of time the key should be valid. Press Enter to specify

The default selection, indicating that the key doesn't expire.

Step10:Verify that your selections are correct.Enter your user ID information.

Type a secure passphrase.

Step11:Use the gpg --list-secret-keys --keyid-format LONG command to list

GPG keys for which you have both a public and private key. A private key

is required for signing commits or tags.

Step12:From the list of GPG keys, copy the GPG key ID to use.

the GPG key ID is 3AA5C34371567BD2:

Step13:Paste the text below, substituting in the GPG key ID to use.

the GPG key ID is 3AA5C34371567BD2:

Step14:If we have multiple keys or are attempting to sign commits or tags with a key that doesn't match your committer identity the signing key.

Step15:When committing changes in our local branch, add the -S flag to the git commit command:

Step16:If we're using GPG, after we create our commit, provide the passphrase you set up when you generated your GPG key.

Step17:When we've finished creating commits locally, push them to our

remote repository on GitHub:

Step18:On GitHub, navigate to our pull request.

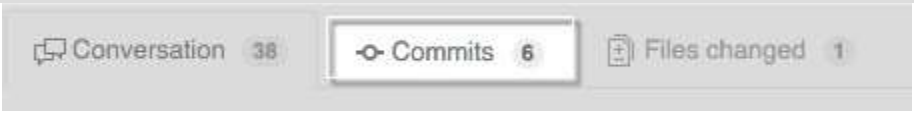
Step19:On the pull request, click **Commits**.

Step20:To view more detailed information about the verified signature, click Verified.

Program:

```
o  gpg --full-generate-key
o  gpg --default-new-key-algo rsa4096 --gen-key
2. gpg --list-secret-keys --keyid-format LONG
3. gpg --list-secret-keys --keyid-format LONG
4. /Users/hubot/.gnupg/secring.gpg
5. ------------------------------------
6. sec   4096R/3AA5C34371567BD2 2016-03-10 [expires: 2017-03-10]
7. uid                          Hubot
8. ssb   4096R/42B317FD4BA89E7A 2016-03-10
9. gpg --armor --export 3AA5C34371567BD2
10.    # Prints the GPG key ID, in ASCII armor format
```

```
11.    git commit -S -m your commit message
12.    # Creates a signed commit
13.    git push
14.    # Pushes your local commits to the remote repository
```



Result:

The above program has been executed successfully and the output is verified.