Ex. No : 1(i) Date :	Perform Encryption and Decryption Using Ceaser Cipher

PROGRAM:

CaesarCipher.java

```
class caesarCipher {
public static String encode(String enc, int offset) {
offset = offset % 26 + 26;
StringBuilder encoded = new StringBuilder();
for (char i : enc.toCharArray()) {
if (Character.isLetter(i)) {
if (Character.isUpperCase(i)) {
encoded.append((char) ('A' + (i - 'A' + offset) % 26));
} else {
encoded.append((char) ('a' + (i - 'a' + offset) % 26));
}
} else {
encoded.append(i);
}
}
return encoded.toString();
}

public static String decode(String enc, int offset) {
return encode(enc, 26 - offset);
}

public static void main(String[] args) throws java.lang.Exception {
String msg = "Anna University";
System.out.println("Simulating Caesar Cipher\n----------------------");
System.out.println("Input : " + msg);
System.out.printf("Encrypted Message : ");
System.out.println(caesarCipher.encode(msg, 3));
System.out.printf("Decrypted Message : ");
System.out.println(caesarCipher.decode(caesarCipher.encode(msg, 3), 3));
}
}
```

OUTPUT:

Simulating Caesar Cipher

Input : Anna University
Encrypted Message : Dqqd Xqlyhuvlwb
Decrypted Message : Anna University

Ex. No : 1(ii) Date :	Perform Encryption and Decryption Using Playfair Cipher

PROGRAM:

playfairCipher.java

```
import java.awt.Point;

class playfairCipher {
private static char[][] charTable;
private static Point[] positions;

private static String prepareText(String s, boolean chgJtoI) {
s = s.toUpperCase().replaceAll("[^A-Z]", "");
return chgJtoI ? s.replace("J", "I") : s.replace("Q", "");
}

private static void createTbl(String key, boolean chgJtoI) {
charTable = new char[5][5];
positions = new Point[26];
String s = prepareText(key + "ABCDEFGHIJKLMNOPQRSTUVWXYZ",
chgJtoI);
int len = s.length();
for (int i = 0, k = 0; i < len; i++) {
char c = s.charAt(i);

if (positions[c - 'A'] == null) {
charTable[k / 5][k % 5] = c;
positions[c - 'A'] = new Point(k % 5, k / 5);
k++;
}
}
}

private static String codec(StringBuilder txt, int dir) {
int len = txt.length();
for (int i = 0; i < len; i += 2) {
char a = txt.charAt(i);
```

```
char b = txt.charAt(i + 1);
int row1 = positions[a - 'A'].y;
int row2 = positions[b - 'A'].y;
int col1 = positions[a - 'A'].x;
int col2 = positions[b - 'A'].x;
if (row1 == row2) {
col1 = (col1 + dir) % 5;
col2 = (col2 + dir) % 5;
} else if (col1 == col2) {

row1 = (row1 + dir) % 5;
row2 = (row2 + dir) % 5;
} else {
int tmp = col1;
col1 = col2;
col2 = tmp;
}
txt.setCharAt(i, charTable[row1][col1]);
txt.setCharAt(i + 1, charTable[row2][col2]);
}
return txt.toString();
}

private static String encode(String s) {
StringBuilder sb = new StringBuilder(s);
for (int i = 0; i < sb.length(); i += 2) {
if (i == sb.length() - 1) {
sb.append(sb.length() % 2 == 1 ? 'X' : "");
} else if (sb.charAt(i) == sb.charAt(i + 1)) {
sb.insert(i + 1, 'X');
}
}
return codec(sb, 1);
}

private static String decode(String s) {
return codec(new StringBuilder(s), 4);
}
```

```
public static void main(String[] args) throws java.lang.Exception {
String key = "CSE";
String txt = "Security Lab"; /* make sure string length is even */ /* change J to I */
boolean chgJtoI = true;
createTbl(key, chgJtoI);
String enc = encode(prepareText(txt, chgJtoI));
System.out.println("Simulating Playfair Cipher\n----------------------");
System.out.println("Input Message : " + txt);
System.out.println("Encrypted Message : " + enc);
System.out.println("Decrypted Message : " + decode(enc));
}
}
```

OUTPUT:

```
Simulating Playfair Cipher
----------------------
Input Message : Security Lab
Encrypted Message : EABPUGYANSEZ
Decrypted Message : SECURITYLABX
```

Ex. No : 1(iii) Date :	Perform Encryption and Decryption Using Hill Cipher

PROGRAM:

HillCipher.java

```
class hillCipher {
/* 3x3 key matrix for 3 characters at once */
public static int[][] keymat = new int[][] { { 1, 2, 1 }, { 2, 3, 2 },
{ 2, 2, 1 } }; /* key inverse matrix */
public static int[][] invkeymat = new int[][] { { -1, 0, 1 }, { 2, -1, 0 }, { -2, 2, -1 }
};
public static String key = "ABCDEFGHIJKLMNOPQRSTUVWXYZ";

private static String encode(char a, char b, char c) {
String ret = "";
int x, y, z;
int posa = (int) a - 65;
int posb = (int) b - 65;
int posc = (int) c - 65;
x = posa * keymat[0][0] + posb * keymat[1][0] + posc * keymat[2][0];
y = posa * keymat[0][1] + posb * keymat[1][1] + posc * keymat[2][1];
z = posa * keymat[0][2] + posb * keymat[1][2] + posc * keymat[2][2];
a = key.charAt(x % 26);
b = key.charAt(y % 26);
c = key.charAt(z % 26);
ret = "" + a + b + c;
return ret;
}
private static String decode(char a, char b, char c) {
String ret = "";
int x, y, z;
int posa = (int) a - 65;
int posb = (int) b - 65;
int posc = (int) c - 65;
x = posa * invkeymat[0][0] + posb * invkeymat[1][0] + posc * invkeymat[2][0];
y = posa * invkeymat[0][1] + posb * invkeymat[1][1] + posc * invkeymat[2][1];
z = posa * invkeymat[0][2] + posb * invkeymat[1][2] + posc * invkeymat[2][2];
a = key.charAt((x % 26 < 0) ? (26 + x % 26) : (x % 26));
b = key.charAt((y % 26 < 0) ? (26 + y % 26) : (y % 26));
```

```
c = key.charAt((z % 26 < 0) ? (26 + z % 26) : (z % 26));
ret = "" + a + b + c;
return ret;
}
public static void main(String[] args) throws java.lang.Exception {
String msg;
String enc = "";
String dec = "";
int n;
msg = ("SecurityLaboratory");
System.out.println("simulation of Hill Cipher\n-------------------------");
System.out.println("Input message : " + msg);
msg = msg.toUpperCase();
msg = msg.replaceAll("\\s", "");
/* remove spaces */ n = msg.length() % 3;
/* append padding text X */ if (n != 0) {
for (int i = 1; i <= (3 - n); i++) {
msg += 'X';
}   }
System.out.println("padded message : " + msg);
char[] pdchars = msg.toCharArray();
for (int i = 0; i < msg.length(); i += 3) {
enc += encode(pdchars[i], pdchars[i + 1], pdchars[i + 2]);
}
System.out.println("encoded message : " + enc);
char[] dechars = enc.toCharArray();
for (int i = 0; i < enc.length(); i += 3) {
dec += decode(dechars[i], dechars[i + 1], dechars[i + 2]);
}
System.out.println("decoded message : " + dec);
}}
```

OUTPUT:

```
Simulating Hill Cipher
------------------------------
Input Message : SecurityLaboratory
Padded Message : SECURITYLABORATORY
Encrypted Message : EACSDKLCAEFQDUKSXU
Decrypted Message : SECURITYLABORATORY
```

Ex. No : 1(iv) Date :	Perform Encryption and Decryption Using Vigenere Cipher

PROGRAM:

vigenereCipher.java

```
public class vigenereCipher {
static String encode(String text, final String key) {
String res = "";
text = text.toUpperCase();
for (int i = 0, j = 0; i < text.length(); i++) {
char c = text.charAt(i);
if (c <'A' || c >'Z') {
continue;
}
res += (char) ((c + key.charAt(j) - 2 * 'A') % 26 + 'A');
j = ++j % key.length();
}
return res;
}

static String decode(String text, final String key) {
String res = "";
text = text.toUpperCase();
for (int i = 0, j = 0; i < text.length(); i++) {
char c = text.charAt(i);
if (c <'A' || c >'Z') {
continue;
}
res += (char) ((c - key.charAt(j) + 26) % 26 + 'A');
j = ++j % key.length();
}
return res;
}

public static void main(String[] args) throws java.lang.Exception {
String key = "VIGENERECIPHER";
String msg = "SecurityLaboratory";
System.out.println("Simulating Vigenere Cipher\n----------------------");
System.out.println("Input Message : " + msg);
```

```
String enc = encode(msg, key);
System.out.println("Encrypted Message : " + enc);
System.out.println("Decrypted Message : " + decode(enc, key));
}
}
```

OUTPUT:

```
Simulating Vigenere Cipher
------------------------
Input Message : SecurityLaboratory
Encrypted Message : NMIYEMKCNIQVVROWXC
Decrypted Message : SECURITYLABORATORY
```

Ex. No : 2(i) Date :	Perform Encryption and Decryption Using Rail Fence Cipher Transposition Technique

PROGRAM:

railFenceCipher.java

```
class railfenceCipherHelper {
int depth;

String encode(String msg, int depth) throws Exception {
int r = depth;
int l = msg.length();
int c = l / depth;
int k = 0;
char mat[][] = new char[r][c];
String enc = "";
for (int i = 0; i < c; i++) {
for (int j = 0; j < r; j++) {
if (k != l) {
mat[j][i] = msg.charAt(k++);
} else {
mat[j][i] = 'X';
}
}
}
for (int i = 0; i < r; i++) {
for (int j = 0; j < c; j++) {
enc += mat[i][j];
}
}
return enc;
}

String decode(String encmsg, int depth) throws Exception {
int r = depth;
int l = encmsg.length();
int c = l / depth;
int k = 0;
char mat[][] = new char[r][c];
String dec = "";
```

```
for (int i = 0; i < r; i++) {
for (int j = 0; j < c; j++) {
mat[i][j] = encmsg.charAt(k++);
}
}
for (int i = 0; i < c; i++) {
for (int j = 0; j < r; j++) {
dec += mat[j][i];
}
}
return dec;
}
}

class railFenceCipher {
public static void main(String[] args) throws java.lang.Exception {
railfenceCipherHelper rf = new railfenceCipherHelper();
String msg, enc, dec;
msg = "Anna University, Chennai";
int depth = 2;
enc = rf.encode(msg, depth);
dec = rf.decode(enc, depth);
System.out.println("Simulating Railfence Cipher\n-------------------------");
System.out.println("Input Message : " + msg);
System.out.println("Encrypted Message : " + enc);
System.out.printf("Decrypted Message : " + dec);
}
}
```

OUTPUT:

```
Simulating Railfence Cipher
---------------------------------------
Input Message : Anna University, Chennai
Encrypted Message : An nvriy hnanaUiest,Ceni
Decrypted Message : Anna University, Chennai
```

Ex. No : 2(ii) Date :	Perform Encryption and Decryption Using Row and Column Transformation Technique

PROGRAM:

TransCipher.java

```
import java.util.*;
class TransCipher {
public static void main(String args[]) {
Scanner sc = new Scanner(System.in);
System.out.println("Enter the plain text");
String pl = sc.nextLine();
sc.close();
String s = "";
int start = 0;
for (int i = 0; i < pl.length(); i++) {
if (pl.charAt(i) == '') {
s = s + pl.substring(start, i);
start = i + 1;
}
}
s = s + pl.substring(start);
System.out.print(s);
System.out.println();
// end of space deletion

int k = s.length();
int l = 0;
int col = 4;
int row = s.length() / col;
char ch[][] = new char[row][col];
for (int i = 0; i < row; i++) {
for (int j = 0; j < col; j++) {
if (l < k) {
ch[i][j] = s.charAt(l);
l++;
} else {
ch[i][j] = '#';
}
```

```
}
}
// arranged in matrix

char trans[][] = new char[col][row];
for (int i = 0; i < row; i++) {
for (int j = 0; j < col; j++) {
trans[j][i] = ch[i][j];
}
}

for (int i = 0; i < col; i++) {
for (int j = 0; j < row; j++) {
System.out.print(trans[i][j]);
}
}
// display
System.out.println();
}
}
```

OUTPUT:

```
Enter the plain text
Security Lab
SecurityLab
Sreictuy
```

Ex. No : 3 Date :	Data Encryption Standard (DES) Algorithm (User Message Encryption)

PROGRAM:

DES.java

```
import java.security.InvalidKeyException;
import java.security.NoSuchAlgorithmException;

import javax.crypto.BadPaddingException;
import javax.crypto.Cipher;
import javax.crypto.IllegalBlockSizeException;
import javax.crypto.KeyGenerator;
import javax.crypto.NoSuchPaddingException;
import javax.crypto.SecretKey;

public class DES
{
        public static void main(String[] argv) {

                try{
System.out.println("Message Encryption Using DES Algorithm\n-------");
                    KeyGenerator keygenerator = KeyGenerator.getInstance("DES");
SecretKey myDesKey = keygenerator.generateKey();
                    Cipher desCipher;
                    desCipher = Cipher.getInstance("DES/ECB/PKCS5Padding");
                    desCipher.init(Cipher.ENCRYPT_MODE, myDesKey);
                    byte[] text = "Secret Information ".getBytes();
                    System.out.println("Message [Byte Format] : " + text);
                    System.out.println("Message : " + new String(text));
                    byte[] textEncrypted = desCipher.doFinal(text);
                    System.out.println("Encrypted Message: " + textEncrypted);
                    desCipher.init(Cipher.DECRYPT_MODE, myDesKey);
                    byte[] textDecrypted = desCipher.doFinal(textEncrypted);
                    System.out.println("Decrypted Message: " + new
String(textDecrypted));
                }catch(NoSuchAlgorithmException e){
                        e.printStackTrace();
                }catch(NoSuchPaddingException e){
```

```
                e.printStackTrace();
            }catch(InvalidKeyException e){
                e.printStackTrace();
            }catch(IllegalBlockSizeException e){
                e.printStackTrace();
            }catch(BadPaddingException e){
                e.printStackTrace();
            }

        }
}
```

OUTPUT:

```
Message Encryption Using DES Algorithm
------------------------------------------------------
Message [Byte Format] : [B@4dcbadb4
Message : Secret Information
Encrypted Message: [B@504bae78
Decrypted Message: Secret Information
```

Ex. No : 4 Date :	Advanced Encryption Standard (AES) Algorithm (URL Encryption)

PROGRAM:

AES.java

```
import java.io.UnsupportedEncodingException;
import java.security.MessageDigest;
import java.security.NoSuchAlgorithmException;
import java.util.Arrays;
import java.util.Base64;
import javax.crypto.Cipher;
import javax.crypto.spec.SecretKeySpec;

public class AES {
private static SecretKeySpec secretKey;
private static byte[] key;
public static void setKey(String myKey) {
MessageDigest sha = null;
try {
key = myKey.getBytes("UTF-8");
sha = MessageDigest.getInstance("SHA-1");
key = sha.digest(key);
key = Arrays.copyOf(key, 16);
secretKey = new SecretKeySpec(key, "AES");
} catch (NoSuchAlgorithmException e) {
e.printStackTrace();
} catch (UnsupportedEncodingException e) {
e.printStackTrace();
}
}

public static String encrypt(String strToEncrypt, String secret) {
try {
setKey(secret);
Cipher cipher = Cipher.getInstance("AES/ECB/PKCS5Padding");
cipher.init(Cipher.ENCRYPT_MODE, secretKey);
return
Base64.getEncoder().encodeToString(cipher.doFinal(strToEncrypt.getBytes("UTF
-8")));
```

```
} catch (Exception e) {
System.out.println("Error while encrypting: " + e.toString());
}
return null;
}

public static String decrypt(String strToDecrypt, String secret) {
try {
setKey(secret);
Cipher cipher = Cipher.getInstance("AES/ECB/PKCS5PADDING");
cipher.init(Cipher.DECRYPT_MODE, secretKey);
return new String(cipher.doFinal(Base64.getDecoder().decode(strToDecrypt)));
} catch (Exception e) {
System.out.println("Error while decrypting: " + e.toString());
}
return null;
}
public static void main(String[] args) {
final String secretKey = "annaUniversity";
String originalString = "www.annauniv.edu";
String encryptedString = AES.encrypt(originalString, secretKey);
String decryptedString = AES.decrypt(encryptedString, secretKey);
System.out.println("URL Encryption Using AES Algorithm\n------------");
System.out.println("Original URL : " + originalString);
System.out.println("Encrypted URL : " + encryptedString);
System.out.println("Decrypted URL : " + decryptedString);
}
}
```

OUTPUT:

```
URL Encryption Using AES Algorithm
-----------------------------------------------
Original URL : www.annauniv.edu
Encrypted URL : vibpFJW6Cvs5Y+L7t4N6YWWe07+JzS1d3CU2h3mEvEg=
Decrypted URL : www.annauniv.edu
```

Ex. No : 5 Date :	RSA Algorithm

PROGRAM:

rsa.html

```
<html>

<head>
<title>RSA Encryption</title>
<meta name="viewport" content="width=device-width, initial-scale=1.0">
</head>

<body>
<center>
<h1>RSA Algorithm</h1>
<h2>Implemented Using HTML & Javascript</h2>
<hr>
<table>
<tr>
<td>Enter First Prime Number:</td>
<td><input type="number" value="53" id="p"></td>
</tr>
<tr>
<td>Enter Second Prime Number:</td>
<td><input type="number" value="59" id="q"></p>
</td>
</tr>
<tr>
<td>Enter the Message(cipher text):<br>[A=1, B=2,...]</td>
<td><input type="number" value="89" id="msg"></p>
</td>
</tr>
<tr>
<td>Public Key:</td>
<td>
<p id="publickey"></p>
</td>
</tr>
<tr>
<td>Exponent:</td>
```

```
<td>
<p id="exponent"></p>
</td>
</tr>
<tr>
<td>Private Key:</td>
<td>
<p id="privatekey"></p>
</td>
</tr>
<tr>
<td>Cipher Text:</td>
<td>
<p id="ciphertext"></p>
</td>
</tr>
<tr>
<td><button onclick="RSA();">Apply RSA</button></td>
</tr>
</table>
</center>
</body>
<script type="text/javascript">
function RSA() {
var gcd, p, q, no, n, t, e, i, x;
gcd = function (a, b) { return (!b) ? a : gcd(b, a % b); };
p = document.getElementById('p').value;
q = document.getElementById('q').value;
no = document.getElementById('msg').value;
n = p * q;
t = (p - 1) * (q - 1);

for (e = 2; e < t; e++) {
if (gcd(e, t) == 1) {
break;
}
}


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

```
x = 1 + i * t
if (x % e == 0) {
d = x / e;
break;
}
}

ctt = Math.pow(no, e).toFixed(0);
ct = ctt % n;

dtt = Math.pow(ct, d).toFixed(0);
dt = dtt % n;

document.getElementById('publickey').innerHTML = n;
document.getElementById('exponent').innerHTML = e;
document.getElementById('privatekey').innerHTML = d;
document.getElementById('ciphertext').innerHTML = ct;
}
</script>
</html>
```

OUTPUT:

RSA Algorithm

Implemented Using HTML & Javascript

Enter First Prime Number:	53
Enter Second Prime Number:	59
Enter the Message(cipher text): [A=1, B=2,...]	89
Public Key:	3127
Exponent:	3
Private Key:	2011
Cipher Text:	1394

Apply RSA

Ex. No : 6 Date :	Diffie-Hellman key exchange algorithm

PROGRAM:

DiffieHellman.java

```
class DiffieHellman {
public static void main(String args[]) {
int p = 23; /* publicly known (prime number) */
int g = 5; /* publicly known (primitive root) */
int x = 4; /* only Alice knows this secret */
int y = 3; /* only Bob knows this secret */
double aliceSends = (Math.pow(g, x)) % p;
double bobComputes = (Math.pow(aliceSends, y)) % p;
double bobSends = (Math.pow(g, y)) % p;
double aliceComputes = (Math.pow(bobSends, x)) % p;
double sharedSecret = (Math.pow(g, (x * y))) % p;
System.out.println("simulation of Diffie-Hellman key exchange algorithm\n");
System.out.println("Alice Sends : " + aliceSends);
System.out.println("Bob Computes : " + bobComputes);
System.out.println("Bob Sends : " + bobSends);
System.out.println("Alice Computes : " + aliceComputes);
System.out.println("Shared Secret : " + sharedSecret);
/* shared secrets should match and equality is transitive */
if ((aliceComputes == sharedSecret) && (aliceComputes == bobComputes))
System.out.println("Success: Shared Secrets Matches! " + sharedSecret);
else
System.out.println("Error: Shared Secrets does not Match");
}
}
```

OUTPUT:

```
simulation of Diffie-Hellman key exchange algorithm
Alice Sends : 4.0
Bob Computes : 18.0
Bob Sends : 10.0
Alice Computes : 18.0
Shared Secret : 18.0
Success: Shared Secrets Matches! 18.0
```

Ex. No : 7 Date :	SHA-1 Algorithm

PROGRAM:

sha1.java

```
import java.security.*;
public class sha1 {
public static void main(String[] a) {
try {
MessageDigest md = MessageDigest.getInstance("SHA1");
System.out.println("Message digest object info:\n----------------");
System.out.println("Algorithm=" + md.getAlgorithm());
System.out.println("Provider=" + md.getProvider());
System.out.println("ToString=" + md.toString());
String input = "";
md.update(input.getBytes());
byte[] output = md.digest();
System.out.println();
System.out.println("SHA1(\"" + input + "\")=" + bytesToHex(output));
input = "abc";
md.update(input.getBytes());
output = md.digest();
System.out.println();
System.out.println("SHA1(\"" + input + "\")=" + bytesToHex(output));
input = "abcdefghijklmnopqrstuvwxyz";
md.update(input.getBytes());
output = md.digest();
System.out.println();
System.out.println("SHA1(\"" + input + "\")=" + bytesToHex(output));
System.out.println();
} catch (Exception e) {
System.out.println("Exception:" + e);
}
}

private static String bytesToHex(byte[] b) {
char hexDigit[] = { '0', '1', '2', '3', '4', '5', '6', '7', '8', '9', 'A', 'B', 'C', 'D', 'E', 'F' };
StringBuffer buf = new StringBuffer();

for (byte aB : b) {
```

```
buf.append(hexDigit[(aB >> 4) & 0x0f]);
buf.append(hexDigit[aB & 0x0f]);
}

return buf.toString();
}
}
```

OUTPUT:

```
Message digest object info:
----------------------------------
Algorithm=SHA1
Provider=SUN version 12
ToString=SHA1 Message Digest from SUN, <initialized>

SHA1("")=DA39A3EE5E6B4B0D3255BFEF95601890AFD80709

SHA1("abc")=A9993E364706816ABA3E25717850C26C9CD0D89D

SHA1("abcdefghijklmnopqrstuvwxyz")=32D10C7B8CF96570CA04CE37F2A19
D84240D3A89
```

Ex. No : 8 Date :	Digital Signature Standard

PROGRAM:

```
import java.security.KeyPair;
import java.security.KeyPairGenerator;
import java.security.PrivateKey;
import java.security.Signature;
import java.util.Scanner;
public class CreatingDigitalSignature {
public static void main(String args[]) throws Exception {
Scanner sc = new Scanner(System.in);
System.out.println("Enter some text");
String msg = sc.nextLine();
KeyPairGenerator keyPairGen = KeyPairGenerator.getInstance("DSA");
keyPairGen.initialize(2048);
KeyPair pair = keyPairGen.generateKeyPair();
PrivateKey privKey = pair.getPrivate();
Signature sign = Signature.getInstance("SHA256withDSA");
sign.initSign(privKey);
byte[] bytes = "msg".getBytes();
sign.update(bytes);
byte[] signature = sign.sign();
System.out.println("Digital signature for given text: "+new String(signature,
"UTF8"));
}
}
```

OUTPUT:

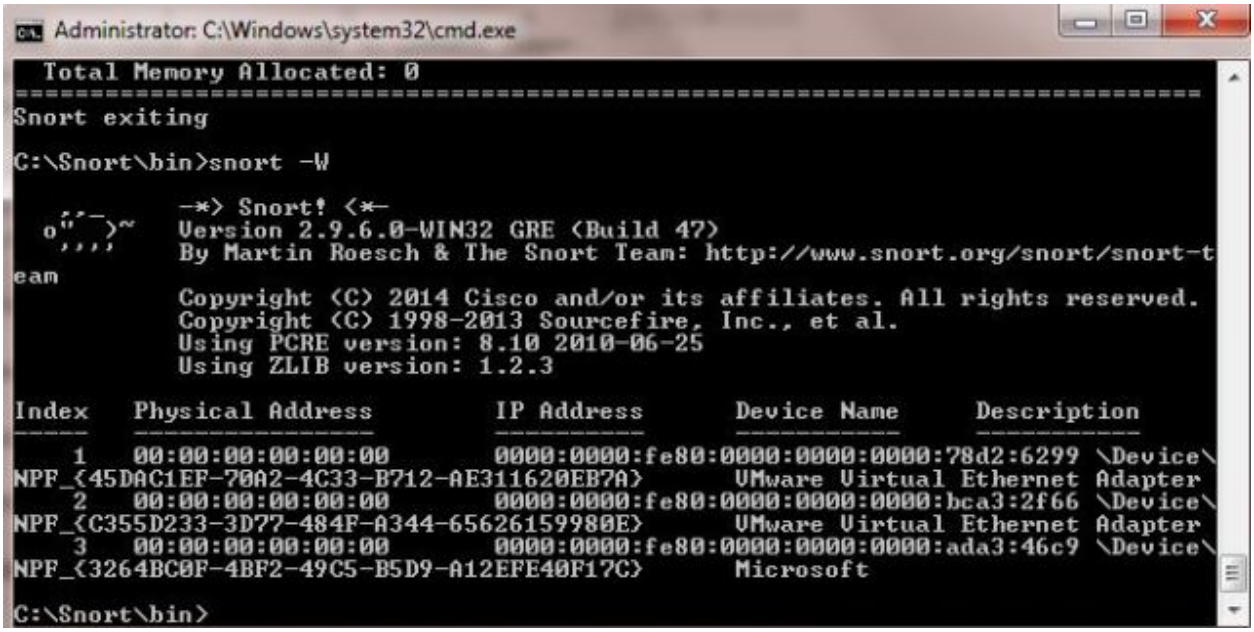
```
Enter some text
Hi how are you
Digital signature for given text: 0=@gRD???-?.???? /yGL?i??a!?
```

Ex. No : 9 Date :	Demonstration of Intrusion Detection System(IDS)

STEPS ON CONFIGURING AND INTRUSION DETECTION:

1. Download Snort from the Snort.org website. (http://www.snort.org/snort-downloads)
2. Download Rules(https://www.snort.org/snort-rules). You must register to get the rules. (You should download these often)
3. Double click on the .exe to install snort. This will install snort in the "C:\Snort" folder.It is important to have WinPcap (https://www.winpcap.org/install/) installed
4. Extract the Rules file. You will need WinRAR for the .gz file.
5. Copy all files from the "rules" folder of the extracted folder. Now paste the rules into *"C:\Snort\rules"* folder.
6. Copy "snort.conf" file from the "etc" folder of the extracted folder. You must paste it into "C:\Snort\etc" folder. Overwrite any existing file. Remember if you modify your snort.conf file and download a new file, you must modify it for Snort to work.
7. Open a command prompt (cmd.exe) and navigate to folder "C:\Snort\bin" folder. (at the Prompt, type cd\snort\bin)
8. To start (execute) snort in sniffer mode use following command:
snort -dev -i 3
-i indicates the interface number. You must pick the correct interface number. In my case, it is 3.
-dev is used to run snort to capture packets on your network.

To check the interface list, use following command:
snort -W



Finding an interface

You can tell which interface to use by looking at the Index number and finding Microsoft. As you can see in the above example, the other interfaces are for VMWare. My interface is 3.

9. To run snort in IDS mode, you will need to configure the file “snort.conf” according to your network environment.
10. To specify the network address that you want to protect in snort.conf file, look for the following line.
var HOME_NET 192.168.1.0/24 (You will normally see any here)
11. You may also want to set the addresses of DNS_SERVERS, if you have some on your network.

Example:
example snort
12. Change the RULE_PATH variable to the path of rules folder.
var RULE_PATH c:\snort\rules

13. Change the path of all library files with the name and path on your system. and you must change the path of snort_dynamicpreprocessorvariable.
C:\Snort\lib\snort_dynamiccpreprocessor
You need to do this to all library files in the "C:\Snort\lib" folder. The old path might be: "/usr/local/lib/…". you will need to replace that path with your system path. Using C:\Snort\lib
14. Change the path of the "dynamicengine" variable value in the "snort.conf" file..
Example:
dynamicengine C:\Snort\lib\snort_dynamicengine\sf_engine.dll

15 Add the paths for "include classification.config" and "include reference.config" files.
include c:\snort\etc\classification.config
include c:\snort\etc\reference.config
16. Remove the comment (#) on the line to allow ICMP rules, if it is commented with a #.
include $RULE_PATH/icmp.rules
17. You can also remove the comment of ICMP-info rules comment, if it is commented.
include $RULE_PATH/icmp-info.rules
18. To add log files to store alerts generated by snort, search for the "output log" test in snort.conf and add the following line:
output alert_fast: snort-alerts.ids
19. Comment (add a #) the whitelist $WHITE_LIST_PATH/white_list.rules and the blacklist

Change the nested_ip inner , \ to nested_ip inner #, \
20. Comment out (#) following lines:
#preprocessor normalize_ip4
#preprocessor normalize_tcp: ips ecn stream
#preprocessor normalize_icmp4
#preprocessor normalize_ip6
#preprocessor normalize_icmp6

21. Save the “snort.conf” file.
22. To start snort in IDS mode, run the following command:

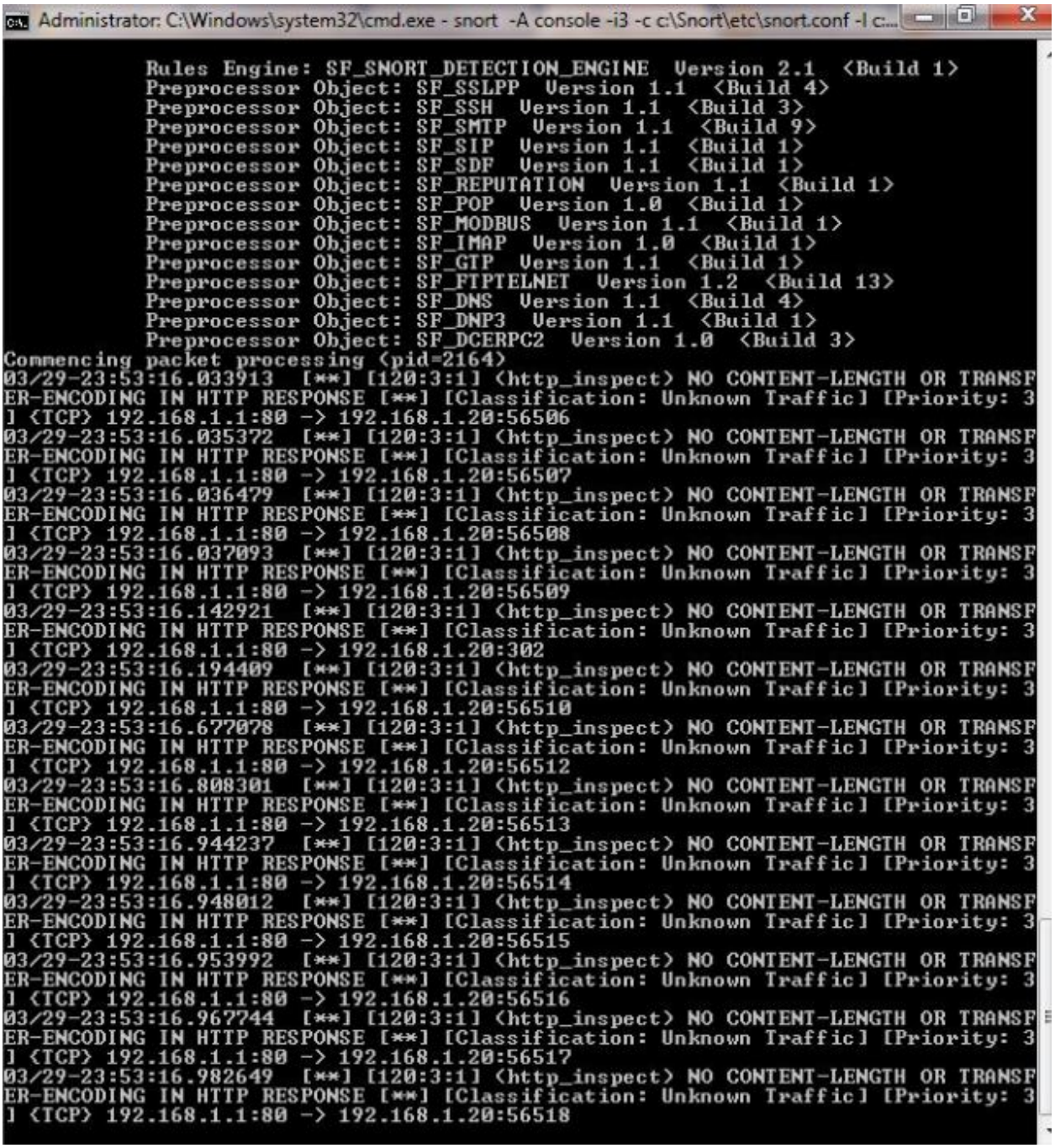
snort -c c:\snort\etc\snort.conf -l c:\snort\log -i 3
(Note: 3 is used for my interface card)

If a log is created, select the appropriate program to open it. You can use WordPard or NotePad++ to read the file.

To generate Log files in ASCII mode, you can use following command while running snort in IDS mode:
snort -A console -i3 -c c:\Snort\etc\snort.conf -l c:\Snort\log -K ascii

23. Scan the computer that is running snort from another computer by using PING or NMap (ZenMap).
After scanning or during the scan you can check the snort-alerts.ids file in the log folder to insure it is logging properly. You will see IP address folders appear.

Snort monitoring traffic –

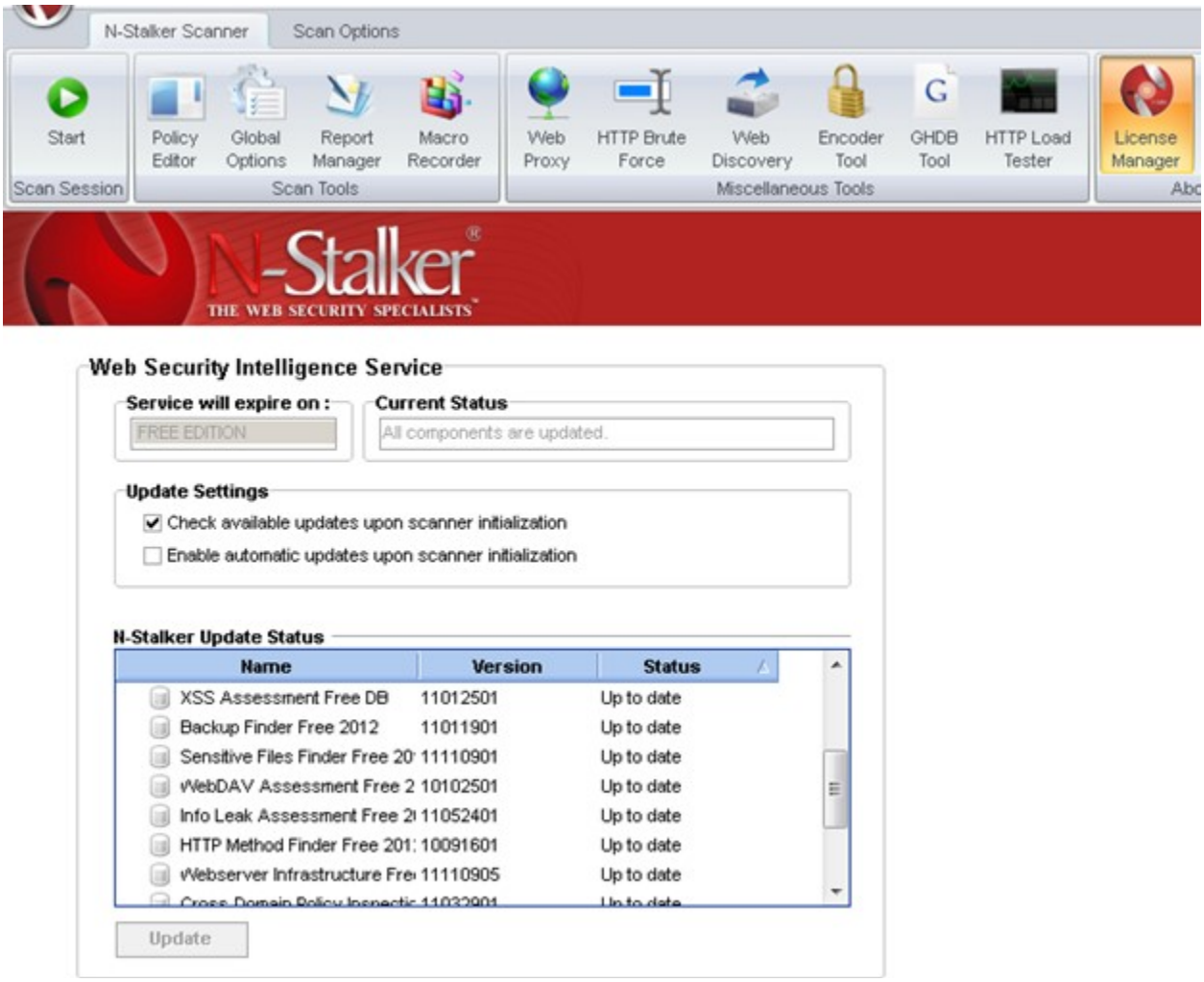


Ex. No : 10 Date :	Exploring N-Stalker, a Vulnerability Assessment Tool

EXPLORING N-STALKER:

- N-Stalker Web Application Security Scanner is a Web security assessment tool.
- It incorporates with a well-known N-Stealth HTTP Security Scanner and 35,000 Web attack signature database.
- This tool also comes in both free and paid version.
- Before scanning the target, go to "License Manager" tab, perform the update.
- Once update, you will note the status as up to date.
- You need to download and install N-Stalker from www.nstalker.com.

1. Start N-Stalker from a Windows computer. The program is installed under Start ⇨ Programs ⇨ N-Stalker ⇨ N-Stalker Free Edition.
2. Enter a host address or a range of addresses to scan.
3. Click Start Scan.
4. After the scan completes, the N-Stalker Report Manager will prompt
5. you to select a format for the resulting report as choose Generate HTML.
6. Review the HTML report for vulnerabilities.



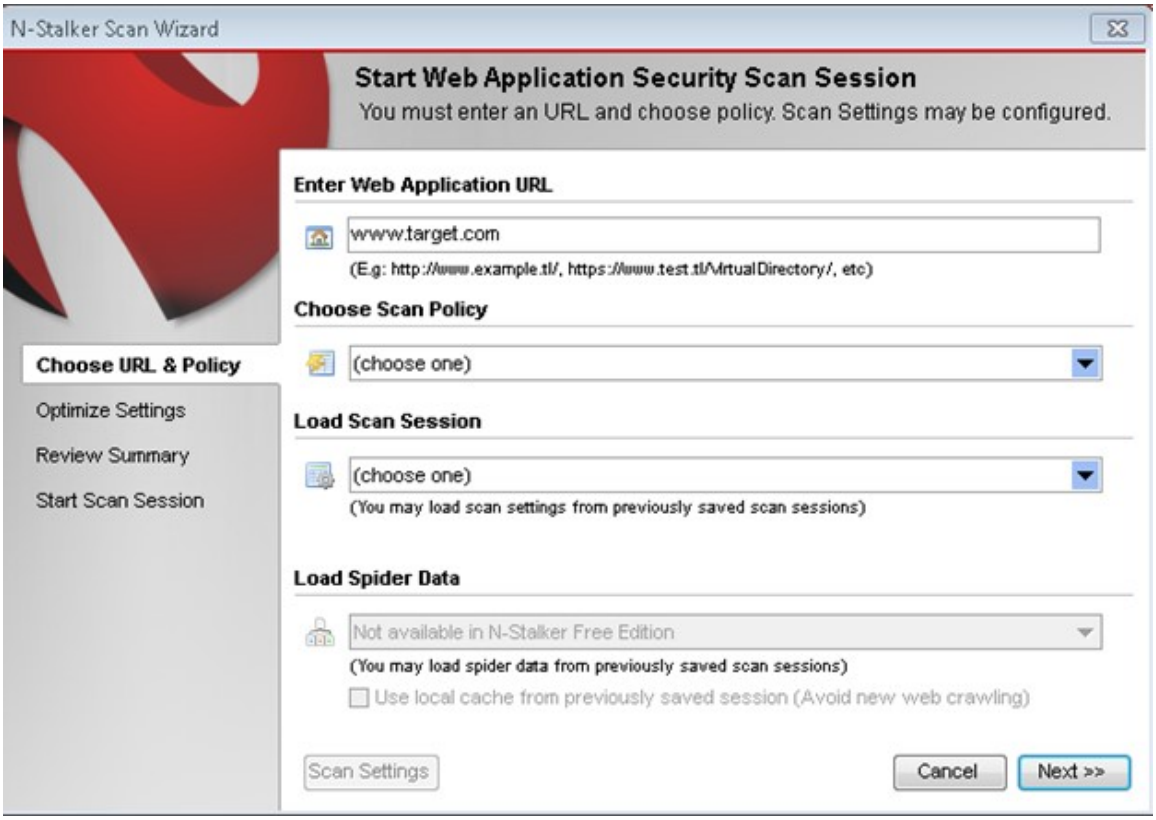
Now goto “Scan Session”, enter the target URL.

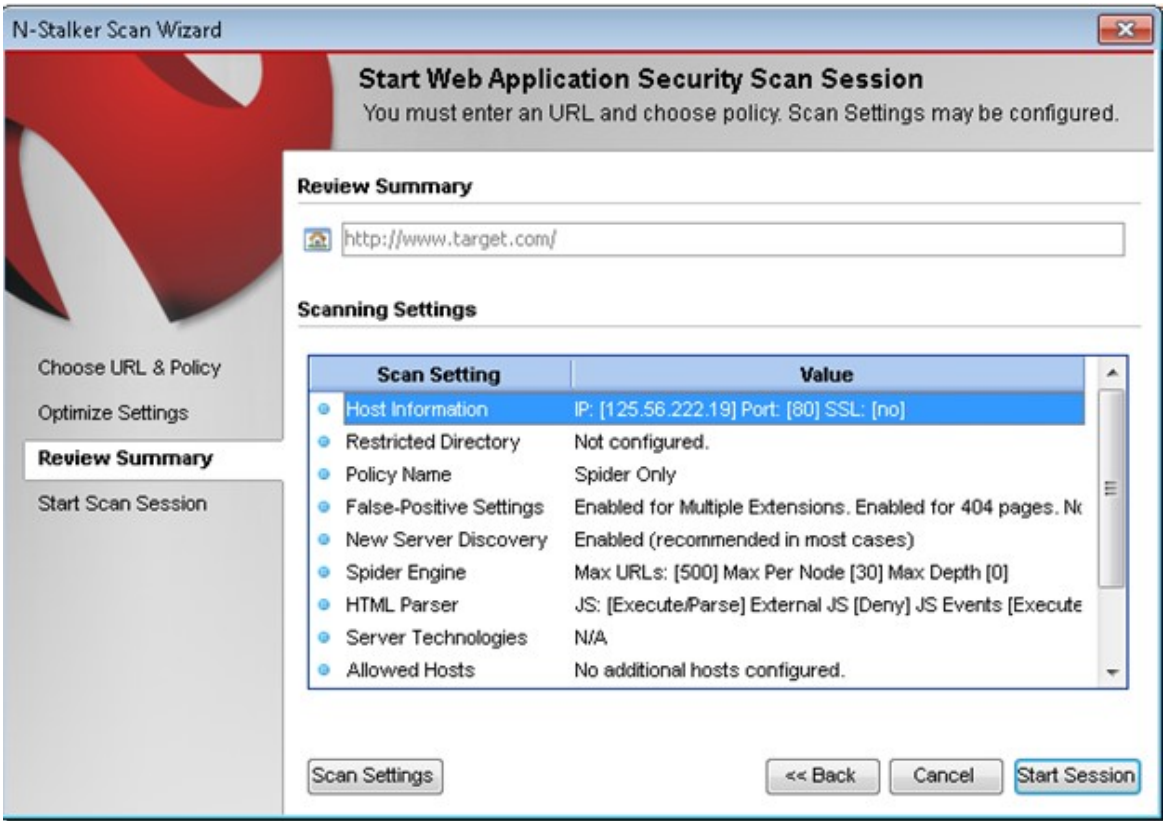
In scan policy, you can select from the four options,

- Manual test which will crawl the website and will be waiting for manual attacks.
- full xss assessment
- owasp policy
- Web server infrastructure analysis.

Once, the option has been selected, next step is “Optimize settings” which will crawl the whole website for further analysis.

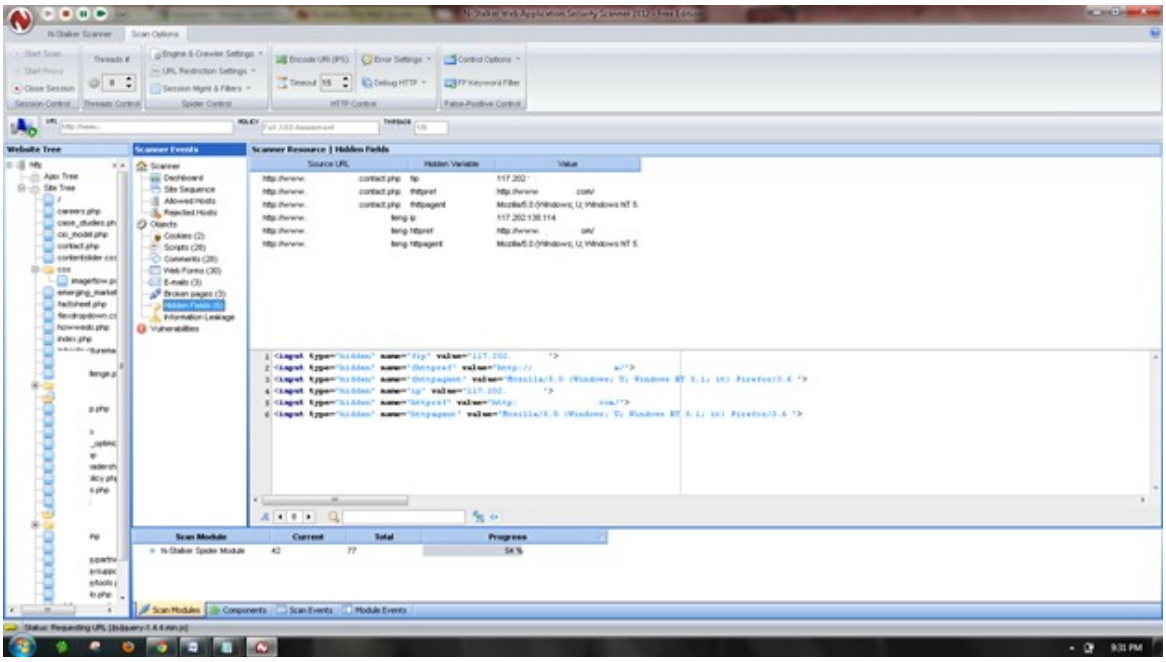
In review option, you can get all the information like host information, technologies used, policy name, etc.


N-Stalker Scan Wizard
Start Web Application Security Scan Session
You must enter an URL and choose policy. Scan Settings may be configured.
Enter Web Application URL
www.target.com
(E.g: http://www.example.tl/, https://www.test.tl/VirtualDirectory/, etc)
Choose Scan Policy
(choose one)
Load Scan Session
(choose one)
(You may load scan settings from previously saved scan sessions)
Load Spider Data
Not available in N-Stalker Free Edition
(You may load spider data from previously saved scan sessions)
Use local cache from previously saved session (Avoid new web crawling)
Choose URL & Policy
Optimize Settings
Review Summary
Start Scan Session
Scan Settings
Cancel
Next >>

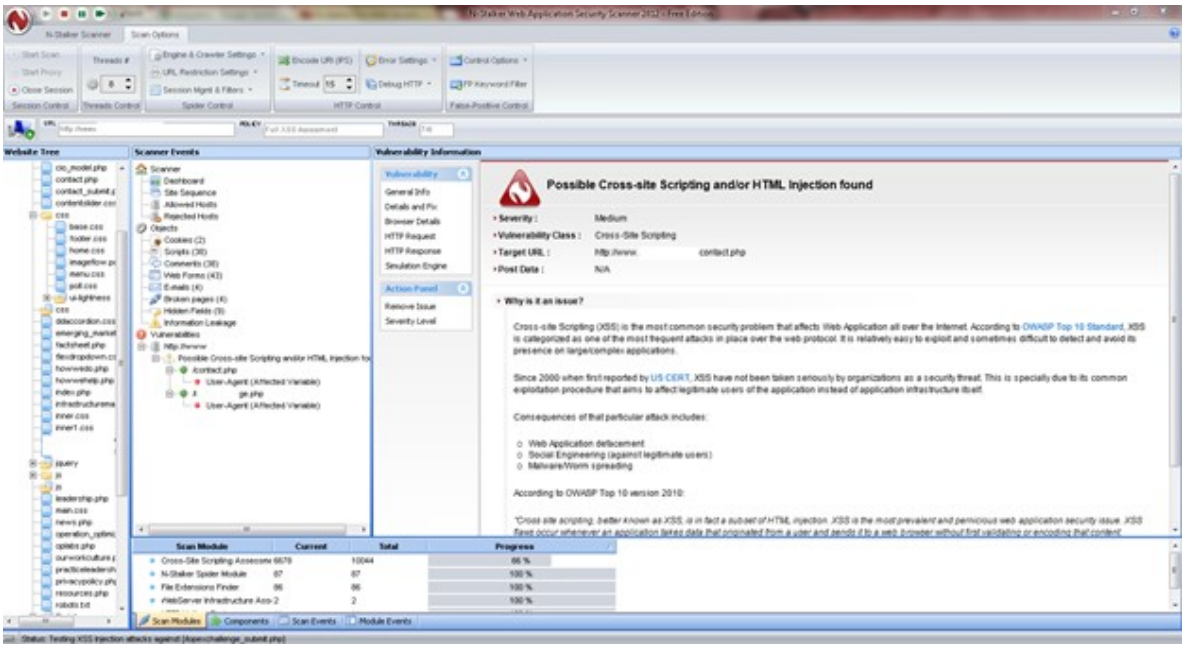

N-Stalker Scan Wizard
Start Web Application Security Scan Session
You must enter an URL and choose policy. Scan Settings may be configured.
Review Summary
http://www.target.com/
Scanning Settings
Scan Setting
Value
Host Information
IP: [125.56.222.19] Port: [80] SSL: [no]
Restricted Directory
Not configured.
Policy Name
Spider Only
False-Positive Settings
Enabled for Multiple Extensions. Enabled for 404 pages. Nc
New Server Discovery
Enabled (recommended in most cases)
Spider Engine
Max URLs: [500] Max Per Node [30] Max Depth [0]
HTML Parser
JS: [Execute/Parse] External JS [Deny] JS Events [Execute
Server Technologies
N/A
Allowed Hosts
No additional hosts configured.
Choose URL & Policy
Optimize Settings
Review Summary
Start Scan Session
Scan Settings
<< Back
Cancel
Start Session

Once done, start the session and start the scan.

The scanner will crawl the whole website and will show the scripts, broken pages, hidden fields, information leakage, web forms related information which helps to analyze further.



Once the scan is completed, the NStalker scanner will show details like severity level, vulnerability class, why is it an issue, the fix for the issue and the URL which is vulnerable to the particular vulnerability?


Possible Cross-site Scripting and/or HTML injection found
Severity : Medium
Vulnerability Class : Cross-Site Scripting
Target URL : http://www. contact.php
Post Data : N/A
Why is it an issue?
Cross-site Scripting (XSS) is the most common security problem that affects Web Application all over the Internet. According to OWASP Top 10 Standard, XSS is categorized as one of the most frequent attacks in place over the web protocol. It is relatively easy to exploit and sometimes difficult to detect and avoid its presence on large/complex applications.
Since 2000 when first reported by US CERT, XSS have not been taken seriously by organizations as a security threat. This is specially due to its common exploitation procedure that aims to affect legitimate users of the application instead of application infrastructure itself.
Consequences of that particular attack includes:
Web Application defacement
Social Engineering (against legitimate users)
Malware/Worm spreading
According to OWASP Top 10 version 2010:
"Cross site scripting, better known as XSS, is in fact a subset of HTML injection. XSS is the most prevalent and pernicious web application security issue. XSS flaws occur whenever an application takes data that originated from a user and sends it to a web browser without first validating or encoding that content.

Ex. No : 11(i) Date :	Defeating Malware - Building Trojans

TROJAN:

- In computing, a Trojan horse,or trojan, is any malware which misleads users of its true intent.

- Trojans are generally spread by some form of social engineering, for example where a user is duped into executing an email attachment disguised to appear not suspicious, (e.g., a routine form to be filled in), or by clicking on some fake advertisement on social media or anywhere else.

- Although their payload can be anything, many modern forms act as a backdoor, contacting a controller which can then have unauthorized access to the affected computer.
- Trojans may allow an attacker to access users' personal information such as banking information, passwords, or personal identity.

- ***Example:****Ransomware* attacks are often carried out using a *trojan*.

CODE:

Trojan.bat

```
@echo off
:x
start mspaint
start notepad
start cmd
start explorer
start control
start calc
goto x
```

OUTPUT

(MS-Paint, Notepad, Command Prompt, Explorer will open infinitely)

Ex. No : 11(ii) Date :	Defeating Malware - Rootkit hunter

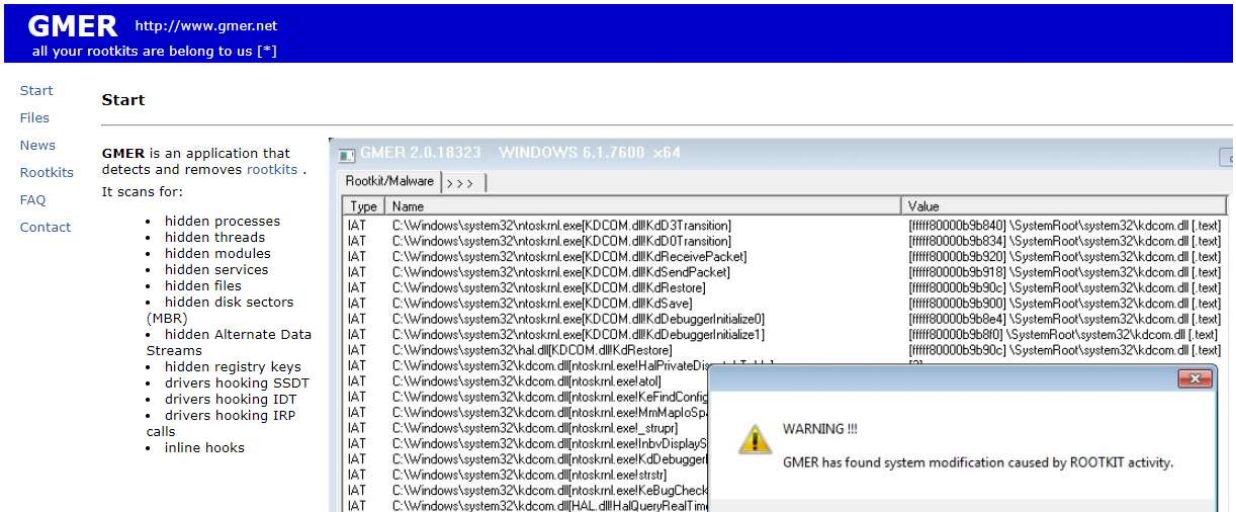
ROOTKIT HUNTER:

- rkhunter (Rootkit Hunter) is a Unix-based tool that scans for rootkits, backdoors and possible local exploits.
- It does this by comparing SHA-1 hashes of important files with known good ones in online databases, searching for default directories (of rootkits), wrong permissions, hidden files, suspicious strings in kernel modules, and special tests for Linux and FreeBSD.
- rkhunter is notable due to its inclusion in popular operating systems (Fedora, Debian, etc.)
- The tool has been written in Bourne shell, to allow for portability. It can run on almost all UNIX-derived systems.

GMER ROOTKIT TOOL:

- GMER is a software tool written by a Polish researcher Przemysław Gmerek, for detecting and removing rootkits.
- It runs on Microsoft Windows and has support for Windows NT, 2000, XP, Vista, 7, 8 and 10. With version 2.0.18327 full support for Windows x64 is added.

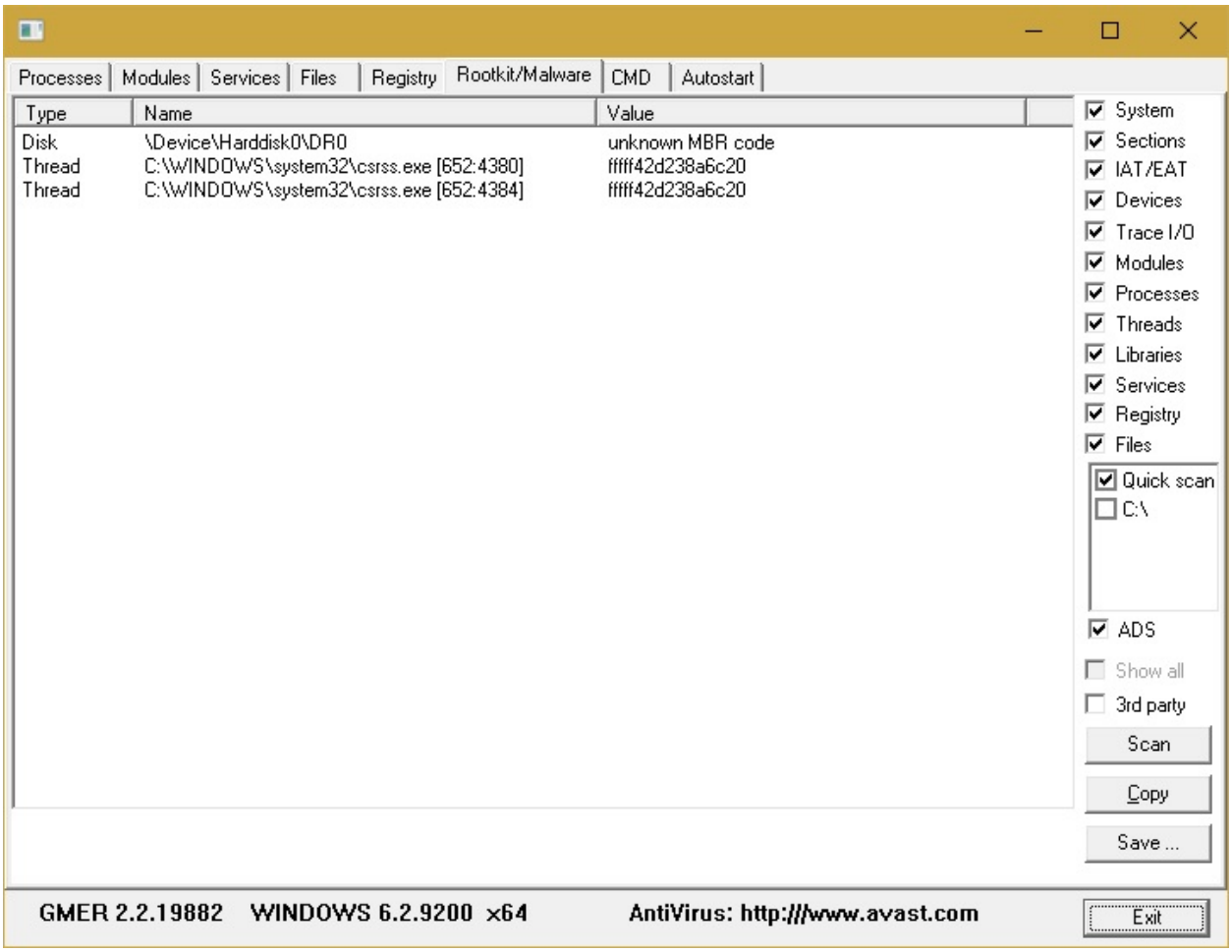
Step 1



Visit GMER's website (see Resources) and download the GMER executable.

Click the "Download EXE" button to download the program with a random file name, as some rootkits will close “gmer.exe” before you can open it.

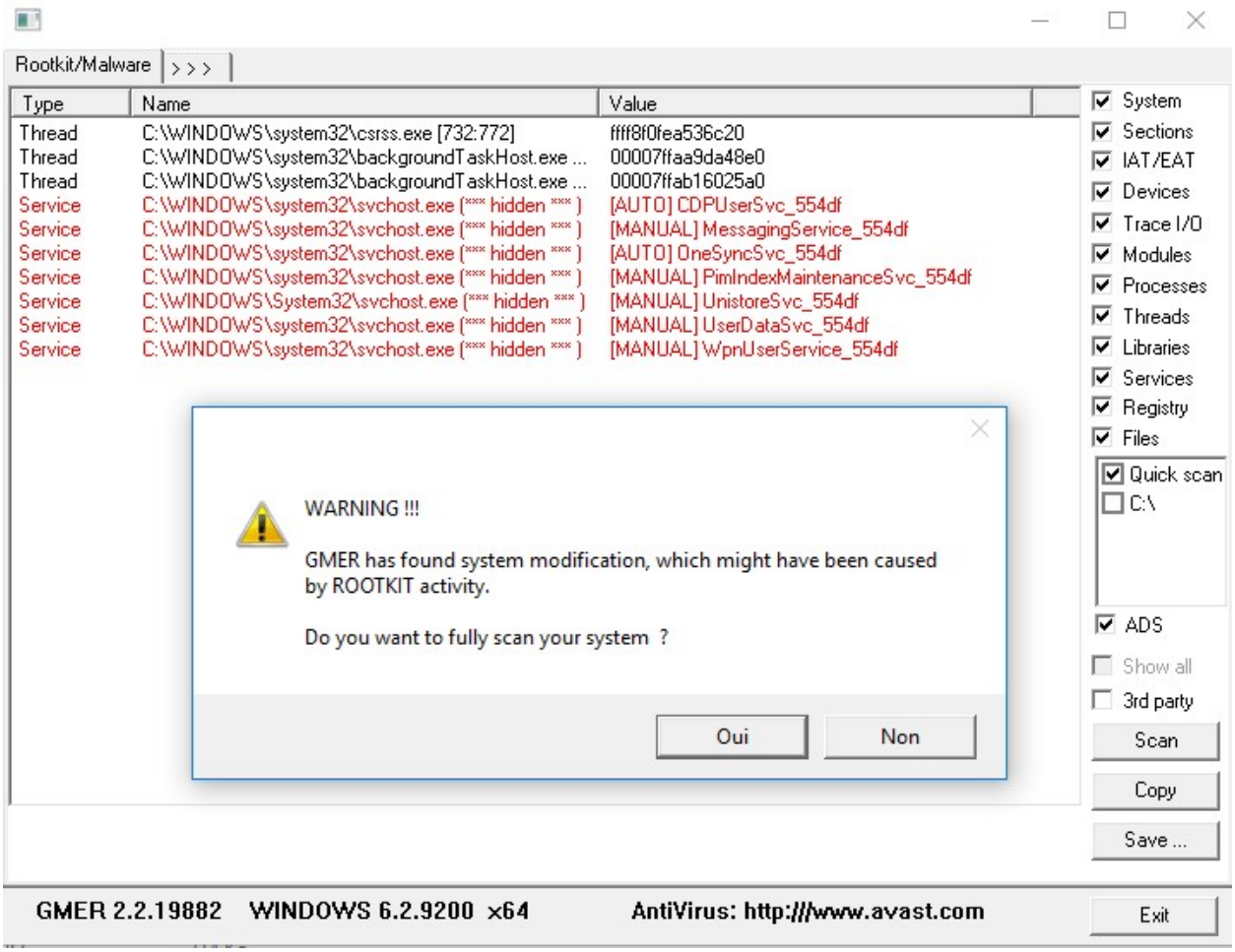
Step 2



Double-click the icon for the program.

Click the "Scan" button in the lower-right corner of the dialog box. Allow the program to scan your entire hard drive.

Step 3



When the program completes its scan, select any program or file listed in red. Right-click it and select "Delete."

If the red item is a service, it may be protected. Right-click the service and select "Disable." Reboot your computer and run the scan again, this time selecting "Delete" when that service is detected.

When your computer is free of Rootkits, close the program and restart your PC.