



# POORNIMA

## COLLEGE OF ENGINEERING

**Manual Ref. No.: 2021-22\_ODD\_7CS4-22\_Cyber Security LAB**

**Session 2021-2022**

**Branch: Computer Science & Engineering**

**Name of the Lab: Cyber Security LAB**

**Year: 4<sup>th</sup> Yr. /VII SEM**

**Lab Code: - 7CS4-22**

**Name of the Faculty: Gaurav Sharma**

**Name of the Technical Assistant: Mr. Jitendra Kumar**

**Verified & Checked by:**  
**Dr. Surendra Kumar Yadav**  
**Name of HOD**

**Approved by:**  
**Dr. Mahesh Bundeale**  
**Director, PCE**

# LAB RULES

## Responsibilities of Users

Users are expected to follow some fairly obvious rules of conduct:



### Always:

- Enter the lab on time and leave at proper time.
- Wait for the previous class to leave before the next class enters.
- Keep the bag outside in the respective racks.
- Utilize lab hours in the corresponding.
- Turn off the machine before leaving the lab unless a member of lab staff has specifically told you not to do so.
- Leave the labs at least as nice as you found them.
- If you notice a problem with a piece of equipment (e.g. a computer doesn't respond) or the room in general (e.g. cooling, heating, lighting) please report it to lab staff immediately. Do not attempt to fix the problem yourself.

**Never:**

- Don't abuse the equipment.
- Do not adjust the heat or air conditioners. If you feel the temperature is not properly set, inform lab staff; we will attempt to maintain a balance that is healthy for people and machines.
- Do not attempt to reboot a computer. Report problems to lab staff.
- Do not remove or modify any software or file without permission.
- Do not remove printers and machines from the network without being explicitly told to do so by lab staff.
- Don't monopolize equipment. If you're going to be away from your machine for more than 10 or 15 minutes, log out before leaving. This is both for the security of your account, and to ensure that others are able to use the lab resources while you are not.
- Don't use internet, internet chat of any kind in your regular lab schedule.
- Do not download or upload of MP3, JPG or MPEG files.
- No games are allowed in the lab sessions.
- No hardware including USB drives can be connected or disconnected in the labs without prior permission of the lab in-charge.
- No food or drink is allowed in the lab or near any of the equipment. Aside from the fact that it leaves a mess and attracts pests, spilling anything on a keyboard or other piece of computer equipment could cause permanent, irreparable, and costly damage. (and in fact *has*) If you need to eat or drink, take a break and do so in the canteen.
- Don't bring any external material in the lab, except your lab record, copy and books.
- Don't bring the mobile phones in the lab. If necessary then keep them in silence mode.
- Please be considerate of those around you, especially in terms of noise level. While labs are a natural place for conversations of all types, kindly keep the volume turned down.

If you are having problems or questions, please go to either the faculty, lab in-charge or the lab supporting staff. They will help you. We need your full support and cooperation for smooth functioning of the lab.

# INSTRUCTIONS

## Before entering in the lab

- All the students are supposed to prepare the theory regarding the next experiment/ Program.
- Students are supposed to bring their lab records as per their lab schedule.
- Previous experiment/program should be written in the lab record.
- If applicable trace paper/graph paper must be pasted in lab record with proper labeling.
- All the students must follow the instructions, failing which he/she may not be allowed in the lab.

## While working in the lab

- Adhere to experimental schedule as instructed by the lab in-charge/faculty.
- Get the previously performed experiment/ program signed by the faculty/ lab in charge.
- Get the output of current experiment/program checked by the faculty/ lab in charge in the lab copy.
- Each student should work on his/her assigned computer at each turn of the lab.
- Take responsibility of valuable accessories.

## SYLLABUS

**Practical Hrs: 2 Hrs/ week**

**Maximum Marks = 100**

For the instructor: Assign any two projects two a group of exactly two students covering all of the experiments from given experiment list.

Each group is required to prepare the following documents for projects assigned to them and develop the software using software engineering methodology.

1. Problem Analysis and Project Planning Thorough study of the problem- identify project scope, infrastructure.
2. Software Requirement Analysis- Describe the individual Phases/modules of the project deliverables.
3. Data Modeling Use work products – data dictionary, use case diagrams and activity diagrams, build and test lass diagrams, sequence diagrams and add interface to class diagrams.
4. Software Developments and Debugging. 5. Software Testing – Prepare test plan, perform validation testing coverage analysis, memory leaks, develop test case hierarchy, Site check and site monitor.
6. Describe: Relevance of CASE tools, high – end and low – end CASE tools, automated support for data dictionaries, DFD, ER diagrams.

Sr. No.	List of Experiments	Software Recommended:
1	Implement the following Substitution & Transposition Techniques concepts	C, C++ , Java, or equivalent Compiler GnuPG, Snort, Wireshark..
2	Implement the Diffie-Hellman Key Exchange mechanism.	
3	Implement the dictionary, brut force attack.	
4	Installation of Wire shark,	
5	Installation of rootkits	
6	Perform an Experiment to Sniff Traffic using ARP Poisoning	
7	Demonstrate intrusion detection system using any tool (snort or any other s/w).	
8	Demonstrate how to provide secure data storage, secure data transmission and for creating digital signatures	

## EVALUATION SCHEME

**(To be verified from RTU syllabus)**

Name of Exam	Conducted By	Experiment Marks	Viva Marks
I Mid Term	PCE	30	10
II Mid Term	PCE	30	10
End Term	RTU	30	10

Name of Exam	Conducted By	Performance Marks	Attendance Marks
Sessional	PCE	30	10

### Distribution of Lab Record Marks per Experiment

Attendance	Record	Performance	Total
2	3	5	10

# INDEX

Sr. No.	Topic	Page Number
1	Lab Plan	
2	Lab Objective and Outcome	
3	List of Experiments (as per RTU)	
3	List of Experiments (with variations)	
4	Experiment No 1	
5	Experiment No 2	
6	Experiment No 3	
7	Experiment No 4	
8	Experiment No 5	
9	Experiment No 6	
10	Experiment No 7	
11	Experiment No 8	
12	References	
13	Viva Questions	

## LAB PLAN

Total number of experiment 8

Total number of turns required 8

### Number of turns required for

Experiment Number	Turns	Scheduled Day
Exp. 1	1	Day 1
Exp. 2	1	Day 2
Exp. 3	1	Day 3
Exp. 4	1	Day 4
Exp. 5	1	Day 5
Exp. 6	1	Day 6
Exp. 7	1	Day 7
Exp. 8	1	Day 8
Project	2	Day 9

### Distribution of lab hours

Attendance 05 minutes


Explanation of features of language 20 minutes

Explanation of experiment 60 minutes

Performance of experiment 80 minutes

Viva / Quiz / Queries 15 min

Department of Computer Science & Engineering

  
**Dr. Mahesh Bunde**  
B.E., M.E., Ph.D.  
Director  
Poonima College of Engineering  
ISI-0, RICO Institutional Area  
Sikapura, JAIPUR



## Lab Objective and Outcome

Objectives of this lab is as follows:


- To study the various algorithms of cryptography.
- To learn the wire shark, root kit and Snort tool .

To learn the fundamental activities to develop a cryptographic algorithms, and networking tools.

### Outcomes:-

- The student should be able to implement the cipher techniques.
- Develop the various security algorithms.
- Use different open source tools for network security and analysis
- To learn & use the new tools and technologies used for designing some security principles.
- Analyse and resolve security issues in networks and computer systems to secure an IT infrastructure.

Department of Computer Science & Engineering

  
**Dr. Mahesh Bunde**  
B.E., M.E., Ph.D.  
Director  
Poonima College of Engineering  
ISI-0, RICO Institutional Area  
Sikapura, JAIPUR


## List of Experiments (As per RTU)

- 1) Implement the following Substitution & Transposition Techniques concepts:
  - a) Caesar Cipher b) Rail fence row & Column Transformation
- 2) Implement the Diffie-Hellman Key Exchange mechanism using HTML and JavaScript. Consider the end user as one of the parties (Alice) and the JavaScript application as other party (bob).
- 3) Implement the following Attack: a) Dictionary Attack b) Brute Force Attack
- 4) Installation of Wire shark, tcpdump, etc and observe data transferred in client server communication using UDP/TCP and identify the UDP/TCP datagram.
- 5) Installation of rootkits and study about the variety of options.
- 6) Perform an Experiment to Sniff Traffic using ARP Poisoning.
- 7) Demonstrate intrusion detection system using any tool (snort or any other s/w).
- 8) Demonstrate how to provide secure data storage, secure data transmission and for creating digital signatures

## List of Experiments (with variations)

- 1) Implement the following Substitution & Transposition Techniques concepts:
  - a) Caesar Cipher b) Rail fence row & Column Transformation
- 2) Implement the Diffie-Hellman Key Exchange mechanism using HTML and JavaScript. Consider the end user as one of the parties (Alice) and the JavaScript application as other party (bob).
- 3) Implement the following Attack: a) Dictionary Attack b) Brute Force Attack
- 4) Installation of Wire shark, tcpdump, etc and observe data transferred in client server communication using UDP/TCP and identify the UDP/TCP datagram.
- 5) Installation of rootkits and study about the variety of options.
- 6) Perform an Experiment to Sniff Traffic using ARP Poisoning.
- 7) Demonstrate intrusion detection system using any tool (snort or any other s/w).
- 8) Demonstrate how to provide secure data storage, secure data transmission and for creating digital signatures

Department of Computer Science & Engineering

  
**Dr. Mahesh Bunde**  
B.E., M.E., Ph.D.  
Director  
Poonima College of Engineering  
ISI-0, RICO Institutional Area  
Sikapura, JAIPUR

# Experiments

## LAB 1:

### Lab 1

1) Implement the following Substitution & Transposition Techniques concepts:

a) Caesar Cipher    b) Rail fence row & Column Transformation

a) Caesar Cipher

// A C++ program to illustrate Caesar Cipher Technique

```
#include <iostream>
```

```
using namespace std;
```

```
// This function receives text and shift and
```

```
// returns the encrypted text
```

```
string encrypt(string text, int s)
```

```
{
```

```
    string result = "";
```

```
    // traverse text
```

```
    for (int i=0;i<text.length();i++)
```

```
    {
```

```
        // apply transformation to each character
```

```
        // Encrypt Uppercase letters
```

```
        if (isupper(text[i]))
```

```
            result += char(int(text[i]+s-65)%26 +65);
```

```
        // Encrypt Lowercase letters
```

```
        else
```

```
            result += char(int(text[i]+s-97)%26 +97);
```

```
    }
```

```
    // Return the resulting string
```

Department of Computer Science & Engineering

  
**Dr. Mahesh Bunde**  
B.E., M.E., Ph.D.  
Director

Peernima College of Engineering  
ISI-0, RIIICO Institutional Area  
Sitapura, JAIPUR

```
        return result;
    }

// Driver program to test the above function
int main()
{
    string text="ATTACKATONCE";
    int s = 4;
    cout << "Text : " << text;
    cout << "\nShift: " << s;
    cout << "\nCipher: " << encrypt(text, s);
    return 0;
}

import java.util.*;
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 ));
                }
            }
        }
    }
}

b)    Rail fence

// C++ program to illustrate Rail Fence Cipher
// Encryption and Decryption
#include <bits/stdc++.h>
using namespace std;

// function to encrypt a message
string encryptRailFence(string text, int key)
{
    // create the matrix to cipher plain text
    // key = rows , length(text) = columns
```

```
char rail[key][(text.length())];

// filling the rail matrix to distinguish filled
// spaces from blank ones
for (int i=0; i < key; i++)
    for (int j = 0; j < text.length(); j++)
        rail[i][j] = '\n';

// to find the direction
bool dir_down = false;
int row = 0, col = 0;

for (int i=0; i < text.length(); i++)
{
    // check the direction of flow
    // reverse the direction if we've just
    // filled the top or bottom rail
    if (row == 0 || row == key-1)
        dir_down = !dir_down;

    // fill the corresponding alphabet
    rail[row][col++] = text[i];

    // find the next row using direction flag
    dir_down?row++ : row--;
}

//now we can construct the cipher using the rail matrix
string result;
for (int i=0; i < key; i++)
    for (int j=0; j < text.length(); j++)
        if (rail[i][j]!='\n')
            result.push_back(rail[i][j]);

return result;
}

// This function receives cipher-text and key
// and returns the original text after decryption
string decryptRailFence(string cipher, int key)
{
    // create the matrix to cipher plain text
    // key = rows , length(text) = columns
```

```
char rail[key][cipher.length()];

// filling the rail matrix to distinguish filled
// spaces from blank ones
for (int i=0; i < key; i++)
    for (int j=0; j < cipher.length(); j++)
        rail[i][j] = '\n';

// to find the direction
bool dir_down;

int row = 0, col = 0;

// mark the places with '*'
for (int i=0; i < cipher.length(); i++)
{
    // check the direction of flow
    if (row == 0)
        dir_down = true;
    if (row == key-1)
        dir_down = false;

    // place the marker
    rail[row][col++] = '*';

    // find the next row using direction flag
    dir_down?row++ : row--;
}

// now we can construct the fill the rail matrix
int index = 0;
for (int i=0; i<key; i++)
    for (int j=0; j<cipher.length(); j++)
        if (rail[i][j] == '*' && index<cipher.length())
            rail[i][j] = cipher[index++];

// now read the matrix in zig-zag manner to construct
// the resultant text
string result;

row = 0, col = 0;
for (int i=0; i< cipher.length(); i++)
```

```
{
    // check the direction of flow
    if (row == 0)
        dir_down = true;
    if (row == key-1)
        dir_down = false;

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

    // find the next row using direction flag
    dir_down?row++: row--;
}
return result;
}

//driver program to check the above functions
int main()
{
    cout << encryptRailFence("attack at once", 2) << endl;
    cout << encryptRailFence("GeeksforGeeks ", 3) << endl;
    cout << encryptRailFence("defend the east wall", 3) << endl;

    //Now decryption of the same cipher-text
    cout << decryptRailFence("GsGsekre eoe",3) << endl;
    cout << decryptRailFence("atc toctaka ne",2) << endl;
    cout << decryptRailFence("dnhaweetees alf tl",3) << endl;

    return 0;
}
```



## LAB 2

Implement the Diffie-Hellman Key Exchange mechanism using HTML and JavaScript. Consider the end user as one of the parties (Alice) and the JavaScript application as other party (bob)

```
import java.util.*;

// create class DiffieHellmanAlgorithmExample to calculate the key for two persons

class DiffieHellmanAlgorithmExample {

    // main() method start

    public static void main(String[] args)

    {

        long P, G, x, a, y, b, ka, kb;

        // create Scanner class object to take input from user

        Scanner sc = new Scanner(System.in);

        System.out.println("Both the users should be agreed upon the public keys G and P");

        // take inputs for public keys from the user

        System.out.println("Enter value for public key G:");

        G = sc.nextLong();

        System.out.println("Enter value for public key P:");

        P = sc.nextLong();

        // get input from user for private keys a and b selected by User1 and User2

        System.out.println("Enter value for private key a selected by user1:");
```

Department of Computer Science & Engineering

```
a = sc.nextLong();

System.out.println("Enter value for private key b selected by user2:");

b = sc.nextLong();


// call calculatePower() method to generate x and y keys

x = calculatePower(G, a, P);

y = calculatePower(G, b, P);


// call calculatePower() method to generate ka and kb secret keys after the exchange of x and y
keys

// calculate secret key for User1

ka = calculatePower(y, a, P);

// calculate secret key for User2

kb = calculatePower(x, b, P);

// print secret keys of user1 and user2

System.out.println("Secret key for User1 is:" + ka);

System.out.println("Secret key for User2 is:" + kb);

}


// create calculatePower() method to find the value of  $x^y \bmod P$ 

private static long calculatePower(long x, long y, long P)

{
```

```
long result = 0;

if (y == 1){

    return x;

}

else{

    result = ((long)Math.pow(x, y)) % P;

    return result;

}

}
```

### Lab 3

Implement the following Attack: a) Dictionary Attack b) Brute Force Attack

```
a) import java.security.*;
import java.io.*;
import java.util.*;
import java.lang.StringBuilder;
```

Department of Computer Science & Engineering

```
import javax.xml.bind.DatatypeConverter;
public class DictionaryAttack {
//Extracted from http://www.jmdoudoux.fr/java/dej/chap-jca.htm
public 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 (int j=0; j<b.length; j++) {
        buf.append(hexDigit[(b[j] >> 4) & 0x0f]);
        buf.append(hexDigit[b[j] & 0x0f]);
    }
    return buf.toString();
}
//This method takes a string, computes its SHA-1 hash,
//and converts it into HEX using the bytesToHex method
public static String stringToSha1(String input) throws Exception {
//Setup a MessageDigest for SHA1
MessageDigest md = MessageDigest.getInstance("SHA1");
md.reset();
//Setup the MessageDigest with our input string
md.update(input.getBytes("UTF-8"));
//Convert the string's digest to HEX
String sha1 = bytesToHex(md.digest());
return sha1;
}
//This method takes a byte array holding a salt and a string input
//and returns the concatenated salt || input in byte array format
public static byte[] concatenate_salt_with_string(byte[] salt, String input) throws
Exception {
//Convert input string to bytes
byte[] input_byte = input.getBytes("UTF-8");
//Create byte array sufficiently large
byte[] concatenated = new byte[salt.length + input_byte.length];
//Insert the salt first
System.arraycopy(salt, 0, concatenated, 0, salt.length);
//Insert the input string converted to bytes
System.arraycopy(input_byte, 0, concatenated, salt.length, input_byte.length);
//Return the concatenated salt and string in a byte array
return concatenated;
}
//This method takes a string, a salt, computes its salted SHA-1 hash,
//and converts it into HEX using the bytesToHex method
public static String stringToSha1_salt(byte[] salt, String input) throws Exception {
```

```
//Setup a MessageDigest for SHA1
MessageDigest md = MessageDigest.getInstance("SHA1");
md.reset();
//Use the concatenate_salt_with_string method to concatenate the salt with the
input
byte[] concatenated = concatenate_salt_with_string(salt, input);
//Setup the MessageDigest with our input string
md.update(concatenated);
//Convert the string's digest to HEX
String sha1 = bytesToHex(md.digest());
return sha1;
}
public static void main(String[] args) throws Exception {
//Notify the user the program is starting.
System.out.println("Let's get things started.");
//Load the provided password file into stream and buffer
File passwords_file = new File("/E:/Cyber Security Lab 7th sem/LAB
PROGRAMS/password.txt");
FileInputStream password_stream = new FileInputStream(passwords_file);
BufferedReader password_buffer = new BufferedReader(new
InputStreamReader(password_stream));
//Initialize 3 hashmaps, one for non-salted passwords, one for salted passwords,
//and one for the salts of salted passwords.
Map<String, String> non_salted_passwords = new HashMap<String, String>();
Map<String, String> salted_passwords = new HashMap<String, String>();
Map<String, String> salted_passwords_salts = new HashMap<String, String>();
//We parse the buffer to extract user account names and passwords
String password_file_line = null;
while ((password_file_line = password_buffer.readLine()) != null) {
String[] splited = password_file_line.split("\\s+");
//First case: password hashed with no salt
if(splited.length == 3){
non_salted_passwords.put(splited[0], splited[2]);
}
//Second case: password hashed with a salt
else{
salted_passwords.put(splited[0], splited[3]);
salted_passwords_salts.put(splited[0], splited[2]);
}
}
//We are done reading the password file, we can close its buffer
password_buffer.close();
//Load the provided Dictionary into stream and buffer
```

```
File fin = new
File("/Users/nicolas/Documents/eclipseworkspace/dictionaryattack/src/english.0");
FileInputStream fis = new FileInputStream(fin);
//Construct BufferedReader from InputStreamReader
BufferedReader br = new BufferedReader(new InputStreamReader(fis));
//We parse the buffer to test matches for hashed password,
//reversed passwords, non vowel passwords, and salted versions of password (if
required).
String line = null;
while ((line = br.readLine()) != null) {
//We first iterate through the non salted passwords
Iterator non_salted_passwords_it = non_salted_passwords.entrySet().iterator();
while (non_salted_passwords_it.hasNext()) {
//We extract the key,value pair from the HashTable entry
Map.Entry pair = (Map.Entry)non_salted_passwords_it.next();
String account_name = pair.getKey().toString();
String account_password_hash = pair.getValue().toString();
//We test if the password matches an unmodified dictionary entry
if(account_password_hash.equals(stringToSha1(line))) {
System.out.println(account_name + "'s password is '" + line + "'");
}
//We test if the password matches a reversed dictionary entry
String reversed_line = new StringBuilder(line).reverse().toString();
if(account_password_hash.equals(stringToSha1(reversed_line))) {
System.out.println(account_name + "'s password is '" + reversed_line + "'");
}
//We test if the password matches a dictionary entry without its vowels
String line_without_vowels = line.replaceAll("[AEIOUaeiou]", "");
if(account_password_hash.equals(stringToSha1(line_without_vowels))) {
System.out.println(account_name + "'s password is '" + line_without_vowels +
"'");
}
}
//We then iterate through the salted passwords
Iterator salted_passwords_it = salted_passwords.entrySet().iterator();
while (salted_passwords_it.hasNext()) {
//We extract the key,value pair from the HashTable entry
Map.Entry salted_pair = (Map.Entry)salted_passwords_it.next();
String account_name = salted_pair.getKey().toString();
String account_password_hash = salted_pair.getValue().toString();
//We extract the corresponding salt from the HashTable of salts
byte[] account_password_hash_salt =
DatatypeConverter.parseHexBinary(salted_passwords_salts.get(account_name));
```

```
//We test if the password matches an unmodified dictionary entry

if(account_password_hash.equals(stringToSha1_salted(account_password_hash_salt,line))) {
    System.out.println(account_name + "'s password is '" + line + "'");
}
//We test if the password matches a reversed dictionary entry
String reversed_line = new StringBuilder(line).reverse().toString();

if(account_password_hash.equals(stringToSha1_salted(account_password_hash_salt,reversed_line))) {
    System.out.println(account_name + "'s password is '" + reversed_line + "'");
}
//We test if the password matches a dictionary entry without its vowels
String line_without_vowels = line.replaceAll("[AEIOUaeiou]", "");

if(account_password_hash.equals(stringToSha1_salted(account_password_hash_salt,line_without_vowels))) {
    System.out.println(account_name + "'s password is '" + line_without_vowels + "'");
}
}
}
}
//We are done using the dictionary file, we can close its buffer
br.close();
//Notify the user our program is done running.
System.out.println("The program terminated.");
}
}
```

#### b) Brute Force Attack

```
import java.util.Scanner;
public class BruteForce {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int i, j, l, k = 97, key = 0, flag = 0, index = 0, keyVal;
        String pt;
        char[] ct1 = new char[10];
        char[] pt1 = new char[10];
        char temp;
        System.out.println("ENTER PLAIN TEXT");
        pt = sc.next();
        System.out.println("ENTER KEY VALUE :");
```

```
key = sc.nextInt();
for (i = 0; i < pt.length(); i++) {
    for (j = 0; j < 26; j++) {
        if (pt.charAt(i) == ' ') {
            flag = 0;
            break;
        }
        temp = (char) (j + k);
        if (pt.charAt(i) == temp) {
            flag = 1;
            index = j;
            break;
        }
    }
    if (flag == 1) {
        char c = (char) (((index + key) % 26) + 97);
        ct1[i] = c;
    }
}
System.out.println("ENCRYPTED DATA:");
for (i = 0; i < pt.length(); i++) {
    System.out.print(ct1[i]);
}
System.out.println("\n" + "DECRYPTION OF DATA USING BRUTE-FORCE ATTACK
:");
key = 1;
while (key <= 26) {
    for (i = 0; i < pt.length(); i++) {
        for (j = 0; j < 26; j++) {
            if (ct1[i] == ' ') {
                flag = 0;
                break;
            }
            temp = (char) (j + k);
            if (ct1[i] == temp) {
                flag = 1;
                index = j;
                break;
            }
        }
        keyVal = index - key;
        if (flag == 1 & keyVal > 0) {
            pt1[i] = (char) ((keyVal % 26) + 97);
        }
    }
    key++;
}
```



```
} else if (flag == 1) {  
pt1[i] = (char) ((26 + keyVal) + 97);  
}  
}  
System.out.print("\n" + "DECRYPTED DATA:");  
for (i = 0; i < pt.length(); i++) {  
System.out.print(pt1[i]);  
}  
key++;  
}  
}  
}
```

#### Lab 4

Installation of Wire shark, tcpdump, etc and observe data transferred in client server communication using UDP/TCP and identify the UDP/TCP datagram.

To install Wireshark:

1. Open Windows Explorer.
2. Select the Downloads folder.
3. Locate the version of Wireshark you downloaded in Activity 2. ...
4. If you see a User Account Control dialog box, select Yes to allow the program to make changes to this computer.
5. Select Next > to start the Setup Wizard.
6. Review the license agreement.

tcpdump installation steps:

1. Download the rpm package for tcpdump.
2. Log in to DSVA via SSH as DSVA user. The default password is “dsva”.

Department of Computer Science & Engineering

3. Switch to root user using this command: `$sudo -s`.
4. Upload the package to DSVA under path:/home/dsva. ...
5. Unpack the tar package: ...
6. Install the rpm packages:
  - 1) Installation of rootkits and study about the variety of options.
  - 2) Perform an Experiment to Sniff Traffic using ARP Poisoning.
  - 3) Demonstrate intrusion detection system using any tool (snort or any other s/w).
  - 4) Demonstrate how to provide secure data storage, secure data transmission and for creating digital signatures

## Lab 5

Installation of rootkits and study about the variety of options.

### PROCEDURE:

STEP-1: Download Rootkit Tool from GMER website [www.gmer.net](http://www.gmer.net).

STEP-2: This displays the Processes, Modules, Services, Files, Registry, RootKit / Malwares, Autostart, CMD of local host.

STEP-3: Select Processes menu and kill any unwanted process if any.

STEP-4: Modules menu displays the various system files like .sys, .dll

STEP-5: Services menu displays the complete services running with Autostart, Enable, Disable, System, Boot.

STEP-6: Files menu displays full files on Hard-Disk volumes.

STEP-7: Registry displays Hkey\_Current\_user and Hkey\_Local\_Machine.

STEP-8: Rootkits / Malwares scans the local drives selected.


Department of Computer Science & Engineering

STEP-9: Autostart displays the registry base Autostart applications.

STEP-10:CMD allows the user to interact with command line utilities or Registry.

---

Department of Computer Science & Engineering

  
**Dr. Mahesh Bunde**  
B.E., M.E., Ph.D.  
Director  
Poornima College of Engineering  
ISI-0, FIICO Institutional Area  
Sikapura, JAIPUR

## Lab 6

### Perform an Experiment to Sniff Traffic using ARP Poisoning.

For this exercise, you would need the following tools –

- VMware workstation
- Kali Linux or Linux Operating system
- Ettercap Tool
- LAN connection

Note – This attack is possible in wired and wireless networks. You can perform this attack in local LAN.

Step 1 – Install the VMware workstation and install the Kali Linux operating system.

Step 2 – Login into the Kali Linux using username pass “root, toor”.

Step 3 – Make sure you are connected to local LAN and check the IP address by typing the command `ifconfig` in the terminal.

Step 4 – Open up the terminal and type “Ettercap –G” to start the graphical version of Ettercap.

Step 5 – Now click the tab “sniff” in the menu bar and select “unified sniffing” and click OK to select the interface. We are going to use “eth0” which means Ethernet connection.

Step 6 – Now click the “hosts” tab in the menu bar and click “scan for hosts”. It will start scanning the whole network for the alive hosts.

Step 7 – Next, click the “hosts” tab and select “hosts list” to see the number of hosts available in

the network. This list also includes the default gateway address. We have to be careful when we select the targets.

Step 8 – Now we have to choose the targets. In MITM, our target is the host machine, and the route will be the router address to forward the traffic. In an MITM attack, the attacker intercepts the network and sniffs the packets. So, we will add the victim as “target 1” and the router address as “target 2.”

Department of Computer Science & Engineering

In VMware environment, the default gateway will always end with “2” because “1” is assigned to the physical machine.

Step 9 – In this scenario, our target is “192.168.121.129” and the router is “192.168.121.2”. So we will add target 1 as victim IP and target 2 as router IP.

Step 10 – Now click on “MITM” and click “ARP poisoning”. Thereafter, check the option “Sniff remote connections” and click OK.

Step 11 – Click “start” and select “start sniffing”. This will start ARP poisoning in the network which means we have enabled our network card in “promiscuous mode” and now the local traffic can be sniffed.

Note – We have allowed only HTTP sniffing with Ettercap, so don’t expect HTTPS packets to be sniffed with this process.

Step 12 – Now it’s time to see the results; if our victim logged into some websites. You can see the results in the toolbar of Ettercap


## Lab 7

Demonstrate intrusion detection system using any tool (snort or any other s/w).

Installing Snort on Windows 10 A Step By Step Process:

1. For Windows 10 64 bit supported SNORT’s executable file can be downloaded from <https://www.snort.org/downloads>.
2. Open the downloaded snort executable file.
3. Click On ‘I Agree’ on the license agreement.
4. Choose components of Snort to be installed.

Department of Computer Science & Engineering

  
**Dr. Mahesh Bunde**  
B.E., M.E., Ph.D.  
Director  
Poornima College of Engineering  
ISI-0, FIICO Institutional Area  
Sikapura, JAIPUR

5. Click “Next” and then choose install location for snort preferably a separate folder in Windows C Drive.
6. Click “Next” Installation process starts and then it completes
7. When you click “ Close” you are prompted with this dialogue box:  
Installing Npcap is required by snort for proper functioning.
9. Npcap for Windows 10 can be downloaded from <https://nmap.org/npcap/>
10. Opening Npcap setup file, Click on ‘I Agree’ To license agreement
11. Now we proceed to choose which components of Npcap are to be installed and then clicking on “Install”.
12. Installation process starts and completes. Clicking on “Next”.
13. Now the window for installation of Npcap shows it has been installed. Clicking “Finish”.
14. After installing Snort and Npcap enter these commands in windows 10 Command prompt to check snorts working.
15. Snort successfully install.

#### Configuring Snort 2.9.17 on Windows 10:

After installing Snort on Windows 10, Another important step to get started with Snort is configuring it on Windows 10.

Note: The italicized portion with a left hand side border states commands which were pre-written

in the configuration file of Snort so we need to make changes according to the commands mentioned

in the images, to be precise we need to enter configuration commands as shown in the images to configure snort.

1. Go to <https://www.snort.org/downloads/#rule-downloads> and download latest snort rule file.

2. Extract 3 folders from the downloaded snortrules-snapshot-29170.tar folder into the Snorts corresponding folders in C drive.

Folders to be extracted are: rules , preproc\_rules , etc

- rules folder contains the rules files and the most important local.rules file. Which we will use to enter all our rules.
- etc folder contains all configuration files and the most important file is snort.conf file which we will use for configuration

3. Now open the snort.conf file through the notepad++ editor or any other text editor to edit configurations of snort to make it work like we want it to.

4. Setup the network addresses you are protecting  
ipvar HOME\_NET any

**Note: Mention your own host IP addresses that you want to protect.**

```
44 # Setup the network addresses you are protecting
45 ipvar HOME_NET 192.168.100.27/24
46
```

5. Setup the external network into anything that is not the home network. That is why ! is used in the command it denotes 'not'.

# Set up the external network addresses. Leave as "any" in most situationsipvar EXTERNAL\_NET any

```
47 # Set up the external network addresses. Leave as "any" in most situations
48 ipvar EXTERNAL_NET !$HOME_NET
49
```

6. Now we have to define the directory for our rules and preproc rules folder

# Path to your rules files (this can be a relative path)# Note for Windows users: You are advised to make this an absolute path,# such as: c:\snort\rulesvar RULE\_PATH ../rulesvar

SO\_RULE\_PATH

```
../so_rulesvar PREPROC_RULE_PATH ../preproc_rules
101 # Path to your rules files (this can be a relative path)
102 # Note for Windows users: You are advised to make this an absolute path,
103 # such as: c:\Snort\rules
104 var RULE_PATH c:\Snort\rules
105 # var SO_RULE_PATH ../so_rules
106 var PREPROC_RULE_PATH c:\Snort\preproc_rules
```

### Setting up path to our rules files and preproc rules folder in Snort

7. Now we have to setup our white list and black list path it will be in our snorts' rule folder

# If you are using reputation preprocessor set thesevar WHITE\_LIST\_PATH ../rulesvar  
BLACK\_LIST\_PATH ../rules

```
113 var WHITE_LIST_PATH c:\Snort\rules
114 var BLACK_LIST_PATH c:\Snort\rules
```

### : Setting up our White List and Black List files paths in Snort

8. Next we have to enable to log directory, so that we store logs in our log folder.

Uncomment this line and set absolute path to log directory

# Configure default log directory for snort to log to. For more information see snort -h command line options (-l)## config logdir:

```
186 config logdir: c:\Snort\log
```

### Setting up Log Directory Path in Snort

9. Now we will set the path to dynamic preprocessors and dynamicengine

# path to dynamic preprocessor libraries  
dynamic preprocessor directory/usr/local/lib/snort\_dynamicpreprocessor/

```
246 # path to dynamic preprocessor libraries
247 dynamicpreprocessor directory c:\Snort\lib\snort_dynamicpreprocessor
```



## Setting up path to dynamic preprocessors and dynamic engine in Snort

10. We will do same thing for dynamic preprocessor engine

```
# path to base preprocessor engine dynamicengine
/usr/local/lib/snort_dynamicengine/libsfs_engine.so

249 # path to base preprocessor engine
250 dynamicengine c:\Snort\lib\snort_dynamicengine\sfs_engine.dll
```

## Setting up the path to dynamic preprocessor engine in Snort

11. Now lets set our reputation preprocessors:

```
# path to dynamic rules libraries# dynamicdetection directory /usr/local/lib/snort_dynamicrules

252 # path to dynamic rules libraries
253 # dynamicdetection directory /usr/local/lib/snort_dynamicrules
```

## Path to dynamic rules libraries in Snort

1. Just comment out these lines as shown in figure 19 in doing so we are excluding packet normalization of different packets.

```
263 # Inline packet normalization. For more information, see README.normalize
264 # Does nothing in IDS mode
265 # preprocessor normalize_ip4
266 # preprocessor normalize_tcp: ips ecn stream
267 # preprocessor normalize_icmp4
268 # preprocessor normalize_ip6
269 # preprocessor normalize_icmp6
```

## Commenting out packet normalization commands in Snort

2. Scroll down to the reputation preprocessors. We will just change the name of the files since white list , black list are not rules they are just the list of IP addresses labelled as black or white

```
# Reputation preprocessor. For more information see README.reputationpreprocessor reputation:  
\memcap 500, \priority whitelist, \nested_ip inner, \whitelist $WHITE_LIST_PATH/whitelist,  
\blacklist $BLACK_LIST_PATH/black.list
```

```
511     whitelist $WHITE_LIST_PATH/white.list, \  
512     blacklist $BLACK_LIST_PATH/black.list
```

## Whitelisting and Blacklisting IPs through the command as shown in figure

3. Converted back slashes to forward slashes in lines 546–651.

```
545 # site specific rules
546 include $RULE_PATH\local.rules
547
548 include $RULE_PATH\app-detect.rules
549 include $RULE_PATH\attack-responses.rules
550 include $RULE_PATH\backdoor.rules
551 include $RULE_PATH\bad-traffic.rules
552 include $RULE_PATH\blacklist.rules
553 include $RULE_PATH\botnet-cnc.rules
554 include $RULE_PATH\browser-chrome.rules
555 include $RULE_PATH\browser-firefox.rules
556 include $RULE_PATH\browser-ie.rules
557 include $RULE_PATH\browser-other.rules
558 include $RULE_PATH\browser-plugins.rules
559 include $RULE_PATH\browser-webkit.rules
560 include $RULE_PATH\chat.rules
561 include $RULE_PATH\content-replace.rules
562 include $RULE_PATH\ddos.rules
563 include $RULE_PATH\dns.rules
564 include $RULE_PATH\dos.rules
565 include $RULE_PATH\experimental.rules
566 include $RULE_PATH\exploit-kit.rules
567 include $RULE_PATH\exploit.rules
568 include $RULE_PATH\file-executable.rules
569 include $RULE_PATH\file-flash.rules
570 include $RULE_PATH\file-identify.rules
571 include $RULE_PATH\file-image.rules
572 include $RULE_PATH\file-multimedia.rules
573 include $RULE_PATH\file-office.rules
574 include $RULE_PATH\file-other.rules
575 include $RULE_PATH\file-pdf.rules
576 include $RULE_PATH\finger.rules
577 include $RULE_PATH\ftp.rules
578 include $RULE_PATH\icmp-info.rules
579 include $RULE_PATH\icmp.rules
```

**Converted back slashes to forward slashes in specific lines in snort.conf file**

```
621 include $RULE_PATH\rservices.rules
622 include $RULE_PATH\scada.rules
623 include $RULE_PATH\scan.rules
624 include $RULE_PATH\server-apache.rules
625 include $RULE_PATH\server-iis.rules
626 include $RULE_PATH\server-mail.rules
627 include $RULE_PATH\server-mssql.rules
628 include $RULE_PATH\server-mysql.rules
629 include $RULE_PATH\server-oracle.rules
630 include $RULE_PATH\server-other.rules
631 include $RULE_PATH\server-webapp.rules
632 include $RULE_PATH\shellcode.rules
633 include $RULE_PATH\smtp.rules
634 include $RULE_PATH\snmp.rules
635 include $RULE_PATH\specific-threats.rules
636 include $RULE_PATH\spyware-put.rules
637 include $RULE_PATH\sql.rules
638 include $RULE_PATH\telnet.rules
639 include $RULE_PATH\tftp.rules
640 include $RULE_PATH\virus.rules
641 include $RULE_PATH\voip.rules
642 include $RULE_PATH\web-activex.rules
643 include $RULE_PATH\web-attacks.rules
644 include $RULE_PATH\web-cgi.rules
645 include $RULE_PATH\web-client.rules
646 include $RULE_PATH\web-coldfusion.rules
647 include $RULE_PATH\web-frontpage.rules
648 include $RULE_PATH\web-iis.rules
649 include $RULE_PATH\web-misc.rules
650 include $RULE_PATH\web-php.rules
651 include $RULE_PATH\xll.rules
```

**Converted back slashes to forward slashes in specific lines in snort.conf file**

4. Again just convert forward slashes to backslashes and uncomment the lines below:

```
# decoder and preprocessor event rules# include $PREPROC_RULE_PATH/preprocessor.rules#
include $PREPROC_RULE_PATH/decoder.rules# include $PREPROC_RULE_PATH/sensitive-
data.rules
```

```
657
658 # decoder and preprocessor event rules
659 include $PREPROC_RULE_PATH\preprocessor.rules
660 include $PREPROC_RULE_PATH\decoder.rules
661 include $PREPROC_RULE_PATH\sensitive-data.rules
```

**Converted back slashes to forward slashes in specific lines and uncommenting specific lines in snort.conf file**

**Now we just need to verify the presence of this command at the bottom of snort.conf file.**

```
688 # Event thresholding or suppression commands. See threshold.conf
689 include threshold.conf
690
```

verifying presence of “include threshold.conf” command in snort.conf file.

5. Click on Save file and save all changes to save the configuration file (snort.conf).

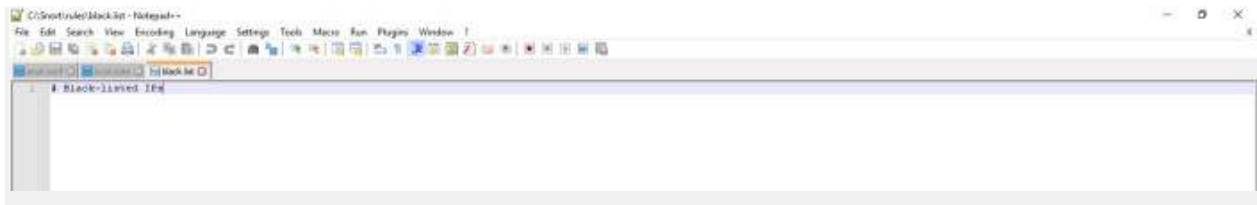
6. Now recalling the Step 13 white list, black list are not rules they are just the list of IP addresses labelled as black or white right now these files don't exist in our rule path which is why we have to create them manually, save them in this folder C:\Snort\rules.

- Go to Notepad++ and create new file.
- Comment it #White-listed IPs.
- Name the file white.list and save the file.



### Creating White List IPs file

- Create another new file.
- Comment it #Black-listed IPs.
- Name the file black.list and save the file.



### Creating Black List IPs file in Snort

1. Now we test snort again by running Command prompt as admin. To check if it's running fine after all the configurations.

```
C:\WINDOWS\system32\cmd.exe
Microsoft Windows [Version 10.0.17763.1282]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\>cd..

C:\Users>cd..

C:\>cd snort

C:\Snort>cd bin

C:\Snort\bin>snort -V

-*> Snort! <*-
o" )~ Version 2.9.17-WIN32 GRE (Build 199)
'...' By Martin Roesch & The Snort Team: http://www.snort.org/contact#team
      Copyright (C) 2014-2020 Cisco and/or its affiliates. All rights reserved.
      Copyright (C) 1998-2013 Sourcefire, Inc., et al.
      Using PCRE version: 8.10 2010-06-25
      Using ZLIB version: 1.2.3

C:\Snort\bin>
```

## Test Running of Snort in Windows 10 after Configuration

2. We can also check the wireless interface cards from which we will be using snort by using the command below we can see the list of our wireless interface cards through entering this command in command prompt.

### Snort — W

3. **configuration validation check command:**

Now we will enter a command To check validation of snort's configuration by choosing a specific wireless interface card (1) the rest of command shows the config file path . The command is :

**snort -i 1 -c C:\Snort\etc\snort.conf -T**

```

-- Initialization Complete --
C:\Snort> snort -i 1 -c C:\Snort\etc\snort.conf -T
Version 2.9.17/ NIDS ONE (build 170)
By Martin Roesch & The Snort Team: http://www.snort.org/contactteam
Copyright (C) 2004-2020 Cisco and/or its affiliates. All rights reserved.
Copyright (C) 1998-2010 Sourcefire, Inc., et al.
Using PCRE version 8.39 2016-06-25
Using LLVM version: 4.2.3

Rules Engine: SF_SNORT DETECTION ENGINE Version 3.1 (Build 1)
Preprocessor Object: SF_SHELL Version 1.1 (Build 4)
Preprocessor Object: SF_SNI Version 1.1 (Build 1)
Preprocessor Object: SF_SSH Version 1.1 (Build 5)
Preprocessor Object: SF_SIP Version 1.1 (Build 1)
Preprocessor Object: SF_SIP Version 1.1 (Build 1)
Preprocessor Object: SF_REPUTATION Version 1.1 (Build 1)
Preprocessor Object: SF_IPSEC Version 1.0 (Build 1)
Preprocessor Object: SF_MQOS Version 1.1 (Build 1)
Preprocessor Object: SF_PMAP Version 1.0 (Build 1)
Preprocessor Object: SF_GTP Version 1.1 (Build 1)
Preprocessor Object: SF_FRAGMENT Version 1.2 (Build 1)
Preprocessor Object: SF_DNS Version 1.1 (Build 4)
Preprocessor Object: SF_DNS Version 1.1 (Build 1)
Preprocessor Object: SF_DETECT Version 1.0 (Build 1)

Snort successfully validated the configuration!
snort exiting
C:\Snort\bin>

```

### Checking Validation of Snort Configuration in Command Prompt

It can be seen in the given figure that Snort successfully validates our configuration. This brings us to the end of our installation and configuration tutorial.



## Lab 8

.

Demonstrate how to provide secure data storage, secure data transmission and for creating digital signatures.

### INSTALLING THE SOFTWARE:

1. Visit [www.gpg4win.org](http://www.gpg4win.org). Click on the “Gpg4win 2.3.0” button
2. On the screen, click the “Download Gpg4win” button.
3. When the “Welcome” screen is displayed, click the “Next” button
4. When the “License Agreement” page is displayed, click the “Next” button
5. Set the check box values as specified below, then click the “Next” button
6. Set the location where you want the software to be installed. The default location is fine. Then, click the “Next” button.
7. Specify where you want shortcuts to the software placed, then click the “Next” button.
8. If you selected to have a GPG shortcut in your Start Menu, specify the folder in which it will be placed. The default “Gpg4win” is OK. Click the “Install” button to continue
9. A warning will be displayed if you have Outlook or Explorer opened. If this occurs, click the “OK” button.
10. The installation process will tell you when it is complete. Click the “Next” button
11. Once the Gpg4win setup wizard is complete, the following screen will be displayed. Click the “Finish” button
12. If you do not uncheck the “Show the README file” check box, the README file will be displayed. The window can be closed after you’ve reviewed it.

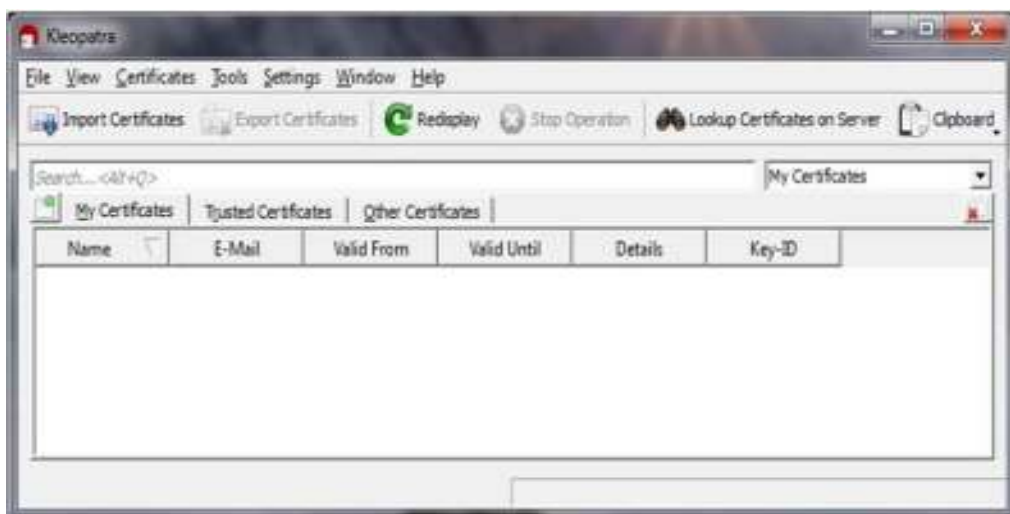
### CREATING YOUR PUBLIC AND PRIVATE KEYS

GPG encryption and decryption is based upon the keys of the person who will be receiving the

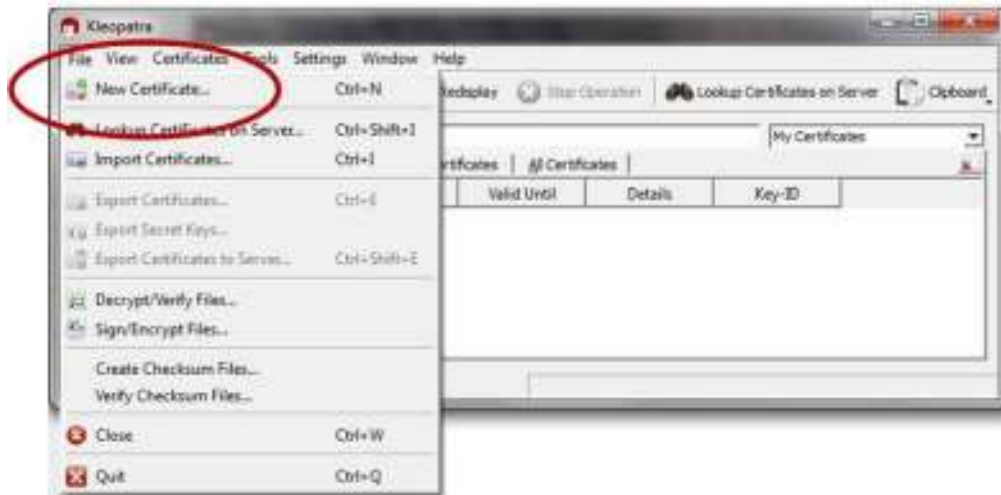
Department of Computer Science & Engineering

encrypted file or message. Any individual who wants to send the person an encrypted file or message must possess the recipient's public key certificate to encrypt the message. The recipient must have the associated private key, which is different than the public key, to be able to decrypt the file. The public and private key pair for an individual is usually generated by the individual on his or her computer using the installed GPG program, called "Kleopatra" and the following procedure:

1. From your start bar, select the "Kleopatra" icon to start the Kleopatra certificate management software.
2. The following screen will be displayed



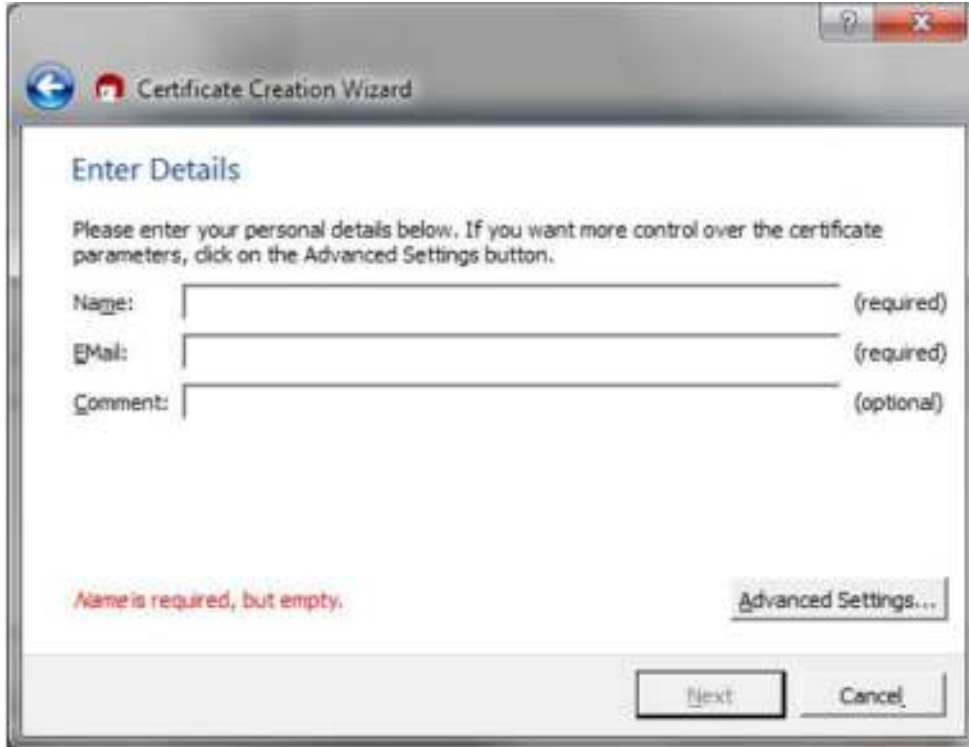
1. From the “File” dropdown, click on the “New Certificate” option



2. The following screen will be displayed. Click on “Create a personal OpenPGP key pair” and the “Next” button

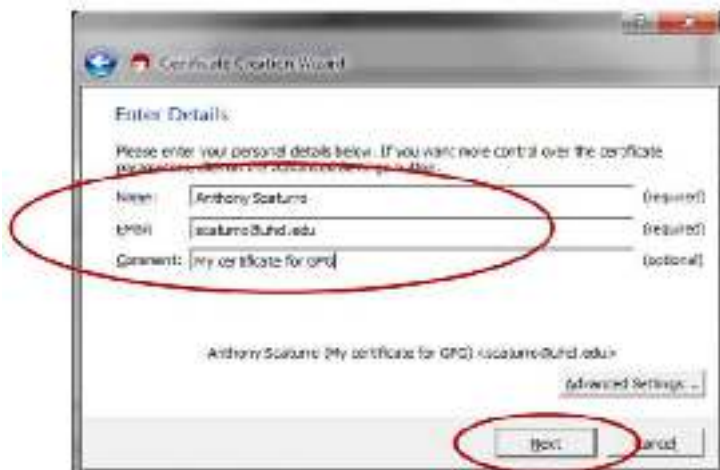


3. The Certificate Creation Wizard will start and display the following:



The screenshot shows the 'Enter Details' step of the Certificate Creation Wizard. It includes three input fields: 'Name:' (required), 'EMail:' (required), and 'Comment:' (optional). A red error message 'Name is required, but empty.' is displayed below the Name field. An 'Advanced Settings...' button is located to the right of the Comment field. At the bottom, there are 'Next' and 'Cancel' buttons.

4. Enter your name and e-mail address. You may also enter an optional comment. Then, click the “Next” button.



The screenshot shows the 'Enter Details' step of the Certificate Creation Wizard with the fields filled. The 'Name' field contains 'Anthony Scaturro', the 'EMail' field contains 'ascaturro@iuhd.edu', and the 'Comment' field contains 'My certificate for GPG'. A red oval highlights the input fields. Another red oval highlights the 'Next' button at the bottom. The 'Advanced Settings...' button is also visible.

5. Review your entered values. If OK, click the “Create Key” button



6. You will be asked to enter a passphrase



7. The passphrase should follow strong password standards. After you've entered your passphrase, click the "OK" button.



8. You will be asked to re-enter the passphrase

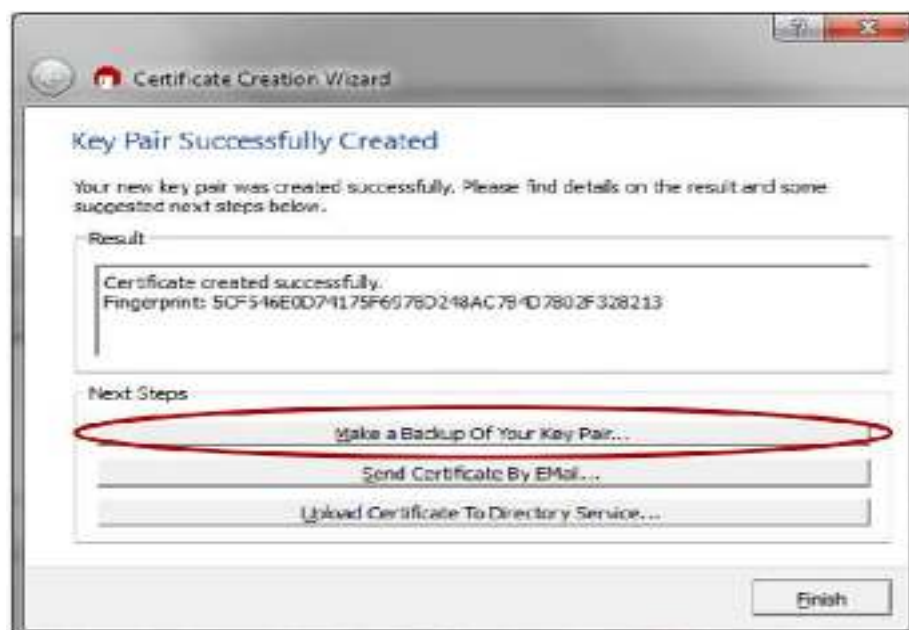




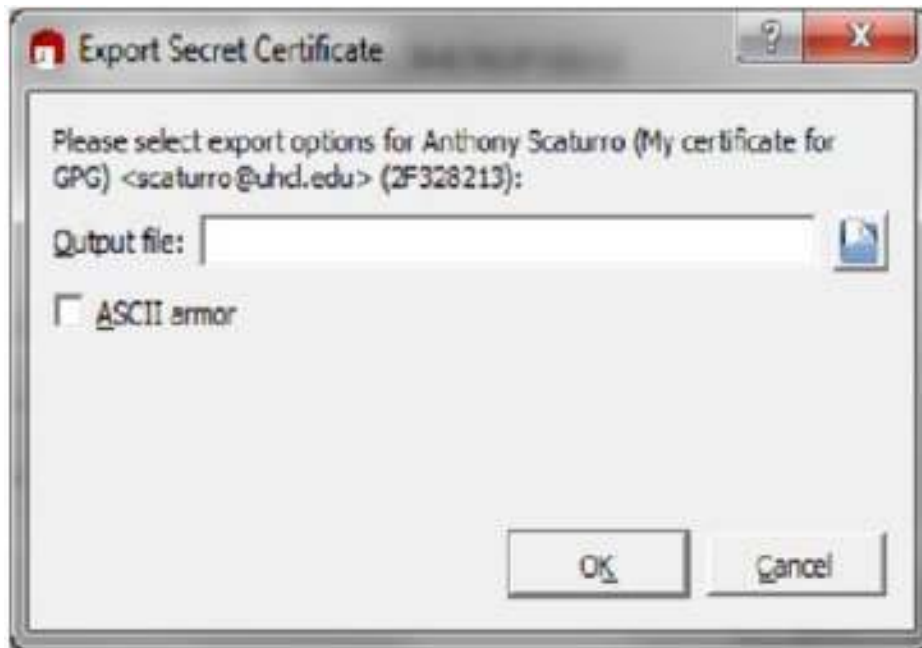
9. Re-enter the passphrase value. Then click the “OK” button. If the passphrases match, the certificate will be created.



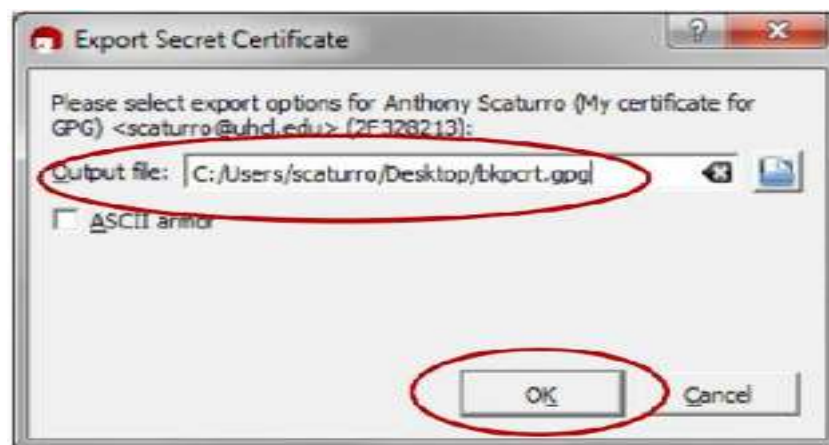
10. Once the certificate is created, the following screen will be displayed. You can save a backup of your public and private keys by clicking the “Make a backup Of Your Key Pair” button. This backup can be used to copy certificates onto other authorized computers.



11. If you choose to backup your key pair, you will be presented with the following screen:



13. Specify the folder and name the file. Then click the “OK” button.

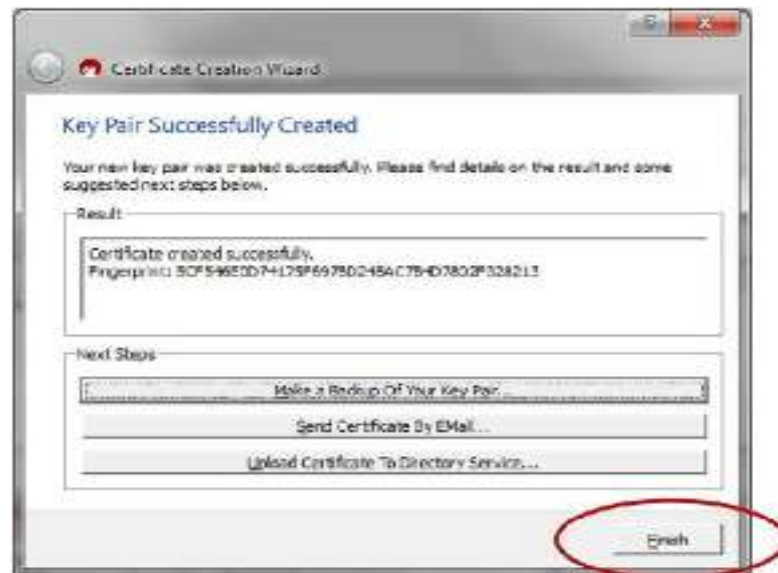




14. After the key is exported, the following will be displayed. Click the “OK” button.



14. You will be returned to the “Key Pair Successfully Created” screen. Click the “Finish” button.



15. Before the program closes, you will need to confirm that you want to close the program by clicking on the “Quit Kleopatra” button.



#### References:

- 1) Cryptography and Network Security: Principles and Practice, William Stallings, Pearson.
- 2) Cryptography & Network Security (McGraw-Hill Forouzan Networking, Behrouz A. Forouzan, McGraw Hill Education.

#### Viva Questions:

##### 1. What is cryptography?

Cryptography is a specialized area of cybersecurity, but it has a broad array of applications that we will examine later. Kaspersky Lab has defined it as follows: “Cryptography is the study of secure communications techniques that allow only the sender and intended recipient of a message to view its contents. In addition, cryptography also covers the obfuscation of information in images using techniques such as microdots or merging.”

##### 2. What exactly are encryption and decryption?

The terms “scrambling” and “descrambling” are commonly known. In terms of decryption, scrambling and descrambling are also known as “encryption” and “decryption.”

For example: when the written message “I LOVE YOU” is scrambled by the sending party, it becomes what is known as the “encrypted message.” This means that the written message has been disguised in such a manner that it would be totally meaningless, or in the terms of cryptography, it would be undecipherable.

Encryption can also be described as conversion of information from a readable state to apparent nonsense. When the receiving party receives this encrypted written message, it must be unscrambled into an understandable and comprehensible state of context. This process of unscrambling is also known as decryption

### 3. What is plaintext or cleartext?

The decrypted message, when it is returned back into its plain or original state of context which is comprehensible and decipherable, is also known as cleartext or plaintext.

### 4. What is ciphertext?

When the message is encrypted into a state which is totally incomprehensible and undecipherable, this is known as the ciphertext. So, to illustrate all of this, with the previous example, when the sending party creates the written message of “I LOVE YOU”, this is the plaintext or the cleartext. Once this message is encrypted into the format of “UYO I VEOL” and while it is in transit, it becomes known as the ciphertext. Then, once the receiving party gets this ciphertext and then decrypts it into a comprehensible and understandable form of “I LOVE YOU,” this message then becomes the plaintext or the cleartext again.

### 5. How does the encryption process actually take place?

This is a question in which we will have more specific answers for later on. But generally speaking, in its simplest form, the text or the written message is encrypted via a special mathematical formula. This formula is specifically known as the “encryption algorithm.” Because the ciphertext is now encrypted by this special mathematical algorithm, it would be rendered useless to a third party with malicious intent, because of its totally garbled nature.

### 6. What are the origins of cryptography?

For almost as long as people have been writing, people have wanted to protect what was written. According to some scholars, cryptography can be traced all the way back to 1900 BC, when the tomb of Khnumhotep II used unknown hieroglyphs to apparently mask the intent of a carved message. Other early messages include simple ciphers on Mesopotamian clay tablets and the Greek use of a “scytale,” a decoding stick, which would reveal a message when a strip of cloth with a cipher on it was wrapped around it.

### 7. What is the Caesar cipher?

In the Caesar methodology, each letter of the text or the written message is substituted with another letter of the alphabet which is so many spaces or letters later in the alphabet. This is probably the simplest form of encryption, because each letter in plain text message is literally substituted by another

letter, thus forming the ciphertext. This methodology (which was said to be used by Julius Caesar) is probably the most-cited type of algorithm in academic literature.

8. What is the goal of cryptography?

Although the main purpose of cryptography appears to be making content and images undecipherable, the true goal of cryptography in an information technology context is to ensure the confidentiality and integrity of any information technology system. In other words, the content and images must remain private between the sending and the receiving parties; while they are in transit across the Internet, assurances must be provided that they will remain intact and not altered in any way.

9. Are there any other ciphers that are available, other than the Caesar cipher?

Yes, there are. As cryptography has evolved over time, so has the degree of sophistication of these other ciphers.

10. Just how important is the field of cryptography?

Cryptography is going to play a very large role in cybersecurity today and in the future. For example, it will be vital to encrypt all kinds and types data, especially as it relates to a business or corporation and their customers.

### Level 2 Questions

1. What is the difference between a private key and a public key?

As it was alluded to earlier, one of the main purposes of cryptography is to scramble forms of content and images into an undecipherable state. You may be wondering how this is all exactly done. The answer is that it primarily involves the use of a key. Traditionally, this is a private key. With this particular key, the sending party can encrypt the plaintext, and from there the content or image will be sent in its garbled state across the network medium to the receiving party. A private key is private to the sender or the receiver, while a public key may be available to a group.

2. What are symmetric and asymmetric key systems?

A symmetric key system uses only the private key, and the asymmetric key system makes use of both the public key and the private key. The latter used primarily in what is known as a Public Key Infrastructure, or PKI for short. It will be discussed in more detail later on.

3. What kinds of threats exist for a cryptographic system?

There are three traditional types of attacks, and they are as follows:

**Ciphertext-only attack:** With this type of attack, only the ciphertext is known to the attacker. But if this particular individual is well-trained in statistics, then he or she can use various statistical techniques to break the ciphertext back into the plaintext

**Known-plaintext attack:** This occurs when the hacker knows some aspect of either the letter pairings; thus, they can consequently crack the ciphertext back into the plaintext

Chosen-plaintext attack: With this type of attack, the hacker can choose the plaintext and view the encrypted output which is being transmitted across the network medium. From this, they can reverse-engineer it back into its ciphertext form in an attempt to figure out the specific encryption scheme

#### 4. What is polyalphabetic encryption?

This was listed as a specific type of cipher earlier. A polyalphabetic cipher is simply a substitution cipher that uses multiple alphabets for substitution.

#### 5. What is a block cipher?

With this method of transposition, the plaintext message is encrypted into its scrambled format by being broken up into blocks and encrypted block-by-block. Let us illustrate this with our example used before, but this time, let us assume a block of three characters, mathematically represented as 3 bits, or where  $k=3$ .

Plaintext: I LOVE YOU

Plaintext Block: ILO VEY OUX

Ciphertext Block: OLI YEV XUO

Ciphertext: OLIYEVXUO

#### 6. What is cipher block chaining?

The initialization vectors are part of a larger process known as cipher block chaining, or CBC. Within this methodology, multiple loops of encryption are created in order to further totally scramble the ciphertext.

Here is the how the process works:

The Initialization Vector is created first

Through a mathematical process known as XOR (which stands for exclusive OR and is used quite frequently to determine if the bits of two strings of data match or not), the first created Initialization Vector is XOR'd with the first block of ciphertext data

The first chunk of data which has been XOR'd is further broken down by another layer of encryption. This process is then continued until all of the blocks of ciphertext have been XOR'd and enveloped with another layer of encryption

This is how cipher block chaining gets its title. For instance, steps 1-4 create the first loop or chain; the second loop or chain is then next initiated, and so on, until the ciphertext has been fully analyzed and encrypted by this methodology.

#### 7. What are the disadvantages of symmetric key cryptography?

Symmetric key cryptography suffers from three major vulnerabilities:

Key storage and recovery

Department of Computer Science & Engineering

## Key distribution

### Open systems

As previously mentioned, symmetric cryptography requires the sharing of secret keys between the two parties (sending and receiving), which further requires the implicit trust that this key will not be shared with any other outside third party. The only way that any type of secrecy can be achieved in this regard would be to establish some sort of trusted channel. An option here would be the use of a so-called designated controller. But this carries third-party risks as well.

With regards to the second vulnerability, since there will be many more lines of communication between the sending and the receiving parties, the need to implement more controllers becomes totally unrealistic as well as unfeasible. Thus, the distribution of the private keys can become a virtual nightmare.

Finally, with the third vulnerability, private or symmetric cryptography works best only when it is used in a very closed or “sterile” environment, where there are at best only a few (or even just a handful) of sending and receiving parties. In other words, given the threat landscape today, it would be completely unrealistic to implement a symmetric cryptography system in an open environment.

### 8. How is a Key Distribution Center (KDC) used?

The Key Distribution Center consists of a database of all of the end users at the place of business or corporation and their respective passwords, as well other trusted servers and computers along the network.

If an end user wishes to communicate with another end user on a different computer system, the sending party enters their password into the KDC using a specialized software called “Kerberos.” When the password is received by the KDC, the Kerberos then uses a special mathematical algorithm which adds the receiving party’s information and converts it over to a cryptographic key.

Once this encrypted key has been established, the KDC then sets up and establishes other keys for the encryption of the communication session between the sending and the receiving party. These other keys are also referred to as tickets. These tickets will actually expire at a predetermined point in time in order to prevent unauthorized use, and it would also be rendered useless if it is stolen, hijacked or intercepted by a third party.

### 9. What are the mathematical algorithms used in symmetric cryptography?

They are as follows:

The Needham-Schroder algorithm

The Digital Encryption Standard algorithm (DES)


The Triple Digit Encryption Standard algorithm (3DES)

The International Data Encryption Algorithm (IDEA)

The Advanced Encryption Standard algorithm (AES)

### 10. What is the hashing function?

Department of Computer Science & Engineering

  
**Dr. Mahesh Bunde**  
B.E., M.E., Ph.D.  
Director  
Poonima College of Engineering  
ISI-0, RICO Institutional Area  
Sitapura, JAIPUR



The hashing function is a one-way mathematical function. This means that it can be used to encode data, but it cannot decode data. Its primary purpose is not to encrypt the ciphertext; rather, its primary purpose is to prove that the message in the ciphertext has not changed in any way, shape or form. This is also referred to as “message integrity.” If the mathematical function has changed in any way, the message has then changed.

### Medium level Questions:

What is asymmetric key cryptography?

In the most simplistic terms, asymmetric cryptography can be likened to that of a safety deposit box at a local bank. In this example, there are normally two set of keys used. One key is the one which the bank gives to you. This can be referred to as the public key, because it is used over and over again. The second key is the private key which the bank keeps in their possession at all times, and only the bank personnel know where it is kept.

The world of asymmetric cryptography is just like this example, though of course, it is much more complex than this in practice.

Let us refer to the public key as “pk” and the private key as “sk.” So, to represent both of these keys together, it would be mathematically demonstrated as (pk, sk). It is then the sending party which uses the public key (pk) to encrypt the message they wish to send to the receiving party, which then uses the private key (sk) to decrypt the ciphertext from the sending party.

2. What are the key differences between asymmetric and symmetric cryptography?

With symmetric cryptography, the complete secrecy of the key must be assured. Whereas asymmetric cryptography requires only half of the secrecy, namely that of the private key (sk).

Secondly, symmetric cryptography utilizes the same secret key for the encryption and decryption of the ciphertext, but in asymmetric cryptography two different keys (namely the public and the private keys) are used for the encryption and the decryption of the ciphertext.

3. What are the disadvantages of asymmetric cryptography?

Despite the advantages that asymmetric cryptography has, it does possess one very serious disadvantage: When compared to symmetric cryptography, it is two to three times slower than symmetric cryptography. This is primarily because of the multiple parties and multiple keys which are involved.

4. What are the mathematical algorithms used in asymmetric cryptography?

There are three of them that are primarily used:

The RSA algorithm

The Diffie-Hellman algorithm

The Elliptical Wave Theory algorithm

### 5. What is the Public Key Infrastructure (PKI)?

Since the public key has become so important in the encryption and the decryption of the ciphertext messages between the sending and receiving parties and given the nature of its public role in the overall communication process, great pains and extensive research have been taken to create an infrastructure which would make the process of creating and sending keys much more secure and robust.

In fact, this infrastructure is a very sophisticated form of asymmetric cryptography, and it is known as the “Public Key Infrastructure” or “PKI” for short. The basic premise of PKI is to help create, organize, store, distribute and maintain the public keys.

### 6. What are the specific components of the Public Key Infrastructure (PKI)?

The PKI consists of the following components:

The Certificate Authority (CA): This is the party who issues the digital certificates

The Digital Certificate: This serves to verify the identity of the certificate holder and is issued by the CA. These digital certificates are typically kept in the local computer of the employee, or even the central server at the place of business or organization

The LDAP or X.500 Directories: These are the databases which collect and distribute the digital certificates from the CA

The Registration Authority (RA): If the place of business or organization is very large (such as a multinational corporation), this entity usually handles and processes the requests for the required digital certificates and then transmits those requests to the CA to process and create the required digital certificates

### 7. What are the technical specifications of the Certificate Authority?

The Certificate Authority consists of the following technical specifications:

The digital certificate version number

The serial number

The signature algorithm identifier

The issuer name

The validity period

The public key

The subject distinguished name

The subject alternate name email

The subject name URL


### 8. How does the Public Key Infrastructure (PKI) work?

At a macro level, this is how the Public Key Infrastructure (PKI) works:

The request for the Digital Certificate is sent to the appropriate Certificate Authority (CA)

After this request has been processed, the Digital Certificate is issued to the person who is requesting it

Department of Computer Science & Engineering

  
**Dr. Mahesh Bunde**  
B.E., M.E., Ph.D.  
Director  
Poonima College of Engineering  
ISI-0, RICO Institutional Area  
Sitapura, JAIPUR



The Digital Certificate then gets signed by confirming the actual identity of the person who is requesting it

The Digital Certificate can now be used to encrypt the plaintext into the ciphertext which is sent from the sending party to the receiving party.

9. What is the LDAP protocol and how is it used in a Public Key Infrastructure (PKI)?

LDAP is an acronym which stands for Lightweight Directory Access Protocol. This is a database protocol used for the updating and searching of the directories which run over the TCP/IP network protocol (this is the network protocol which is primarily used by the PKI infrastructure).

It is the job of the LDAP server of the Public Key Infrastructure to contain information and data as it relates to the digital certificates and the public and the private key storage locations, as well as the matching public and private key labels.

The Certificate Authority uses a combination of the end user name and the matching tags to specifically locate the digital certificates on the LDAP server. From that point onwards, the LDAP server checks to see if the requested digital certificate is valid or not, and if it is valid, it then retrieves a digital certificate which can then be sent to the end user.

Although all digital certificates have a finite lifespan when they are first issued, they can also be revoked for any reason at any time by the Public Key Infrastructure Administrator.

10. What are the security vulnerabilities of hashing functions?

One major security vulnerability of using hashes is that they can be altered while it is en route. In other words, a cyber-attacker can intercept the ciphertext and its associated hash, alter both and create a brand-new ciphertext and hash.


As a result, the receiving party is fooled into believing that this new, altered ciphertext and new, altered hash are the original sent by the sending party while the cyber-attacker keeps the actual ciphertext and hash which was generated the first time around.

To fix this, the ciphertext is combined with a “secret key” at the point of origination first, then the hash is created. As a result, this hash will contain specific information and data about the secret itself. As a result, the receiving party can even be further convinced that the ciphertext they have received is the original one sent by the sending party.


This is so because even if the ciphertext, the hash and the associated secret key were to be intercepted, there is very little that a hacker can do to alter the ciphertext and its associated hash. This is because they have to have the information and data about the secret key, which is of course something they will never gain access to.

---


Department of Computer Science & Engineering

  
**Dr. Mahesh Bunde**  
B.E., M.E., Ph.D.  
Director  
Poornima College of Engineering  
ISI-0, FIICO Institutional Area  
Sikapura, JAIPUR

Department of Computer Science & Engineering

  
**Dr. Mahesh Bunde**  
B.E., M.E., Ph.D.  
Director  
Poornima College of Engineering  
ISI-0, FIICO Institutional Area  
Sikapura, JAIPUR

Department of Computer Science & Engineering

  
**Dr. Mahesh Bunde**  
B.E., M.E., Ph.D.  
Director  
Poornima College of Engineering  
ISI-0, FIICO Institutional Area  
Sikapura, JAIPUR

