Indira Gandhi National Open University

MSEP – 028

"**IMPLEMENTING WEB SERVICES SECURITY WITH PCA**"

By
Prakash Chourasia
Enrolment No: 118035704

Synopsis
Submitted to the School of Vocational Education and Training, IGNOU
in partial fulfilment of the requirements
for the award of the degree
PG Diploma in Information Security (PGDIS)
2013

Indira Gandhi National Open University
Maidan Garhi
New Delhi – 110068

Table of Contents

- **Introduction :**

In current day scenario, the need to protect information has become very important and hence the need for cryptographic algorithms is high. When it comes to enterprise security and even home security, encryption is one of the essentials from corporate security to shopping online via the internet.
For many, encryption is a transparent technology that is relied upon as being adequate for the task. To start with, encryption is only part of the process required to ensure that data is secure. As with any form of protection, an item is only secure as long as no one has access to it. encryption is similar in that it uses keys to 'lock' up the data which can only be solved if you have the appropriate key.

This technique is known as symmetric or private key encryption. When files are encrypted, the creator uses a private-key to encrypt and decrypt the file. This private key is then shared to those who need to decrypt the file or send encrypted messages back to the original creator. Symmetric key cryptography is both highly efficient and secure when used with appropriate key management techniques.

On the other hand, in direct parallel is asymmetric encryption or public key cryptography, which was devised in the 1970s. This technique uses two sets of keys, one to encrypt and one to decrypt.
Public key /private key cryptography is use of asymmetric key algorithms because the key used to encrypt a message is not the same as the key used to decrypt it.

Here, we extend the parallel key encryption algorithm and bring out its full potential by implementing the various cryptographic modes such as cipher block chaining and

interleaved cipher block chaining where commendable increase in efficiency and reduction in encryption and decryption time can be seen. We have also considerably increased the key size by using 1024 bit and 2048 bit keys for the algorithmic implementation and in CBC and interleaved CBC execution. Our practical analysis has brought to front the salient features of the parallel key encryption algorithm and its ability to provide enhanced data protection when using a larger key size along with its randomness property. In theoretical analysis, it can be shown that remarkable reduction in encryption and decryption time of cryptographic systems is achieved and an enhanced strength to the system against brute force attacks is achieved.

Furthermore, PCA can be extended for different cryptographic modes, using varying key size and number of keys. There have been several multiple key encryption algorithms proposed for different applications but for the very first time we have extended and implemented the cryptographic modes such as cipher block chaining and interleaved cipher block chaining. In this proposed algorithm, we have substantially used a larger key size of 1024 bit and 2048 bit range in order to provide for unmatched protection and security to data. Moreover, with an increased key size the encryption and decryption time taken have been reduced commendably.

We have implemented the parallel key encryption algorithm to its fullest potential along with increased key size strengthens the system against factorization attack, brute force attacks and provides for unparalleled protection of sensitive data. Similarly, it can also be applied in digital signatures and in internet transactions. This makes our proposed mechanism one of the best mechanisms in the cryptographic world.

Feasibility Study:

Feasibility studies aim to objectively and rationally uncover the strengths and weaknesses of the existing business or proposed venture, opportunities and threats as presented by the environment, the resources required to carry through, and ultimately the prospects for success.

Feasibility studies aim to objectively and rationally uncover the strengths and weaknesses of the existing business or proposed venture, opportunities and threats as presented by the environment, the resources required to carry through, and ultimately the prospects for success. In its simplest term, the two criteria to judge feasibility are cost required and value to be attained.[3] As such, a well-designed feasibility study should provide a historical background of the business or project, description of the product or service, accounting statements, details of the operations and management, marketing research and policies, financial data, legal requirements and tax obligations.[1] Generally, feasibility studies precede technical development and project implementation.

In our project, we have implemented the task to the latest systems and software and therefore feasibility of the project is as required. In latest environment, all resources are upto the mark to execute the task as per the requirement. The various feasibilities are as follows:

1) System & Technical Feasibility: Since hardware and software available in the industry are used to develop the project and hence it has been found that the system will work on almost the current environment resources. Since JAVA has been used to implement the task therefore software is platform independent and supports all OS available in the industry.

2) Economic Feasibility: Software cost is within the limits and does not impose any additional cost for implementation of the system therefore project is economically feasible for implementation in a company.

3) Legal Feasibility: Digitally signing a document or text increases the legality of the software and it is not restricted to sign any document across the world to it never violets any legal terms.

4) Operational Feasibility: The system requires normal configurations during execution and do not impose any additional system configuration. It uses a local area network and normal databases for it execution.

• Research Methodology

- Object of our project is to provide improved security over network and generating trust among the participating nodes; by applying threshold cryptography techniques.
- The proposed work focuses on Threshold Cryptography using PCA algorithm because nodes have large computational and storage capacity as per the requirement of the PCA Algorithm in contrast with RSA algorithm which are applied when nodes have smaller computational and storage capacity.

- It is a branch of public key cryptography in general, and multi-party computation in particular. Essentially, in a k-out-of-n threshold crypto-system,

• Any k or more out of n total parties, when they come together, they can "reconstruct" the secret d in a way which enables them to decrypt or sign a message. This should be done in a way that does not reveal the value of d and its shares to any one in the scheme.

• Secondly, signing or decryption will be totally impossible in the circumstance

where less than k parties are present.

- The idea of threshold cryptography is to protect information (or computation) by fault-tolerantly distributing it among a cluster of cooperating computers. First consider the fundamental problem of threshold cryptography, a problem of secure sharing of a secret. A secret sharing scheme allows one to distribute

a piece of secret information among several servers in a way that meets the following requirements

(1) no group of corrupt servers (smaller than a given threshold) can figure out what the secret is, even if they cooperate;

(2) when it becomes necessary that the secret information be reconstructed, a large enough number of servers (a number larger than the above threshold) can always do it.

A very useful extension of secret sharing is function sharing. Its main idea is that a highly sensitive operation, such as decryption or signing, can be performed by a group of cooperating servers in such a way that no minority of servers is able to perform this operation by themselves, nor would they be able to prevent the other servers from performing the operation when it is required.

In the threshold setting, we would like to implement, via efficient protocols, the most secure cryptosystems and signature schemes.

various considerations we make when modeling computer faults:

- *The size of the threshold:* What fraction of the servers can be corrupted by the adversary without any harm to the service (e.g. signature or decryption) that these servers implement?
- *Efficiency considerations:* How much communication, storage, and computation do these fault-tolerant protocols require?
- *Model of commmunication:* How realistic are the requirements we place on it? Do we require synchronous or partially synchronous communication, authenticated broadcast and secure links between servers?
- *Type of adversary we tolerate:* How does the adversary choose which players to corrupt? Can a server securely erase its local data so that it cannot be retrieved by the adversary once the server is infiltrated?

General Outline Of Report

In this report, we are providing the minute details of the project, that is how it is working and the various module which are implemented in this project. It includes the design diagrams and textual analysis of the project.

At end we have discussed about the future enhancements which can be done in the project and conclusion of the project includes the various unique features we have included alongwith the advantages and disadvantages of the project.

- # Analysis

User Requirement

To develop system software which is more secure to vulnerable to attacks and flexible enough to compute the work in a beneficial manner. The software should posses the functions like it should be able to maintain its resistance with the new attacks in present global world. It should provide the fast way to perform encryption & decryption with high security so that it can be used in business or private networks.

System Level Requirement

1.System design

1. Modular decomposition and explanation.

In this algorithm first we divided the whole process into three basic modules

1. Parallel key generation process
2. Cipher block chaining
3. Interleaved cipher block chaining.

2. Cohesion and coupling between modules.

In this process, first the random number will be generated by key generation algorithm. This will generate a unique number for every new execution. This generated number is a primary key ie. This will be different from every other key generated in this process.

Then the data will be divided into number of similar blocks, these block divides the data into frames.

When all frames are formed, these are collectively joined with encryption key and then the newly formed encrypted blocks will be obtained. Likewise, this process is repeated and the frames are send to the receiving end where the decryption of data frames is done.

Decryption is just the opposite process of encryption. In this the joined frames are decoded and separated and thus the data is obtained

Software Model and Justification

Software Development Life Cycle

Systems Development Life Cycle (SDLC) is a process used by a systems analyst to develop an information system, including requirements, validation, training, and user (stakeholder) ownership. Any SDLC should result in a high quality system that meets or exceeds customer expectations, reaches completion within time and

cost estimates, works effectively and efficiently in the current and planned Information Technology infrastructure, and is inexpensive to maintain and cost-effective to enhance.

Software development life cycle has various steps well defined and specified. We have followed all the steps of SDLC strictly to develop this project and have included all the steps of as we have followed in this report.

System Development Life Cycle: We are using incremental model of the SDLC, which requires enhancing the project in modules.

The **Systems Development Life Cycle (SDLC)**, or *Software Development Life Cycle* in systems engineering, information systems and software engineering, is the process of creating or altering systems, and the models and methodologies that people use to develop these systems. The concept generally refers to computer or information systems. In software engineering the SDLC concept underpins many kinds of software development methodologies. These methodologies form the framework for planning and controlling the creation of an information system.

A Systems Development Life Cycle (SDLC) adheres to important phases that are essential for developers, such as planning, analysis, design, and implementation, and are explained in the section below. A number of system development life cycle (SDLC) models have been created: waterfall, fountain, spiral, build and fix, rapid prototyping, incremental, and synchronize and stabilize. The oldest of these, and the best known, is the waterfall model: a sequence of stages in which the output of each stage becomes the input for the next. These stages can be characterized and divided up in different ways, including the following:

- **Project planning, feasibility study**: Establishes a high-level view of the intended project and determines its goals.

- **Systems analysis, requirements definition**: Defines project goals into defined functions and operation of the intended application. Analyzes end-user information needs.
- **Systems design**: Describes desired features and operations in detail, including screen layouts, business rules, process diagrams, pseudocode and other documentation.
- **Implementation**: The real code is written here.
- **Integration and testing**: Brings all the pieces together into a special testing environment, then checks for errors, bugs and interoperability.
- **Acceptance, installation, deployment**: The final stage of initial development, where the software is put into production and runs actual business.
- **Maintenance**: What happens during the rest of the software's life: changes, correction, additions, moves to a different computing platform and more. This, the least glamorous and perhaps most important step of all, goes on seemingly forever.

Cost Estimation

Considering economical factors, cost for the whole projects is minimum. The need of security required for system where this software will be used determines the cost.

This project is mainly designed for highly secure transmission of data and information. The cost for initial setup is same as in previous proposed models.

Cocomo model is used to estimate the cost for this project. It includes the cost of developing the architecture , cost for programmers , system design and implementation.

The software is developed in stages consulting each phase with the user requirement. Overall, prototype model is used to implement this software with ease and use.

Scheduling Estimation

The schedule estimation can be done through “COCOMO Model”. There are various model for “COCOMO Model”, I have taken the first model name – “Model –I”.

Model1. The basic COCOMO model computes software developments efforts (and cost) as a function of program size expressed in estimated lines of code.

COCOMO consists of a hierarchy of three increasingly detailed and accurate forms. The first level, *Basic COCOMO* is good for quick, early, rough order of magnitude estimates of software costs, but its accuracy is limited due to its lack of factors to account for difference in project attributes (*Cost Drivers*). *Intermediate*

COCOMO takes these Cost Drivers into account and *Detailed COCOMO*additionally accounts for the influence of individual project phases.

When estimating software size, the best way may be to get as much details as possible about the software to be developed and to be aware of our biases when estimating the size of the various components .By obtaining details and using them for size estimation, the estimates are likely to be closer to the actual size of the final software. LOC is used as the measure of size in the model.

Constants for different project types: -

System	A	b
Organic	3.2	1.05
Semidetached	3.0	1.12
Embedded	2.8	1.20

If project is organic, the values are: -

c = 2.5 and d = 0.38

From the requirements there will be few modules in the software, these are; tender, quotation, employees details, supply, stock, bill. It is clear from the requirement that this project will fall in the organic category.

The size of different tasks and overall system estimated as follows: -

---

Preliminary Investigation = 5 * 25 = 125 = 125/1000=0.125 KDLOC

System Analysis = 8 * 25 = 200 = 200/1000=0.2 KDLOC

System Design = 14 *25 = 350 = 350/1000=0.35 KDLOC

Coding = 138 * 25 =3450= 3450/1000= 3.45 KDLOC

Testing = 3 * 25 = 75 = 75/1000 = 0.075 KDLOC

Total = 4.2 KDLOC

1. The initial efforts estimate for the project is obtained from the relevant equations , we have

   $E= a * (KDLOC)^{1.05}$

   $E= 3.2 * (4.2)^{1.05}$

   E= 14.43 PM

2. Overall Duration :-

   D = c * (E) d

   $D= 2.5 * (14.43)^{0.38}$

   D= 6.89 month approx

3.To get the phase-wise breakup of cost. We use the distribution of costs. Thus the phase-wise cost breakup for the project is :-

---

Preliminary Investigation = 0.125 *14.43 = 1.8 PM

System Analysis = 0.2*14.43 = 2.88 PM

System Design = 0.35 *14.43 = 5.05 PM

Coding = 3.4 *14.43 = 49.06 PM

Testing = 0.07 *14.43 = 1.01 PM

4.The distribution of the schedule in different activities is:

Preliminary Investigation =10 % = 0.10* E = 0.10* 14.43 = 1.43 PM

System Analysis = 10% = 0.10*E = 0.10* 14.43 = 1.43 PM

System Design = 25 % = 0.25*E = 0.25* 14.43= 3.60 PM

Coding = 40% = 0.40* E = 0.40* 14.43 = 5.77 PM

Testing = 15% = 0.15* E = 0.15* 14.43= 2.16 PM

- # Case Study

In this project first we have divided the whole process into three basic modules

1. Parallel key generation process
2. Interleaved cipher block chaining.

First lets us see the parallel key generation module in detail. In PCA, there are keys namely major and minor key. The length of the major key is greater than the minor key.

Key Generating process:

This a method of generating the two pairs of key used for encryption and decryption. The key size chosen for minor key is 1024 bit and for major key is 2048 bit. Here, we have made use of the concept of BigInteger in java for the key generation process.

ENCRYPTION:

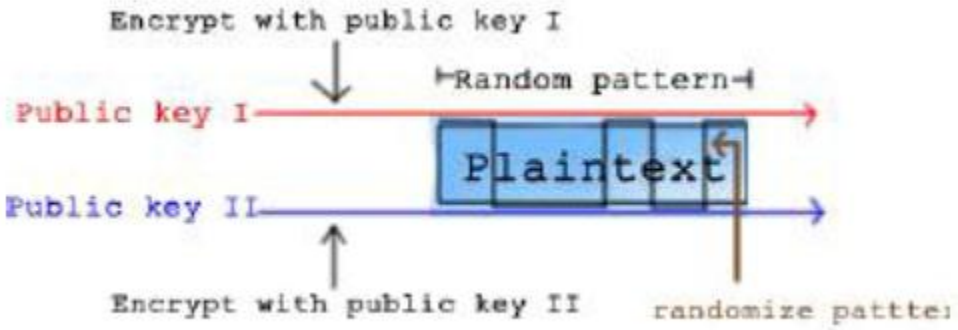


Figure 1. Encryption process of Parallel-key Cryptographic Algorithm

From the above diagram we show the encryption of a plain text block which is being encrypted by the major and minor key .Here, for experimental purpose we had taken a text file as the input file and divided into a number of blocks of a determined size say, 1024. Once the file was divided into blocks of specified size then they were respectively encrypted by the major and minor key.

RANDOM PATTERN:

Figure 2 Random pattern of Parallel-key Cryptographic Algorithm

This is one of the unique features of PCA. Here, we generate a random pattern whenever the PCA module is called. The pattern determines the order in which the blocks of data will be encrypted and decrypted –major and minor

key [1]. It is this randomness which provides the enhanced protection and edge to PCA when compared to other algorithms.

DECRYPTION:

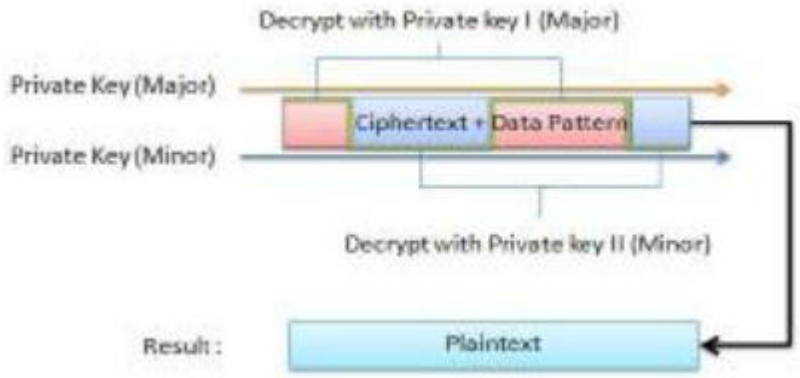


Figure 3 Decryption process of Parallel-key Cryptographic Algorithm

Here in the decryption process the true potential of PCA is put into front by using the minor key which substantially accelerates the time taken to decrypt and at the same time providing enhanced protection of data.

CIPHER BLOCK CHAINING-

This is one of the important cryptographic modes protecting data against factorization and replay attacks. Here the input text file is divided into blocks as in above and the first block is XORed with a randomly generated initialization vector and then encrypted. The cipher text hence formed is used to XOR with the subsequent text blocks during encryption. The reverse

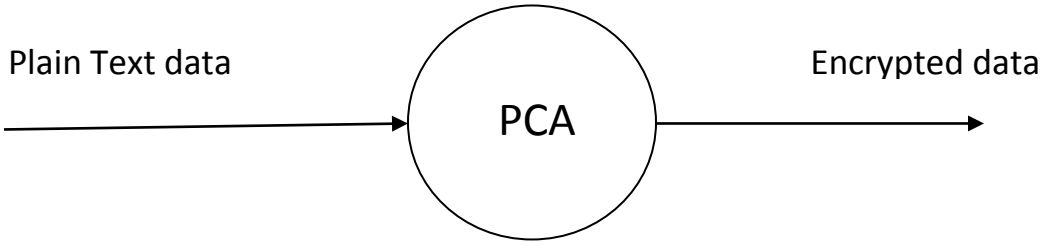
process is implemented for decryption and for the first time Parallel key encryption algorithm is extended for CBC and substantial reduction in time is being recorded experimentally. Moreover, with an increase in key size it further accelerates the entire encryption and decryption process.

INTERLEAVED CIPHER BLOCK CHAINING:

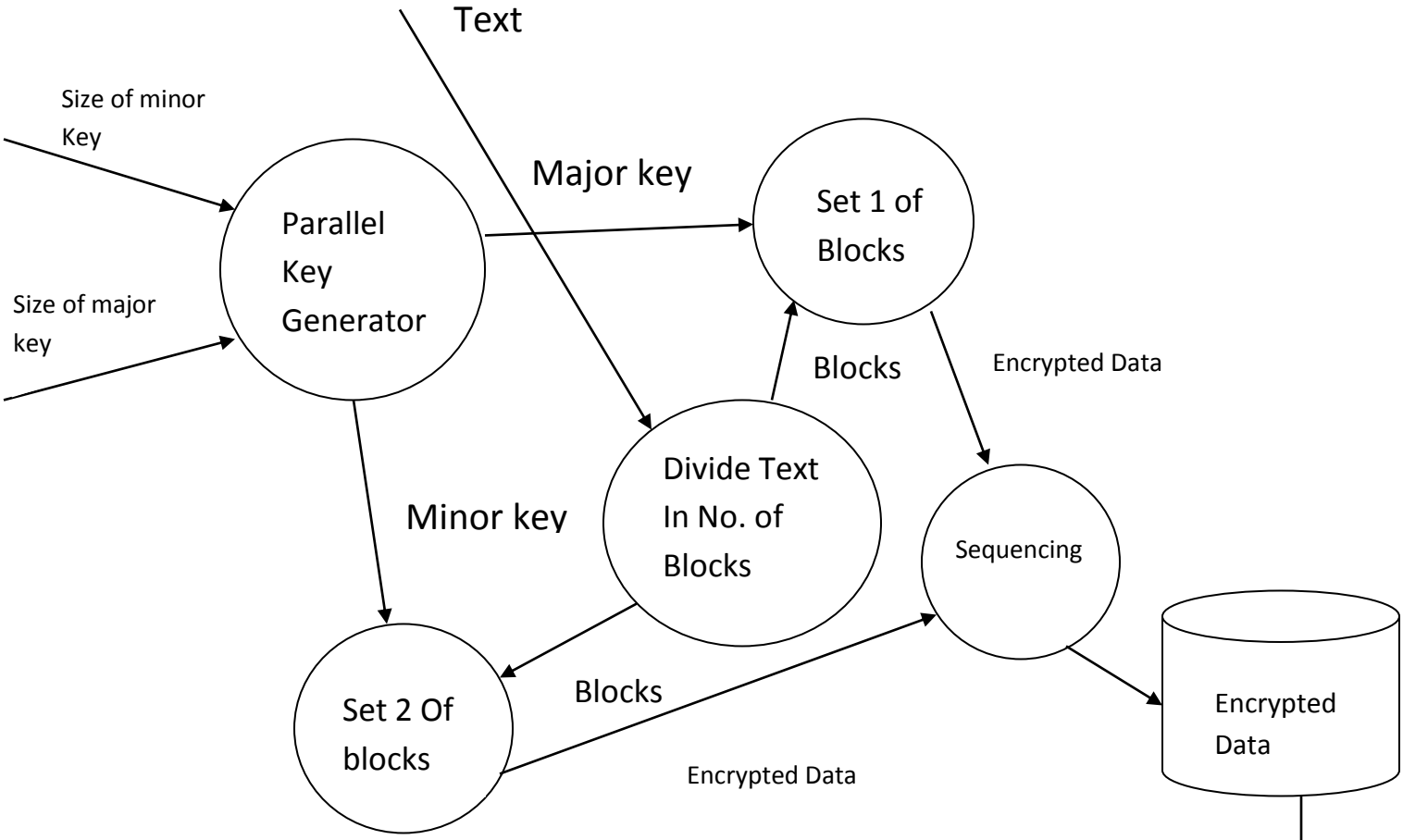
Here we propose a complete execution of cryptographic mode ICBC using parallel key encryption is done for the very first time. In ICBC, the division of blocks remains the same as in CBC but here the execution of blocks happens simultaneously thereby commendably reducing the time taken for the entire process execution. Here for the first time we have implemented ICBC with a key size range of 1024 and 2048 bit thereby surpassing all other cryptographic algorithms in terms of speed and security. The encryption and decryption process in ICBC are similar to cipher block chaining except here the blocks of data are interleaved and executed using PCA.

Data Flow Diagram:-

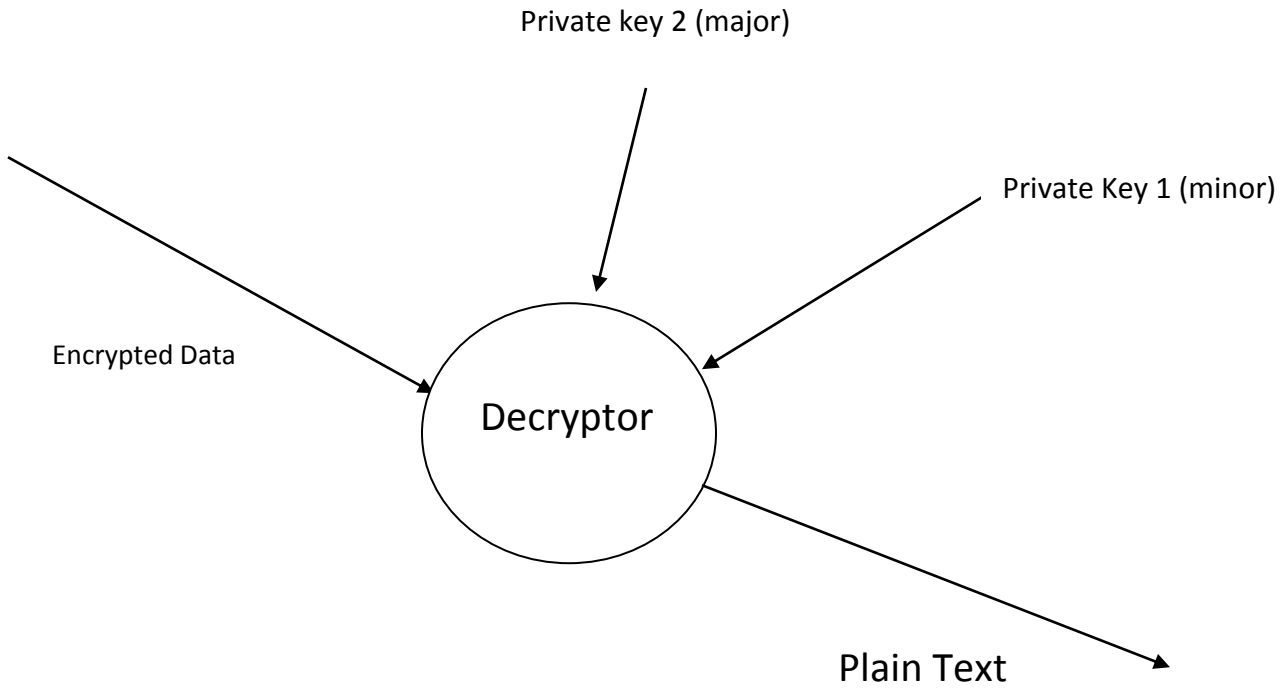
Data flow diagram



Level 0 DFD



Level 1 DFD



Reciever Side

Tools and Utilization Justification

Tools

i. Available tools (for this project).

- Microsoft Visual Studio 2005 Enterprise Edition
- My sql
- Adobe Photoshop CS4

ii. Tools comparison

Visual Studio:

Microsoft Visual Studio is an integrated development environment (IDE) from Microsoft. It can be used to develop console and graphical user interface applications along with Windows Forms applications, web sites, web applications, and web services in both native code together with managed code for all platforms supported by Microsoft Windows, Windows Mobile, Windows CE, .NET Framework, .NET Compact Framework and Microsoft Silverlight.

Visual Studio includes a code editor supporting IntelliSense as well as code refactoring. The integrated debugger works both as a source-level debugger and a machine-level debugger. Other built-in tools include a forms designer for building GUI applications, web

designer, class designer, and database schema designer. It accepts plug-ins that enhance the functionality at almost every level—including adding support for source-control systems (like Subversion and Visual SourceSafe) and adding new toolsets like editors and visual designers for domain-specific languages or toolsets for other aspects of the software development lifecycle (like the Team Foundation Server client: Team Explorer).

Visual Studio supports different programming languages by means of language services, which allow the code editor and debugger to support (to varying degrees) nearly any programming language, provided a language-specific service exists. Built-in languages include C/C++ (via Visual C++), VB.NET (via Visual Basic .NET), C# (via Visual C#), and F# (as of Visual Studio 2010. Support for other languages such as M, Python, and Ruby among others is available via language services installed separately. It also supports XML/XSLT, HTML/XHTML, JavaScript and CSS. Individual language-specific versions of Visual Studio also exist which provide more limited language services to the user: Microsoft Visual Basic, Visual J#, Visual C#, and Visual C++.

MYSQL :

MySQL is a relational database management system (RDBMS) that runs as a server providing multi-user access to a number of databases. MySQL is officially pronouncedy(S-Q-L") but is often also pronounced ("My Sequel"). It is named after developer Michael Widenius' daughter, My. The SQL phrase stands

for Structured Query Language. The MySQL development project has made its source code available under the terms of the GNU General Public License, as well as under a variety of proprietary agreements. MySQL was owned and sponsored by a single for-profit firm, the Swedish company MySQL AB, now owned by Oracle Corporation.Members of the MySQL community have created several forks (variations) such as Drizzle, OurDelta, Percona Server, and MariaDB. All of these forks were in progress before the Oracle acquisition; Drizzle was announced eight months before the Sun acquisition.

Free-software projects that require a full-featured database management system often use MySQL. Such projects include (for example) WordPress, phpBB, Drupal and other software built on the LAMP software stack. MySQL is also used in many high-profile, large-scale World Wide Web products, including Wikipedia.

Coding

Master Page

```
<%@ Master Language="C#" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
"http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head id="Head1" runat="server">
```

```
<style type="text/css">
    html
    {
        background-color: #E0FFFF;
        font: 14px Georgia,Serif;
    }
    .content
    {
        width: 900px;
        margin: auto;
        border-style: solid;
        background-color: White;
        padding: 10px;
    }
    .tabstrip
    {
        padding: 3px;
        border-top: solid 1px black;
        border-bottom: solid 1px black;
        width: 882px;
    }
    .tabstrip a
    {
        font: 14px Arial;
        color: Green;
        text-decoration: none;
    }
```

```
.tabstrip a:hover
{
   color: red;
   font: 16px Arial;
   letter-spacing: 1px;
}
a:active
{
   background-color: Silver;
   color: Blue;
}
.column
{
   float: left;
   padding: 10px;
   border-right: solid 1px black;
}
.rightColumn
{
   float: left;
   padding: 10px;
}
.clear
{
   clear: both;
}
.bottom
```

```
{
   border-top: solid 1px black;
}
.scroll a
{
   text-decoration: none;
   font: 14px Arial;
   color: Maroon;
}
.head
{
   font: 16px Arial;
   font-size: medium;
   text-align: center;
   font-weight: bold;
   color: Blue;
}
.detail
{
   text-decoration: none;
   font: 14px Arial;
   color: Black;
}
.style1
{
   width: 671px;
}
```

```
    </style>
    <title>Proruption Home Page</title>
    <asp:ContentPlaceHolder ID="head" runat="server">
    </asp:ContentPlaceHolder>
</head>
<body>
    <form id="form1" runat="server">
    <div class="content">
        <asp:Image ID="imgLogo" ImageUrl="~/Images/logo.jpg"
AlternateText="Website Logo"
            runat="server" Style="width: 200px; height: 100px" />
        <asp:Image ID="imgslogan" ImageUrl="~/Images/slogan.jpg"
AlternateText="Website Slogan"
            runat="server" Style="width: 690px; height: 100px" />
        <div class="tabstrip">
            <asp:HyperLink ID="linkHome" Text="Home"
NavigateUrl="~/Default.aspx" runat="server" /> |<asp:HyperLink
                ID="linkProjects" Text="RSA" NavigateUrl="~/Projects.aspx"
runat="server" /> |<asp:HyperLink
                    ID="linkTraining" Text="PCA" NavigateUrl="~/Training.aspx"
runat="server" /> |<asp:HyperLink
                        ID="linkTeam" Text="Team" NavigateUrl="~/Team.aspx"
runat="server" />
             </div>
        <table cellspacing="10px" align="left">
            <tr>
                <td style="width: 200px; height: 200px">
```

```
            <img height="34" alt="PTES, Jabalpur" title="PTES, Jabalpur"
src="images/alerts1.jpg"
                usemap="#Map2" style="width: 200px" /><br />
            <marquee class="scroll" onmouseover="this.stop();"
onmouseout="this.start();" scrollamount="3"
                scrolldelay="0" direction="up">
                <ul >
                    <li><a href="Training.aspx">Registration open for six months
industrial training (session - january, 2011 - june, 2011)</a></li>
                    <li><a href="Training.aspx">Six Weeks/Months Live Project
based Industrial Training for B.Tech. (CSE/ IT/ ECE) & MCA Students</a></li>
                    <li><a href="Training.aspx">Industrial Training on latest
technologies .Net, Java  , Database, Networking (LAN & WAN), Web Designing,
PHP, Embedded,  Telecom System & IT Security & Ethical Hacking</a></li>
                    <li><a href="Training.aspx">Proposal to Engineering Colleges for
In-house Campus Training for their Students</a></li>
                </ul></marquee>
            <img alt="" src="images/left_base.jpg" height="33" id="Img1"
width="200px" />
        </td>
        <td rowspan="2" class="style1">
            <asp:ContentPlaceHolder ID="ContentPlaceHolder1" runat="server">
            </asp:ContentPlaceHolder>
        </td>
    </tr>
    <tr>
        <td>
```

```
            </td>
          </tr>
        </table>
        <br class="clear" />
        <div class="bottom">
          <center>
            Website Visited:&amp;nbsp;Website Visited:&amp;nbsp;<img
src="http://www.easycounter.com/counter.php?dhimanmanoj"
              alt="Free Hit Counter" />
            <br />
            Copyright &amp;copy; to PTES, Jabalpur. All Right Reserved. Designed &
Managed by Group
            no. 9, IT-VIII sem<br />
            <br />
          </center>
        </div>
    </div>
    </form>
</body>
</html>
```

Home Page

```
<%@ Page Language="C#" MasterPageFile="~/MasterPage.master"
AutoEventWireup="true"
    CodeFile="Default.aspx.cs" Inherits="_Default" Title="Home Page" %>

<asp:Content ID="Content1" ContentPlaceHolderID="ContentPlaceHolder1"
runat="Server">
    <p class="head">
        PRORUPTION TECHNICAL EDUCATION SOCIETY
    </p>
    <p class="detail">
        PTES is one of the leading Software, Embedded System, Networking &amp;amp;
Telecommunication,
        Ethical Hacking Education Service Provider with an extensive experience in
designing
        and development at cutting edge IT &amp;amp; Telecom solutions. We are a group
of highly
        motivated IT &amp;amp; Telecom professionals with an ability to innovate a strong
desire
        to excel and successfully running its training. Registered under the the Societies
        Act, 1973. PTES was set up for providing Indian telecom expertise in all fields
        of telecom and IT for human welfare and education to young students who are
the
        future of our country. Society's core competence is in the fields of Switching,
        Transmission Systems, Cellular services, Rural Telecommunication, IT &
Networking
```

Solutions, Application Software, e-Governence, 3G Network, WIMAX Technology projects.

```
</p>
<br />
<p class="head">
    MISSION
</p>
<p class="detail">
    &#8220;A successful organization knows that it takes more than a good plan to succeed
    in business. It takes an empowered organization, focused on realistic goals, with
    impassioned leadership. It takes vision. It takes consensus. It takes a sense of
    purpose! &#8221;.
</p>
<p class="detail">
    Our Mission is to make every child of india innovative through the medium of education
    and training. Our aim is to provide <b>TRUST</b> (Technical Research And Upturn
    in Scintific Tasks) to India and poor students.</p>
<br />
<br />
</asp:Content>
```

RSA

User interface

```
<%@ Page Language="C#" MasterPageFile="~/MasterPage.master"
AutoEventWireup="true" CodeFile="Projects.aspx.cs" Inherits="Projects"
Title="RSA Implementation" %>

<asp:Content ID="Content1" ContentPlaceHolderID="head" Runat="Server">
</asp:Content>
<asp:Content ID="Content2" ContentPlaceHolderID="ContentPlaceHolder1"
Runat="Server">
<div>
    <center>
        <asp:Label ID="Label1" runat="server" Text="Enter text to
encrypt"></asp:Label>
        <asp:TextBox ID="txt1" TextMode="MultiLine" runat="server" Rows="10"
Columns="20"
            Height="115px" Style="margin-bottom: 0px"
Width="660px"></asp:TextBox>
        <br />
        <asp:Button ID="Assignkey" runat="server" Text="AssignKey"
OnClick="Assignkey_Click" />
        <asp:Button ID="Encrypt" runat="server" Text="Encrypt"
OnClick="Encrypt_Click" />
        <br />
```

```
        <asp:Label ID="Label2" runat="server" Text="Encrypted text is -
"></asp:Label>
        <br />
        <asp:TextBox ID="txt2" TextMode="MultiLine" runat="server" Rows="10"
Columns="20"
            Height="115px" Style="margin-bottom: 0px"
Width="660px"></asp:TextBox>
        <br />
        <asp:Button ID="Decrypt" runat="server" Text="Decrypt"
OnClick="Decrypt_Click" />
        <br />
        <asp:Label ID="Label3" runat="server" Text="Text after Decryption -
"></asp:Label>
        <br />
        <asp:TextBox ID="txt3" TextMode="MultiLine" runat="server" Rows="10"
Columns="20"
            Height="115px" Style="margin-bottom: 0px" Width="660px"
            ></asp:TextBox>
      </center>
   </div>

</asp:Content>
```

Coding:

```
using System;
using System.IO;
using System.Configuration;
using System.Data;
using System.Linq;
using System.Web;
using System.Web.Security;
using System.Web.UI;
using System.Web.UI.HtmlControls;
using System.Web.UI.WebControls;
using System.Web.UI.WebControls.WebParts;
using System.Xml.Linq;
using System.Security.Cryptography;

public partial class Projects : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {

    }
    protected void Assignkey_Click(object sender, EventArgs e)
    {
        Cryptography.AssignNewKey();
    }
    protected void Encrypt_Click(object sender, EventArgs e)
```

```
    {
        txt2.Text = Cryptography.EncryptData(txt1.Text);
    }
    protected void Decrypt_Click(object sender, EventArgs e)
    {
        txt3.Text = Cryptography.DecryptData(txt2.Text);
    }
}
public class Cryptography
{
    public static RSACryptoServiceProvider rsa;
    public static void AssignParameter()
    {
        const int PROVIDER_RSA_FULL = 1;
        const string CONTAINER_NAME = "SpiderContainer";
        CspParameters cspParams;
        cspParams = new CspParameters(PROVIDER_RSA_FULL);
        cspParams.KeyContainerName = CONTAINER_NAME;
        cspParams.Flags = CspProviderFlags.UseMachineKeyStore;
        cspParams.ProviderName = "Microsoft Strong Cryptographic Provider";
        rsa = new RSACryptoServiceProvider(cspParams);
    }
    public static string EncryptData(string data2Encrypt)
    {
        AssignParameter();
        StreamReader reader = new StreamReader(@"E:\Study material\.net
projects\Proruption2 file input\publickey.xml");
```

```
        string publicOnlyKeyXML = reader.ReadToEnd();
        rsa.FromXmlString(publicOnlyKeyXML);
        reader.Close();         //read plaintext, encrypt it to ciphertext
        byte[] plainbytes = System.Text.Encoding.UTF8.GetBytes(data2Encrypt);
        byte[] cipherbytes = rsa.Encrypt(plainbytes, false);
        return Convert.ToBase64String(cipherbytes);
    }
    public static void AssignNewKey()
    {
        AssignParameter();
        RSACryptoServiceProvider rsa = new RSACryptoServiceProvider(1024);
        if (File.Exists(@"E:\Study material\.net projects\Proruption2 file
input\privatekey.xml") == true)
            File.Delete(@"E:\Study material\.net projects\Proruption2 file
input\privatekey.xml");
        if (File.Exists(@"E:\Study material\.net projects\Proruption2 file
input\publickey.xml") == true)
            File.Delete(@"E:\Study material\.net projects\Proruption2 file
input\publickey.xml");
        //provide public and private RSA params
        FileStream fs1 = new FileStream(@"E:\Study material\.net
projects\Proruption2 file input\privatekey.xml", FileMode.CreateNew,
FileAccess.ReadWrite);
        StreamWriter sw1 = new StreamWriter(fs1);
        string publicPrivateKeyXML = rsa.ToXmlString(true);
        sw1.Write(publicPrivateKeyXML);
        sw1.Close(); //provide public only RSA params
```

```
        fs1.Close();
        FileStream fs2 = new FileStream(@"E:\Study material\.net
projects\Proruption2 file input\publickey.xml", FileMode.CreateNew,
FileAccess.ReadWrite);
        StreamWriter sw2 = new StreamWriter(fs2);
        string publicOnlyKeyXML = rsa.ToXmlString(false);
        sw2.Write(publicOnlyKeyXML);
        sw2.Close();
        fs2.Close();
    }
    public static string DecryptData(string data2Decrypt)
    {
        AssignParameter();
        byte[] getpassword = Convert.FromBase64String(data2Decrypt);
        StreamReader reader = new StreamReader(@"E:\Study material\.net
projects\Proruption2 file input\privatekey.xml");
        string publicPrivateKeyXML = reader.ReadToEnd();
        rsa.FromXmlString(publicPrivateKeyXML);
        reader.Close();                         //read ciphertext, decrypt it to plaintext

        byte[] plain = rsa.Decrypt(getpassword, false);
        return System.Text.Encoding.UTF8.GetString(plain);
    }
}
```

Input text :-

The excellent Mr. Morris was an Englishman, and he lived in the days of Queen Victoria the Good.

Output text :-

icpeYzStxHAVGkDf097dBW62dp4HzbkIehkF83VPY/9NTlPtekapebCc9Zcg0IijoADn/hew2O04mKJxD81kvbVkHx/dU4nxJly7PF3MlQZEiGLOg0+OjxbFnt4wloFk2o3T1xRWPjMVU4XpdFx6F3hW073oK3i2TdUUCpGds/w=

Public key

<RSAKeyValue>

<Modulus>

p7cq/5YopTr9Qwrmi9GLSViYtS0CnJpr4HOq0wUbjaqv4DaAdbFSZbnW9FLNVovB5EOqUDXQARQe9hW66BvWLuoe7ZlCBeBE5v/1XJKL2GqXOMApC+kr7q+F53/jA+wxE59facvyi/JfojbTSOUwQDkGq1IyU3QwgrjtAF1Ocks=

</Modulus>

<Exponent>

AQAB

</Exponent>

</RSAKeyValue>

Private Key :-

<RSAKeyValue>

<Modulus>

p7cq/5YopTr9Qwrmi9GLSViYtS0CnJpr4HOq0wUbjaqv4DaAdbFSZbnW9FLNVo
vB5EOqUDXQARQe9hW66BvWLuoe7ZlCBeBE5v/1XJKL2GqXOMApC+kr7q+
F53/jA+wxE59facvyi/JfojbTSOUwQDkGq1IyU3QwgrjtAF1Ocks=

</Modulus>

<Exponent>

AQAB

</Exponent>

<P>

1fXls4hQs5iMs3IKE2+YjrVHePGinGPKvjUdjch5Cv+ZZaxJ2TAqlXZx13K5b3pHK59
XGEgYPcbSC7ScEBsD8Q==

</P>

<Q>

yKsrbGPPdOq+1T/FG1J5dkDEsZRIwfn9rAOjyrfqU2/xo94bnIx3n1CnAPfn/sPZrMM+/
da6LW2W3q8fO/vl+w==

</Q>

<DP>

HO9vTBNQTql/xbjTcgNyj0Im001iX2IDhiuqvoTxEbK1bRGavxuLdsEY7jkeqNvRii23
WppNZPVhbKHVRTxSEQ==

</DP>

<DQ>

d+X+XhcM/7i7KUOfDAItXMC6Rbhp+KzcKwlJtnhJZG1Anj65lMJBP0m7X1LApXJz/
0vxuU2xlC8ktA1Cxl5uPQ==

</DQ>

<InverseQ>

BWQbWWizwOwX5N3TyURLMiy+JJGSUv3oni2OmuTPpOBigI6cldehuLRRoexfS89
amviz6WOvxKQ8w0LxxB+K2A==

</InverseQ>

<D>

RBEczFo+6MBGJT15T+mdZAknAPDhjHlHqvKGs9nw0tAhTRgRjrKGxG5mBQbN
+4ABEwk7SA5u+uEVp74t+yHBDOUEkcRUzkSDuPrwWaPUMkbw4M1iRUTQY
cvigceeSZb+otZce8QZYWHnVGkShFTL81wfe9bD+6yP1FF0eKmCJkE=

</D>

</RSAKeyValue>

PCA

User interface:

```
<%@ Page Language="C#" MasterPageFile="~/MasterPage.master"
AutoEventWireup="true" CodeFile="Training.aspx.cs" Inherits="Training"
Title="PCA Implementation" %>

<asp:Content ID="Content1" ContentPlaceHolderID="head" Runat="Server">
</asp:Content>
<asp:Content ID="Content2" ContentPlaceHolderID="ContentPlaceHolder1"
Runat="Server">
<div>
    <center>
        enter text file to encrypt
        <asp:FileUpload ID="FileUpload1" runat="server" />
        <br />
         <asp:Button ID="Encrypt" runat="server" Text="Upload and
Encrypt"
            OnClick="Encrypt_Click" />
        <br />
        <br />
        <asp:Label ID="Label1" runat="server" Enabled="false"
Text=""></asp:Label>
        <br />
```

```
        <asp:TextBox ID="txt2" TextMode="MultiLine" runat="server" Rows="10"
Columns="20"
            Height="115px" Style="margin-bottom: 0px"
Width="660px"></asp:TextBox>
        <br />
        <asp:Button ID="saveEncryptedText" runat="server" Text="Save in a file"
            onclick="saveEncryptedText_Click" />
        <br />
        <asp:Label ID="Label2" runat="server" Text="Encrypted text is saved in file
-" Enabled="false"></asp:Label>
        <br />
        <asp:Button ID="Decrypt" runat="server" Text="Decrypt"
OnClick="Decrypt_Click" />
        <br />
        <asp:Label ID="Label3" runat="server" Text="Text after Decryption -
"></asp:Label>
        <br />
        <asp:TextBox ID="txt3" TextMode="MultiLine" runat="server" Rows="10"
Columns="20"
            Height="115px" Style="margin-bottom: 0px" Width="660px"
            ></asp:TextBox>
      </center>
   </div>
</asp:Content>
```

Coding:

```
using System;
using System.IO;
using System.Configuration;
using System.Data;
using System.Linq;
using System.Web;
using System.Web.Security;
using System.Web.UI;
using System.Web.UI.HtmlControls;
using System.Web.UI.WebControls;
using System.Web.UI.WebControls.WebParts;
using System.Xml.Linq;
using System.Security.Cryptography;
public partial class Training : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
        Label2.Enabled = false;
    }
    protected void Encrypt_Click(object sender, EventArgs e)
    {
        cryptography1.AssignNewKey();
        cryptography2.AssignNewKey();
        string TextToEncrypt="";
```

```
        string encryptedText = "";
        string[] datablocks=new string[1000];
        if (FileUpload1.HasFile && Path.GetExtension(FileUpload1.FileName) ==
".txt")
        {
            FileUpload1.PostedFile.SaveAs(@"D:\Uploads\" + FileUpload1.FileName);
            StreamReader fileReader = new StreamReader(new
FileStream(@"D:\Uploads\" + FileUpload1.FileName, FileMode.Open,
FileAccess.Read));
            TextToEncrypt = fileReader.ReadToEnd();
            string text = TextToEncrypt;
            int i = 0, offset = 0;
            while (offset < (text.Length))
            {
               if (i % 2 == 0)
               {
                  if ((offset + 64) < text.Length)
                  {
                     datablocks[i] = text.Substring(offset, 64);
                  }
                  else datablocks[i] = text.Substring(offset, text.Length - offset);
                  offset += 64;
                  if(i==0)
                  encryptedText = encryptedText +
cryptography1.EncryptData(datablocks[i]);
                  else
```

```
                encryptedText = encryptedText + '|' +
cryptography1.EncryptData(datablocks[i]);
            }
            else
            {
               if ((offset + 128) < text.Length)
               {
                  datablocks[i] = text.Substring(offset, 128);
               }
               else datablocks[i] = text.Substring(offset, text.Length - offset);
               offset += 128;
               encryptedText = encryptedText + '|' +
cryptography2.EncryptData(datablocks[i]);
            }
            i++;
         }
      }
      else
      {
         Label1.Text = "No text file received.";
         return;
      }
      txt2.Text = encryptedText;
   }
   protected void Decrypt_Click(object sender, EventArgs e)
   {
      string textToDecrypt = txt2.Text;
```

```
        string decryptedData = "";
        string[] encryptedBlocks = new string[1000];
        int i = 0, j = 0,k=0;
        while (k < textToDecrypt.Length)
        {
            while (textToDecrypt[i] != '|')
            {
                encryptedBlocks[j] = encryptedBlocks[j] + textToDecrypt[i];
                i++;
                if (i >= textToDecrypt.Length) break;
                k++;
            }
            if (j % 2 == 0)
            {
                decryptedData = decryptedData +
cryptography1.DecryptData(encryptedBlocks[j]);
            }
            else
            {
                decryptedData = decryptedData +
cryptography2.DecryptData(encryptedBlocks[j]);
            }
            j++;
            i++;
            k++;
        }
        txt3.Text = decryptedData;
```

```
    }
    public class cryptography2
    {
        public static RSACryptoServiceProvider rsa;
        public static void AssignParameter()
        {
            const int PROVIDER_RSA_FULL = 1;
            const string CONTAINER_NAME = "SpiderContainer";
            CspParameters cspParams;
            cspParams = new CspParameters(PROVIDER_RSA_FULL);
            cspParams.KeyContainerName = CONTAINER_NAME;
            cspParams.Flags = CspProviderFlags.UseMachineKeyStore;
            cspParams.ProviderName = "Microsoft Strong Cryptographic Provider";
            rsa = new RSACryptoServiceProvider(cspParams);
        }
        public static string EncryptData(string data2Encrypt)
        {
            AssignParameter();
            StreamReader reader = new StreamReader(@"E:\Study material\.net
projects\Proruption2 file input\privatekey2.xml");
            string publicOnlyKeyXML = reader.ReadToEnd();
            rsa.FromXmlString(publicOnlyKeyXML);
            reader.Close();     //read plaintext, encrypt it to ciphertext
            byte[] plainbytes = System.Text.Encoding.UTF8.GetBytes(data2Encrypt);
            byte[] cipherbytes = rsa.Encrypt(plainbytes, false);
            return Convert.ToBase64String(cipherbytes);
        }
```

```
public static void AssignNewKey()
{
    AssignParameter();
    RSACryptoServiceProvider rsa = new RSACryptoServiceProvider(2048);
    if (File.Exists(@"E:\Study material\.net projects\Proruption2 file
input\privatekey2.xml") == true)
        File.Delete(@"E:\Study material\.net projects\Proruption2 file
input\privatekey2.xml");
    if (File.Exists(@"E:\Study material\.net projects\Proruption2 file
input\publickey2.xml") == true)
        File.Delete(@"E:\Study material\.net projects\Proruption2 file
input\publickey2.xml");
    //provide public and private RSA params
    FileStream fs1 = new FileStream(@"E:\Study material\.net
projects\Proruption2 file input\privatekey2.xml", FileMode.CreateNew,
FileAccess.ReadWrite);
    StreamWriter sw1 = new StreamWriter(fs1);
    string publicprivatekeyXML = rsa.ToXmlString(true);
    sw1.Write(publicprivatekeyXML);
    sw1.Close(); //provide public only RSA params
    fs1.Close();
    FileStream fs2 = new FileStream(@"E:\Study material\.net
projects\Proruption2 file input\publickey2.xml", FileMode.CreateNew,
FileAccess.ReadWrite);
    StreamWriter sw2 = new StreamWriter(fs2);
    string publicOnlyKeyXML = rsa.ToXmlString(false);
    sw2.Write(publicOnlyKeyXML);
```

```
            sw2.Close();
            fs2.Close();
        }
        public static string DecryptData(string data2Decrypt)
        {
            AssignParameter();
            byte[] getpassword = Convert.FromBase64String(data2Decrypt);
            StreamReader reader = new StreamReader(@"E:\Study material\.net
projects\Proruption2 file input\privatekey2.xml");
            string publicprivatekeyXML = reader.ReadToEnd();
            rsa.FromXmlString(publicprivatekeyXML);
            reader.Close();                         //read ciphertext, decrypt it to plaintext

            byte[] plain = rsa.Decrypt(getpassword, false);
            return System.Text.Encoding.UTF8.GetString(plain);
        }
    }
    public class cryptography1
    {
        public static RSACryptoServiceProvider rsa;
        public static void AssignParameter()
        {
            const int PROVIDER_RSA_FULL = 1;
            const string CONTAINER_NAME = "SpiderContainer";
            CspParameters cspParams;
            cspParams = new CspParameters(PROVIDER_RSA_FULL);
            cspParams.KeyContainerName = CONTAINER_NAME;
```

```
            cspParams.Flags = CspProviderFlags.UseMachineKeyStore;
            cspParams.ProviderName = "Microsoft Strong Cryptographic Provider";
            rsa = new RSACryptoServiceProvider(cspParams);
        }
        public static string EncryptData(string data2Encrypt)
        {
            AssignParameter();
            StreamReader reader = new StreamReader(@"E:\Study material\.net
projects\Proruption2 file input\publickey1.xml");
            string publicOnlyKeyXML = reader.ReadToEnd();
            rsa.FromXmlString(publicOnlyKeyXML);
            reader.Close();     //read plaintext, encrypt it to ciphertext
            byte[] plainbytes = System.Text.Encoding.UTF8.GetBytes(data2Encrypt);
            byte[] cipherbytes = rsa.Encrypt(plainbytes, false);
            return Convert.ToBase64String(cipherbytes);
        }
        public static void AssignNewKey()
        {
            AssignParameter();
            RSACryptoServiceProvider rsa = new RSACryptoServiceProvider(1024);
            if (File.Exists(@"E:\Study material\.net projects\Proruption2 file
input\privatekey1.xml") == true)
                File.Delete(@"E:\Study material\.net projects\Proruption2 file
input\privatekey1.xml");
            if (File.Exists(@"E:\Study material\.net projects\Proruption2 file
input\publickey1.xml") == true)
```

```
            File.Delete(@"E:\Study material\.net projects\Proruption2 file input\publickey1.xml");
            //provide public and private RSA params
            FileStream fs1 = new FileStream(@"E:\Study material\.net projects\Proruption2 file input\privatekey1.xml", FileMode.CreateNew, FileAccess.ReadWrite);
            StreamWriter sw1 = new StreamWriter(fs1);
            string publicprivatekeyXML = rsa.ToXmlString(true);
            sw1.Write(publicprivatekeyXML);
            sw1.Close(); //provide public only RSA params
            fs1.Close();
            FileStream fs2 = new FileStream(@"E:\Study material\.net projects\Proruption2 file input\publickey1.xml", FileMode.CreateNew, FileAccess.ReadWrite);
            StreamWriter sw2 = new StreamWriter(fs2);
            string publicOnlyKeyXML = rsa.ToXmlString(false);
            sw2.Write(publicOnlyKeyXML);
            sw2.Close();
            fs2.Close();
        }
        public static string DecryptData(string data2Decrypt)
        {
            AssignParameter();
            byte[] getpassword = Convert.FromBase64String(data2Decrypt);
            StreamReader reader = new StreamReader(@"E:\Study material\.net projects\Proruption2 file input\privatekey1.xml");
            string publicprivatekeyXML = reader.ReadToEnd();
```

```
            rsa.FromXmlString(publicprivatekeyXML);
            reader.Close();                    //read ciphertext, decrypt it to plaintext

            byte[] plain = rsa.Decrypt(getpassword, false);
            return System.Text.Encoding.UTF8.GetString(plain);
        }
    }
    protected void saveEncryptedText_Click(object sender, EventArgs e)
    {
        if (File.Exists(@"D:\Uploads\encrypted\encrypted.txt") == true)
            File.Delete(@"D:\Uploads\encrypted\encrypted.txt");
        FileStream fs=new
FileStream(@"D:\Uploads\encrypted\encrypted.txt",FileMode.CreateNew,FileAcces
s.Write);
        StreamWriter fileWriter = new StreamWriter(fs);
        fileWriter.Write(txt2.Text);
        fileWriter.Close();
        Label2.Enabled = true;
        Label2.Text = Label2.Text + " " + @"D:\Uploads\encrypted\encrypted.txt";
    }
}
```

Input file:-

```
clear all;
clc;
x1=[1 1 -1 -1];
x2=[1 -1 1 -1];
x3=[1 1 1 1];
t=[-1 1 -1 -1];
w1=0.1;w2=0.1;b=0.1;
eta=0.1;
dw1=0;dw2=0;db=0;
e=2;
epoch=0;
while(e>1.018)
        epoch=epoch+1;
      e=0;
      for i=1:4
          nety(i) =w1*x1(i)+w2*x2(i)+b;
          nt=[ nety(i) t(i)];
          dw1=eta*(t(i)-nety(i))*(x1(i));
          dw2=eta*(t(i)-nety(i))*(x2(i));
          db= eta*(t(i)-nety(i));
          wc=[dw1 dw2 db];
          w1=w1+dw1;
          w2=w2+dw2;
          b=b+db;
          w=[w1 w2 b];
```

```
            x=[x1(i) x2(i) x3(i)];
           pnt=[x nt wc w]
       end
    for i=1:4
        nety(i) =w1*x1(i)+w2*x2(i)+b;
        e=e+(t(i)-nety(i))^2;
    end
end
```

Output :-

aQ2XJbATWbKr9Lm/sSGNy9XECOB9v7x3oYHailzUF/wmtneT3vvK/5T9U1Cvp
i4fbUf9+TEddOkTIiS3pzpcgxdd3sBLVh1KcCmf6VeLt2BQ7iSF+v4f/XHv0RAFLi
EAy0S7NeQ8qNadTUzvEvu86lzGMexZD5VbWNKNrf0+3s4=|l0lxo7bXiVF+aeFP
XF8zYmaVG8VsmhKOYKxx6I7Nqv4cXXrYnny32coJwCx8GUY5ZbMutwQ/w/J
eLiUFySL5JHAOrJMmDN1E62W9FcIXqA3CIkEvV+qfgyBc9KB5dtkAcSHQ/I2L
53qX6gXLnZtiDD/wPq6yNaoMir9O0DE9rhwpTUdPAYn5MxCP5JkXIRUsKNop
EnjOZLf/vZ/j3SbFwlwuNIZz0JOyUNX0AMAPaYALGV7v4Sy3Yy52LH/NhJwH
VtlLJ9gwhqlhNy82STbDz7kEwfsd7TkE8QlzT9R5A6FhltsXcNU5jWr5jcG9q91ht
X5DRWoZv8PmvNxW4/o9aQ==|jHvfIsVQEKwkdqpm440PgmaCh797gcl4rYIsMe
4oVMjz9m0aCe3RasFThj6M2suDg5jNEsI56o4+e6t/jYpNnflbC8dIEJevInTDFFV1
Ig8Ms3KSS/YLI08H7RKxVwLDR3nnGUBKVCPHlUiTV/jjczFyaqN+9TnNRjgoY
qv+tWw=|TIgBq4NMM5V4KehljzDRv2noNGdnZo5vivdrf8GwKDOS6bGe1IjWO
OSDl82oK4/RqaUcDgVI5vEKI2If7rR+L3HqgLG3yKI8SH4LRmyPpBkBBTDaA9
cVa3yGaPc3I92LIPwWGOLB48IRTiCqMVO61iaNTvddEmCQgW/Gjhcbm3O/D
NULDwF1DAy6lpIekiasp7WYe2Cr16RCwBtVpjD7WpR+1Do9YqEdq97S0aw9/d
BFnqIp3/S+tnWz+hDDdN89ji1UTZioOcVMe0Z1FtAfDsPESiuifveIrQmQcBIdRL

4ny6yLB909VSsSgIz7F5oYyp6A9X/F1qiQxzsEFvhogg==|TM8nbEMo3mwnDVtb
zL5j2D7k1xcyukurJ9L7DEkS5aHM/f1hqk88Kov4FAp7i8H3ffFgXFSlCm9EtKWU
eoYweZxVz9+xsdsEH8jKnMZ1lNrg4lXeyOVbUT4DYVp+nDkCMbQy5DWKign
71lwgOPG9S/2qAwehjjI7kCZx1Hs/tCY=|JTs/ROi8QQbdfHwJXdStccPiK//0w99v
XWmQ0OYdyT4QgzwkiBUk2RIyfkDexBC2ZYl2qsPaZ48DMrTwAFcUt2lN9yeI
wDiJZBreAoX0gsRUZDlYk5r1wezSbw2zl+0rK0biRvsxodHA8EZ0XPQm1IQpK4
I8Rv8HiswdP8NMZ/tqQZ2Sm/f3DA5S7PtJNNYLAXpcxnT4dm1G1YWoBFdxUB
6tOR/pmnfawmLFcVp+d/zcMa0rX/IglanF3NbOdJ1aoOdHeh63af5nhdBThry7lwX
1UgEF93//yEtAtTzhMzDxcGy74Hn6pc92NXHLbsu8mHX6PkBr7o3F7xcrG7qSEg
==|iVdNz5D8VFTfdAW7aEC/x7/4z9Xo4n5BeH8IavjyZ4whRDtPwSDe8ilVxj8AU
/6VgaDfEAxfTjptk9t1sr/WCmxdfmV6UdsIEEUEURXXGtMAsgeelrKuVehGsYpr
cmPfgo+e/LQr5a6n6DEUTjLRXTDa9aiT5v6/mBFzdbI+lCU=|poDM/Vs8lq0bmpk
S4/SlTMST/6XRoFPVGp0TZZFdi83wSIMKa8u5BVU1OzziuxD1ou6MTSq1kFGh
gh/yvesrzi/HvFhCov4UwygO9FPyOdU9iffKZErcqi3lNTnBj19xlN/vOBy++aJKaJ1
pLqz9ZtRLKthCpMlvhhL5JS9Yhhz1Jzrm/FZUOCZ9UqoHpy6f1jzIz0zX5wHtHHr
t3eIxCVmYJHmaQxGz4TbA4DPFkeKAwqA8vEWXewX4ADNn7WKOvazL0wV
ByOUFtdBwdEyHvezWqNwsUZ8aB2QBswrFDHBZ0oE/Am3Km4L/XxoEQNZ9n
gwM/9RDb/4pbCMk8TJhvQ==

Minor Key:-

<RSAKeyValue>
<Modulus>
o3dHcm3daKU0/3NCROgIVSv/2Gxhw+rW8lXYGEyBuZiq+OKwj5K
xwBpbU8yghgAkX8QYGQTEI9zSIS2N0AeVv8Dc5JQJFY2X0U6FB
QUKweEkIRdGyNTJSV1CZBaCXii3YzY2tX/UsNtoUd/qXSyR2joN6
7SimmIAJDsvlO9324s=

</Modulus>
<Exponent>
AQAB
</Exponent>
<P>
0Rw8yrBhIGpNWr0ZMTssN4bL+Jom9tjTFZKpFpAZfRAWnoEuNJS
CPGwcX318G7w60lZDESoEkwRiUiWp8DaHRw==
</P>
<Q>
yB7fmXdX8sTRLZSb8eMCpt0yT/8/8H221QHTJfZIsV20xywRxrY3h
D6IgLcGDPm5mYQRqT2KDNLY1lka665znQ==
</Q>
<DP>
hw28uY/W+w+M7DjLR3+XHDmkDOjKl40ML5esO1VQEPTOFwsJ
o43y20FbxdSOzn7wYp6x7Cj+EVG4iqOmsO0Wyw==
</DP>
<DQ>
lbysF/t5XhgwYAebK8FtQYeu1rapk/usqvMDl5sRR01TL9hksKe1IIldv
9myeFD+s8RjJpv6SKAXSUNA5moQ7Q==
</DQ>

<InverseQ>
e9WgmQpqgzlSBnqrbgVMEbe2q1ofGr5wfEqGrDpa1YOHixM/4EVtl
+5aXYInr4VUSiNiYzDtm4TBfOMvssjRjQ==
</InverseQ>

<D>
UuELLI1fkMsSj+IUW3N5phVk5ootF3Rd/whfRBKmQz+QkBf/hQvP
a6LxYw+A4+/2gUefIDA2cihnscKzACKYXY0ijakZDhZSNzIemgXm
OLNQT1WSuu/1GfQ/ECqJhDxmS77Dsv28Jbk9KhWMUrsmJuHhbk7
42QFcVK/RTe5ereE=
</D>
</RSAKeyValue>

Major Key

<RSAKeyValue>
<Modulus>
qz4oEIE3d8VObj87IB2zbZBC548YqK5tMXRR7t+SWrqKvjzsLkJ19
KQsFCRD85MgNwSAQNZLGI7Nv7Um/h2H2wYqewSCllywnuB/tva
U5BKvVGMbtKr2le4tlIBhCRWMH0vmmCLBm0HnbeDzkdZzp0rH6
Z6FT6QBybTAfrcCdquHyVmoMWGPQyNMZfwXYpIAQfIb8w1aE
BGj9R5cPEG6Cvx0HgS8t8oF/xahH5Iy8yRT5jgbVs0kZs7y9jey8lZDB
LmzvkLJ3smjd9GSn7vZCDlTVf4FGxDRn1sQpDtHKDUF9Er9xK+ly
gDRKjsvU9YwZP8K/KMeqcrKTXgVzLKHjQ==
</Modulus>

<Exponent>
AQAB
</Exponent>
<P>

4UvMwFt9fDDUnZEyQbtW1qSDR1q4Bc3Q3VHbplqbJJIMA7OFLib
aw/+M84T5j1SSkx1CcIan4HLAjOYbbnKJHYDC1Bn6C75ls9VpiO9/
GGMDI5g8XU3QLXeLXVbVFq3On88zqvE8NOLVPYdJMmSjp/0kj
XbXQN7zdYDS5QGWXRc=
</P>
<Q>
wpSHTtiVplPkOpu94bem2Asb0Bml6OyMP6I6XC9Ag9n6ppYa+qwi4
YSS4cpzZF67ygXdI/XUwsHR7iRBSgVPT43bNRDhzGQAji847Fl0yl
IoVeIlvASqMfI0LvMx7/VVxks0f/d0rP6cBa5sgysRRmnMculdBd+A
M17WRO0CDvs=
</Q>
<DP>
uxUrmEwg3sDZvWI1vnmneaOw087UYxVurGnuIKy6idqxGV24aQO
gxALwc8NnTIspD/rLEg/vVaVUmWv3OXbov1QN9P0BM0FOyM+dz
DU8b+7AS41+Cuizxx2u37pcRnHb077Go9ugaDY4ZthxguQpNjUyZjp
aMqcwudFYzwvLZc8=
</DP>
<DQ>
rVlJmnLWD9JZV+V8KuVEnOB4rNLMvlBPOjm7KHloTo8aAONlsc
E9ahCEEugcGWVUg8zpkKnoHII0FqbyC5HjcIp+qi4aG89Tt1hWrCm
eJMlA6hKJb5HwVoLi/pjm+lJlUj2Zzg+VQLGsKluUjN8HiuWioRVK
LHtLaXiqB8nmtZU=
</DQ>
<InverseQ>
AtMB40SWK4WmkI7uIVekK7E7nTqPTIW6wAy+IvVuFw+t0uWAS
7+VdhSfRxrFAM2VTOUVs51tPdvb/Bwznn9kH8qejOYIW6nDBPntc

+bi93OzdEcu73/a1bNzk6M6SvflJMF2keQ3QJ7yMLx7EPvdWjCgnw
U6zVtuN55c/TMwj3A=
</InverseQ>
<D>
D1MjsL5voVdV+oXMG8asrQxxAFoNAeOCQB/ByVfWuCnjirzXAhj
Oh5M9opvQPPcYtfPAKo1C8TKudQ6YK7um0dIOcS3pJd2rrLw4kb4l
eve1Gx83doaqRKapC5YlmPOFs7NOMqVBoExYkrvD8HQ5q2oE1AJ
sucDarBtx8qUcIlR34f2vQFQJZU0Pk59fODyUxDyLtW6rc2MRheUia
dW6k/g5FLnfhmVH6Rk3CiybKXJTgAgs4VK9DSreLIxPSDEEud6l+
EUJMyvNp8o6nO6OO8aMjZu5kRztoaCGSrBMadwgAhbLegalr6VR
NeXQDN5REpRdHLzKs/P7OOGAuEKr3Q==
</D>
</RSAKeyValue>

Public Key 1:-

<RSAKeyValue>
<Modulus>
o3dHcm3daKU0/3NCROgIVSv/2Gxhw+rW8lXYGEyBuZiq+OKwj5KxwBp
bU8yghgAkX8QYGQTEI9zSIS2N0AeVv8Dc5JQJFY2X0U6FBQUKweEkI
RdGyNTJSV1CZBaCXii3YzY2tX/UsNtoUd/qXSyR2joN67SimmIAJDsvlO9
324s=
</Modulus>
<Exponent>
AQAB
</Exponent>
</RSAKeyValue>

Public Key 2:-

<RSAKeyValue>

<Modulus>

qz4oEIE3d8VObj87IB2zbZBC548YqK5tMXRR7t+SWrqKvjzsLkJ19
KQsFCRD85MgNwSAQNZLGI7Nv7Um/h2H2wYqewSCllywnuB/tv
aU5BKvVGMbtKr2le4tlIBhCRWMH0vmmCLBm0HnbeDzkdZzp0rH6
Z6FT6QBybTAfrcCdquHyVmoMWGPQyNMZfwXYpIAQfIb8w1aEBG
j9R5cPEG6Cvx0HgS8t8oF/xahH5Iy8yRT5jgbVs0kZs7y9jey8lZDB
LmzvkLJ3smjd9GSn7vZCDlTVf4FGxDRn1sQpDtHKDUF9Er9xK+ly
gDRKjsvU9YwZP8K/KMeqcrKTXgVzLKHjQ==

</Modulus>

<Exponent>

AQAB

</Exponent>

</RSAKeyValue>

Testing

Testing techniques and testing strategies

Software testing is an investigation conducted to provide stakeholders with information about the quality of the product or service under test.[1] Software testing also provides an objective, independent view of the software to allow the business to appreciate and understand the risks of software implementation. Test techniques include, but are not limited to, the process of executing a program or application with the intent of finding software bugs (errors or other defects).

Software testing can also be stated as the process of validating and verifying that a software program/application/product:

1. meets the business and technical requirements that guided its design and development;
2. works as expected; and
3. can be implemented with the same characteristics.

Software testing, depending on the testing method employed, can be implemented at any time in the development process. However, most of the test effort occurs after the requirements have been defined and the coding process has been completed. As such, the methodology of the test is governed by the software development methodology adopted.

Unit Testing:

Unit testing deals with testing a unit as a whole. This would test the interaction of many functions but confine the test within one unit. The exact scope of a unit is left to interpretation. Supporting test code, sometimes called **scaffolding**, may be necessary to support an individual test. This type of testing is driven by the architecture and implementation teams. This focus is also called black-box testing because only the details of the interface are visible to the test. Limits that are global to a unit are tested here.

In the construction industry, scaffolding is a temporary, easy to assemble and disassemble, frame placed around a building to facilitate the construction of the building. The construction workers first build the scaffolding and then the building. Later the scaffolding is removed, exposing the completed building. Similarly, in software testing, one particular test may need some supporting software. This software establishes an environment around the test. Only when this environment is established can a correct evaluation of the test take place. The scaffolding software may establish state and values for data structures as well as providing dummy external functions for the test. Different scaffolding software may be needed from one test to another test. Scaffolding software rarely is considered part of the system.

Sometimes the scaffolding software becomes larger than the system software being tested. Usually the scaffolding software is not of the same quality as the system software and frequently is quite fragile. A small change in the test may lead to much larger changes in the scaffolding.

Internal and unit testing can be automated with the help of coverage tools. A coverage tool analyzes the source code and generates a test that will execute every alternative thread of execution. It is still up to the programmer to combine this test into meaningful cases to validate the result of each thread of execution. Typically, the coverage tool is used in a slightly different way. First the coverage tool is used to augment the source by placing informational prints after each line of code. Then the testing suite is executed generating an audit trail. This audit trail is analyzed and reports the percent of the total system code executed during the test suite. If

the coverage is high and the untested source lines are of low impact to the system's overall quality, then no more additional tests are required.

The primary goal of unit testing is to take the smallest piece of testable software in the application, isolate it from the remainder of the code, and determine whether it behaves exactly as you expect. Each unit is tested separately before integrating them into modules to test the interfaces between modules. Unit testing has proven its value in that a large percentage of defects are identified during its use.

The most common approach to unit testing requires drivers and stubs to be written. The driver simulates a calling unit and the stub simulates a called unit. The investment of developer time in this activity sometimes results in demoting unit testing to a lower level of priority and that is almost always a mistake. Even though the drivers and stubs cost time and money, unit testing provides some undeniable advantages. It allows for automation of the testing process, reduces difficulties of discovering errors contained in more complex pieces of the application, and test coverage is often enhanced because attention is given to each unit.

Integrated Testing:

Integration testing is a logical extension of unit testing. In its simplest form, two units that have already been tested are combined into a component and the interface between them is tested. A component, in this sense, refers to an integrated aggregate of more than one unit. In a realistic scenario, many units are combined into components, which are in turn aggregated into even larger parts of the program. The idea is to test combinations of pieces and eventually expand the process to test your modules with those of other groups. Eventually all the modules

making up a process are tested together. Beyond that, if the program is composed of more than one process, they should be tested in pairs rather than all at once.

Integration testing identifies problems that occur when units are combined. By using a test plan that requires you to test each unit and ensure the viability of each before combining units, you know that any errors discovered when combining units are likely related to the interface between units. This method reduces the number of possibilities to a far simpler level of analysis.

We can do integration testing in a variety of ways but the following are three common strategies:

- The top-down approach to integration testing requires the highest-level modules be test and integrated first. This allows high-level logic and data flow to be tested early in the process and it tends to minimize the need for drivers. However, the need for stubs complicates test management and low-level utilities are tested relatively late in the development cycle. Another disadvantage of top-down integration testing is its poor support for early release of limited functionality.

- The bottom-up approach requires the lowest-level units be tested and integrated first. These units are frequently referred to as utility modules. By using this approach, utility modules are tested early in the development process and the need for stubs is minimized. The downside, however, is that the need for drivers complicates test management and high-level logic and data flow are tested late. Like the top-down approach, the bottom-up

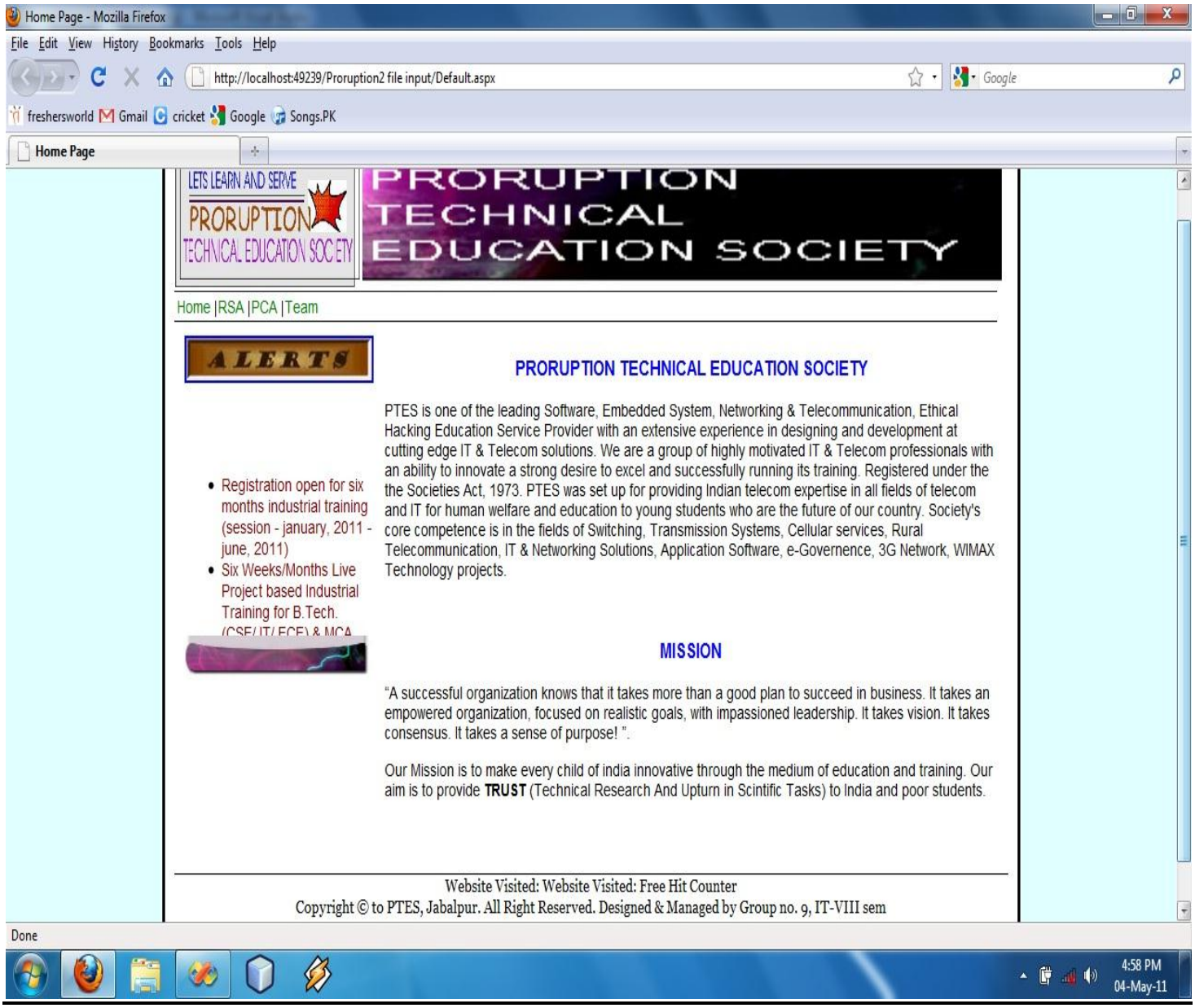
approach also provides poor support for early release of limited functionality.

- The third approach, sometimes referred to as the umbrella approach, requires testing along functional data and control-flow paths. First, the inputs for functions are integrated in the bottom-up pattern discussed above. The outputs for each function are then integrated in the top-down manner. The primary advantage of this approach is the degree of support for early release of limited functionality. It also helps minimize the need for stubs and drivers. The potential weaknesses of this approach are significant, however, in that it can be less systematic than the other two approaches, leading to the need for more regression testing.

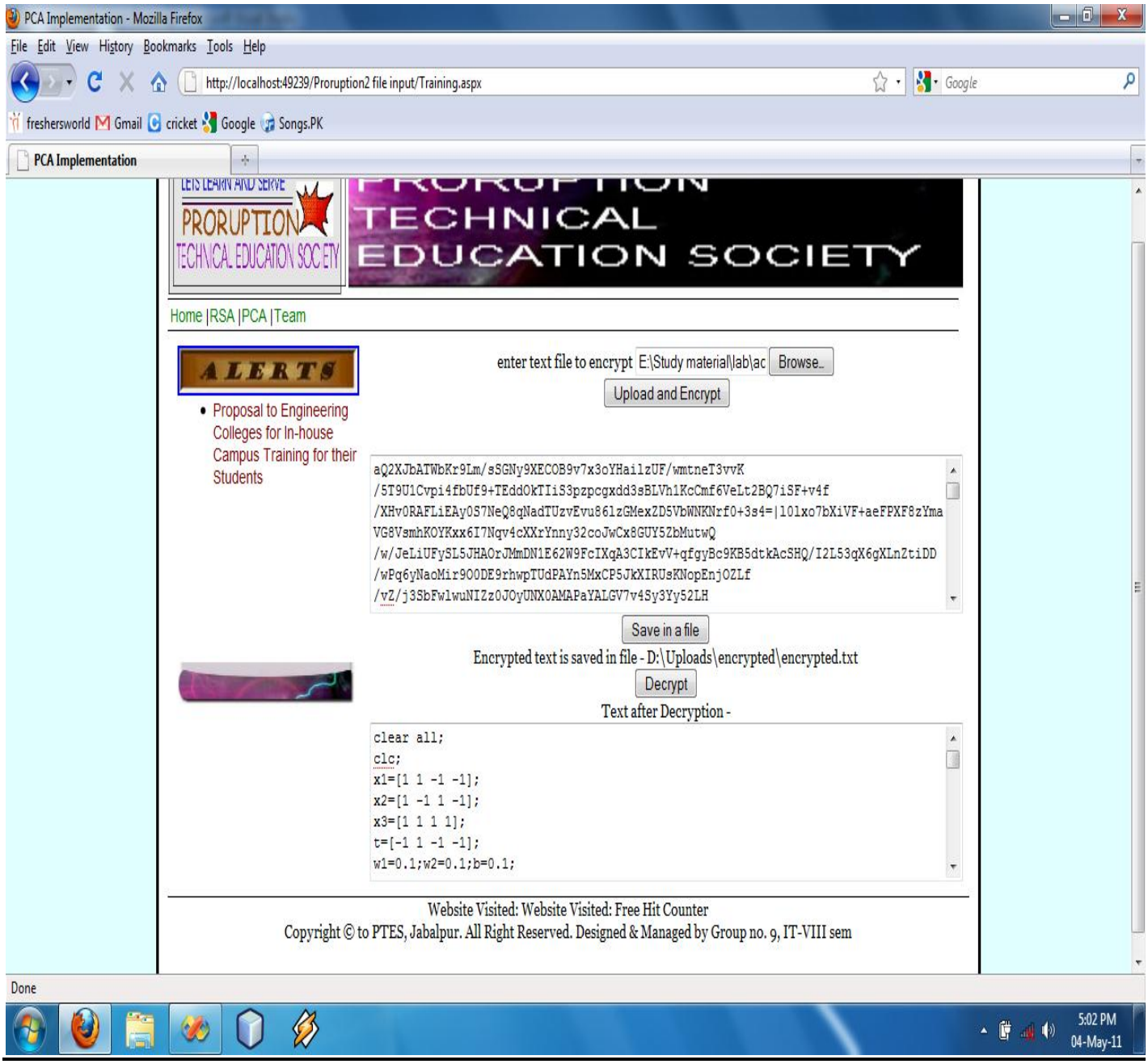
Results and Analysis

Snapshots

Home Page:-



RSA Encryption



PCA Encryption

OBSERVATION

Attack Analysis

There is a number of attacks on Asymmetric-key cryptography which PCA based on. Hence, attacks on Asymmetric key are the same types on PCA. The important types of attack on PCA can be categorized as follows: • Factorization Attack • Brute force attack

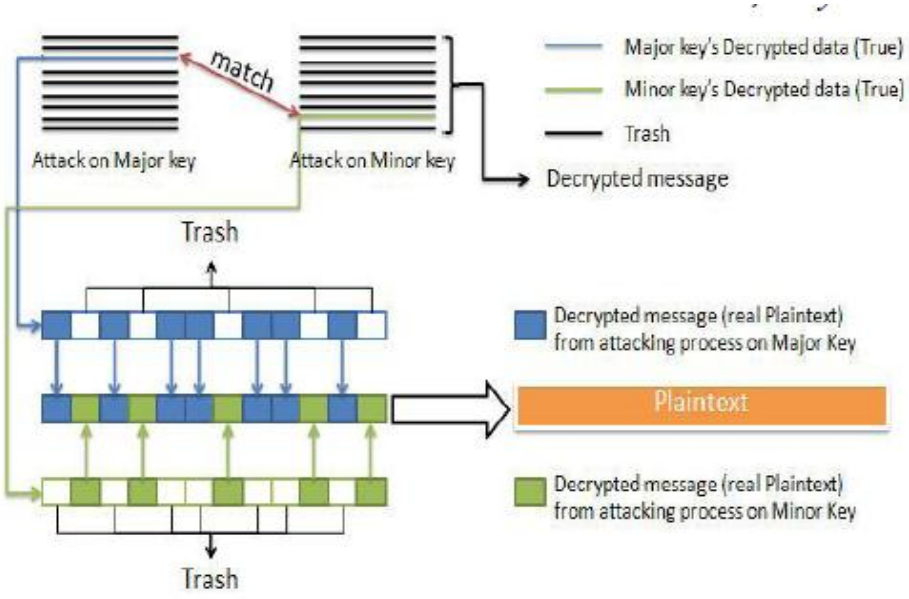
Factorization Attack:

This is a type of attack where it tries to find the factors of a large number called ―public Modulus‖, n with the polynomial time complexity in the shortest possible time. By using the parallel-key pair algorithm comprising of two key pairs, it can prevent the message from attacker in more secure environment because PCA has two key pairs. Then, attacker has to do more work in breaking two asymmetric-key pair with Public Modulus of their own.

Brute Force Attack:

This is a type of attack which tries to find all possible private key for complete decryption of the data. In practice, the small private key d would help in making the decryption faster. But, from the study of key size, wiener showed that a small size of d is easy to break with the special type of attack based of continuous fraction. Then, for RSA it is recommended to have to prevent low decryption exponent attack, where n is a Public Modulus. By

using the benefit of PCA, we can fight against Brute force attack better than common asymmetric-key cryptographic algorithm. From theoretical calculation of brute force attack, we assume that major key and minor key are of length 2048 and 1024 bits respectively. The unique feature of PCA is due to its application of random pattern i.e. the blocks of plain text are encrypted by the major key and minor key randomly and hence it adds to the difficulty in identifying which key is used for the encryption process. The randomness property of PCA accounts for its uniqueness and hence provides enhanced protection of data. The texts are randomly encrypted and correspondingly they are decrypted i.e. texts encrypted public major will be decrypted by private major key. So, once the text is encrypted and then decrypted the method of selection and combining the texts can be shown as:



From the above diagram we can understand how the PCA is more effective against the brute force attack and is more competent than RSA. [4] From experimental analysis, the time taken for brute force attack for PCA and RSA algorithm is given as follows:-

TIME OF BRUTE FORCE ATTACK

ALGORITHM	TIME OF BRUTE FORCE ATTACK(years)
PCA 512 bits(minor)	(1.224*10^63) +(3.596*10^24)+ (1.497*10^126)
RSA	1.224 *10^63

From the difference of time in Brute force attack on RSA and PCA, it shows that PCA can provide better security against Brute force attack. As we will see, in the calculation of Brute force attack on PCA and RSA, PCA can provide security of a plaintext longer approximately than RSA 1.497 *10126 years by attacking with 1 million operations /second computer.

Project Objectives & Related Benefits:

1. Enhanced Security :

A security audit involves checking a system for potential security problems. Things to check for include:

- Accounts without passwords
- Expired or dormant accounts
- Unusual accounts that have been added to the system
- Files that have no owners
- Duplicate UID's and GID's
- Changes in file permissions

- SUID and GUID programs, particularly those that are SUID root
- Copies of basic system programs in odd places-i.e. possible Trojan horses
- .rhosts files in odd places
- Changes in kernel files
- New commands that have been added
- Unauthorized changes to any system file
- Unusual system activity

2. Time :

   Here we propose a complete execution of cryptographic mode ICBC using parallel key encryption is done for the very first time. In ICBC, the division of blocks remains the same as in CBC but here the execution of blocks happens simultaneously thereby commendably reducing the time taken for the entire process execution. Here for the first time we have implemented ICBC with a key size range of 1024 and 2048 bit thereby surpassing all other cryptographic algorithms in terms of speed and security.

3. Efficiency

EFFICIENCY TABLE

INPUT File Size(KB)	CBC Execution Time(mS)	ICBC Execution Time(mS)
10	15052	8429
20	29232	17154
30	43132	24684
40	57988	34534
50	72037	43085
60	85928	52108

- # CONCLUSION AND SUGGESTIONS

We have proposed a implementation of the parallel key cryptographic algorithm with key sizes of 1024 bit and 2048 bit .We have also implemented cipher block chaining and interleaved cipher block chaining using the above mentioned keys. It has also been proven that PCA can substantially reduce the encryption and decryption time taken in the above cryptographic modes with higher speed and efficiency. From the theoretical results, it can be known that PCA can encrypt and decrypt messages faster than RSA. In attack analysis, it can be verified that the time taken in brute force attack is comparatively high than the other algorithms and hence enhanced protection of data. This algorithm is more flexible in transferring of data through communication networks or insecure channels without key limitations and key agreements. It can be extended by incrementing the number of keys used in the encryption and decryption process and also enhancing the key size further. It can have its applications in digital signatures , credit card transactions, message authentication (sign cryption) and transfer of confidential and sensitive data.

SWOT Analysis :

SWOT analysis is a strategic planning method used to evaluate the Strengths, Weaknesses, Opportunities, and Threats involved in a project or in a business venture. It involves specifying the objective of the business venture or project and identifying the internal and external factors that are favorable and unfavorable to achieve that objective. The technique is credited to Albert Humphrey, who led a

convention at Stanford University in the 1960s and 1970s using data from Fortune 500 companies.

A SWOT analysis must first start with defining a desired end state or objective. A SWOT analysis may be incorporated into the strategic planning model. Strategic Planning has been the subject of much research.

- Strengths: high speed, high security, Time and efficiency.
- Weaknesses: very complex to design as we are using two variable length keys.
- Opportunities: web services need high security nowadays, banking etc.
- Threats: currently there are no threats for this project, but in future it may be possible as the technology is advancing day by day.

- Another way of utilizing SWOT is matching and converting.
- Matching is used to find competitive advantages by matching the strengths to opportunities.
- Converting is to apply conversion strategies to convert weaknesses or threats into strengths or opportunities.
- An example of conversion strategy is to find new markets.
- If the threats or weaknesses cannot be converted a company should try to minimize or avoid them.

The aim of any SWOT analysis is to identify the key internal and external factors that are important to achieving the objective. These come from within the company's unique value chain. SWOT analysis groups key pieces of information into two main categories:

- Internal factors – The strengths and weaknesses internal to the organization.
- External factors – The opportunities and threats presented by the external environment to the organization.

The internal factors may be viewed as strengths or weaknesses depending upon their impact on the organization's objectives. What may represent strengths with respect to one objective may be weaknesses for another objective. The factors may include all of the 4P's; as well as personnel, finance, manufacturing capabilities, and so on. The external factors may include macroeconomic matters, technological change, legislation, and socio-cultural changes, as well as changes in the marketplace or competitive position. The results are often presented in the form of a matrix.

SWOT analysis is just one method of categorization and has its own weaknesses. For example, it may tend to persuade companies to compile lists rather than think about what is actually important in achieving objectives. It also presents the resulting lists uncritically and without clear prioritization so that, for example, weak opportunities may appear to balance strong threats.

It is prudent not to eliminate too quickly any candidate SWOT entry. The importance of individual SWOTs will be revealed by the value of the strategies it generates. A SWOT item that produces valuable strategies is important. A SWOT item that generates no strategies is not important.

- # FUTURE SCOPE AND FUETHER ENHANCEMENT OF THE PROJECT

We have proposed a implementation of the parallel key cryptographic algorithm with key sizes of 1024 bit and 2048 bit .This paper has also implemented cipher block chaining and interleaved cipher block chaining using the above mentioned keys. It has also been proven that PCA can substantially reduce the encryption and decryption time taken in the above cryptographic modes with higher speed and efficiency. From the theoretical results, it can be known that PCA can encrypt and decrypt messages faster than RSA.

In attack analysis, it can be verified that the time taken in brute force attack is comparatively high than the other algorithms and hence enhanced protection of data. This algorithm is more flexible in transferring of data through communication networks or insecure channels without key limitations and key agreements. The paper can be extended by incrementing the number of keys used in the encryption and decryption process and also enhancing the key size further. It can have its applications in digital signatures , credit card transactions, message authentication (sign cryption) and transfer of confidential and sensitive data.

- **Bibliography**

[1] Thongpon Teerakonak,Sinchai KamolPhiwong ―Accelerating Asymmetric key cryptography using Parallel Key Cryptographic Algorithm (PCA)‖

[2] Applied Cryptography, Bruce Schneider, second edition john wiley & sons, 1996

[3]http://en.wikipedia.org/wiki/Block_cipher_modes_of_operation

[4] http://en.wikipedia.org/wiki/RSA

[5] Yun Chen, Xin Chen, YiMu, ―A parallel key generation algorithm for efficient Diffie- Hellmann key agreement‖,IEEE transaction on information theory, 1-4244-0605-6/06.

[6] Ying Wang, Chunyan Han and Yuanyi Liu, ―A Parallel Encryption Algorithm for Color Images Based on Lorenz Chaotic Sequences‖, IEEE transaction on information Theory, 1-4244-0332-4/06, pp.9744-9747,2006