ISSN: 2321-7782 (Online)
Volume 3, Issue 1, January 2015
International Journal of Advance Research in Computer Science and Management Studies
Research Article / Survey Paper / Case Study
Available online at: www.ijarcsms.com

A Framework for Secure 3D Password using Genetic Algorithm

Dr. Poornima G. Naik[1]
Department of Computer Studies
Chh Shahu Institute of Business Education and Research
Kolhapur, India

Girish R. Naik[2]
Production Department
KIT's College of Engineering
Kolhapur, India

Abstract: In today's information age information sharing and transfer has increased exponentially. With the popularization of Internet and exponential increase in e-commerce transactions security has become an inevitable and an integral part of any e-commerce application. Data integrity, confidentiality, authenticity, non-repudiation have gained tremendous importance and have become important components of information security. There are many risks involved in communication of plain text over Internet. Cryptography is a technique of encoding and decoding messages so that they cannot be interpreted by anybody except the sender and the intended recipient. In most of the e-commerce applications where security is of prime importance, a single encryption algorithm is used for encrypting a password and the authentication data is stored on a single server which becomes amenable to risks against computer hacks. A novel solution to this problem is to generate a distributed password where the authentication information is distributed over multiple servers which are geographically separated. In this paper we have made an attempt to generate a distributed 3D password incorporating textual, graphical, and barcode authentication where the authentication information is distributed over multiple authentication servers. The client application communicates with the authentication servers using RMI technology. In the current work, we have adopted two-tier architecture where RMI client and RMI server reside on two different logical as well as physical tiers. RMI server is integrated with the data tier. The application can be rendered more maintainable by switching to 3-tier architecture by separating database server from the RMI server and using RMI security manager for remote connection. The encryption key is divided into three sub keys and stored on three different authentication servers. Based on the level of security and infrastructure availability the end user can make selection between 1 to 3 different levels and authentication methods. What is common to all methods is that all of them adopt Genetic Algorithm as a technique for the encryption and decryption of the password components. In the current work we have exploited the randomness involved in crossover and mutation processes for generating a asymmetric key pair for encryption and decryption of a password. The number of crossover points and number of mutation points together dictate the length of the secret key and hence the strength of the algorithm. The authentication data is encrypted on the client side and the encrypted data is transmitted over the network which is then compared with the encrypted data stored on respective authentication servers. Hence both during transmission and storage, data is encrypted to incorporate the data confidentiality in to the application. The randomness together with permutation makes the algorithm robust and hard to break. Finally, the algorithm is implemented in Java and applied for the encryption and decryption of a password of employees in an hypothetical organization.

Keywords: Barcode; Cross-over; Cryptography; Distributed; Genetic Algorithm; Mutation

I. INTRODUCTION

Genetic algorithms (GA) are adaptive heuristic search algorithms based on the evolutionary ideas of natural selection and genetics which mimics the process of natural selection. Genetic Algorithm has a wide application in generating useful solutions to search and optimization problems. In our current work, we have exploited the randomness inherently present in GA for generating a onetime symmetric key for encryption of authentication data. In a symmetric key encryption or secret key encryption only one key is used by both the sender and the intended receiver for both the encryption and decryption of the message. Both the sender and the intended receiver must agree upon the key before any communication begins. Fig. 1 Shows the working of symmetric key encryption. At the sender's end the key is used to encrypt the original message into an encrypted form known as a cipher text. At the receiver's end the same key is used to decrypt the encrypted message and restore a plain text from it.

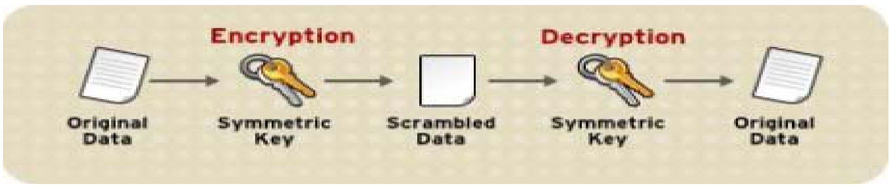


Fig. 1 Symmetric Key Encryption

CRYPTOGRAPHY

Cryptography is the science of writing in secret code. The purpose of cryptography is to protect transmitted information from being read and understood by anyone except the intended recipient. In the ideal sense, unauthorized individuals can never read an enciphered message Cryptographic systems are generally classified among three independent dimensions. Types of Operations – All encrypted algorithms are based on two general principles, substitution and transposition. The fundamental requirements are that no information is lost and all operations are reversible. The length of the key determines the strength of the security.

In our current work, GA algorithm transfers 16-byte plain text phrase into 16-byte cipher block. Each parent is a 8-byte plain text block On applying various crossover and mutation operations using a randomly generated one time symmetric key the corresponding cipher child block is generated. The symmetric key is generated randomly and consists of the following components. Randomly generated cross-over points in the range 0-7 Randomly generated mutation points in the range 0-7. The graphical data is encrypted using an algorithm designed by us, which involves fixed cross-over and mutation points and as such the dynamically generated key depends only on the number of points used in generating the pattern.

The strength of the key depends on the number of cross-over points and mutation points. The entire process is depicted in Fig. 2.

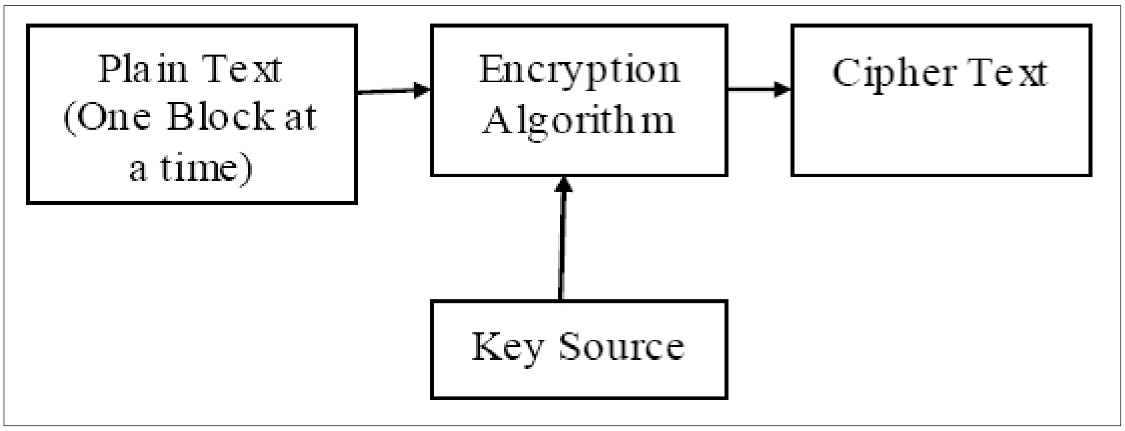


Fig. 2. Encryption Process

GENETIC ALGORITHM

Generally, a Genetic Algorithm consists of three basic operations.

- Selection
- Crossover
- Mutation

The first step consists of searching individuals for reproduction. In our problem, we have selected two vectors of 8 bytes each as parents for reproduction. Since the problem is of encryption, there is no special preference given to any particular selection method. The text phrase entered by the user is divided into two vectors of 8 bytes each. Blank padding is employed to generate vectors of desired length. Cross-over is the process of taking two parents and producing from them a child. In an optimization problem, crossover operator is applied to the mating pool with the hope that it creates a better offspring. For the problem under consideration, crossover is taken as one of the steps in producing a decrypted vector. We have employed two-point crossover method. In the case of optimization problem, selecting more than two crossover points will result in the disruption of building blocks whereas in the case of encryption larger the disruption better is the algorithm which makes it robust and difficult to break.

After crossover, the vectors are subject to mutation. In optimization problem, mutation prevents the algorithm from being trapped in a local minimum. Mutation plays the role of recovering the lost genetic matter as well as for a randomly distributed genetic information. In encryption problem, mutation is employed for inducing disorder into the vector. It introduces a new genetic structure in the population by randomly modifying some of the building blocks and maintains diversity into the population. We have employed flipping method, in which for a character 1 in mutation chromosome, the corresponding character b in the parent chromosome is flipped from b to (128-b) for character data and to (10-b) for numeric data and corresponding child chromosome is produced. In the following example 1 occurs at two random places of mutation chromosome, the corresponding characters in parent chromosomes are flipped and the child chromosomes are generated.

Parent Chromosome	b0	b1	b2	b3	b4	b5	b6	b7
Mutation Chromosome	1	0	0	0	0	0	0	1
Child Chomosome (for numeric data)	10-b0	b1	b2	b3	b4	b5	b6	10-b7
Child Chomosome (for character data)	128-b0	b1	b2	b3	b4	b5	b6	128-b7

STRUCTURE OF CODE-128 BARCODE

Barcodes consists of a series of lines that vary in width. They correspond to various numeric, alphanumeric, or multicode configurations readable by a laser barcode scanner. Code 128 is a very effective, high-density symbology which enables the encoding of alphanumeric data. It includes verification protection both through a checksum digit and byte parity checking. This symbology has been widely implemented in many applications where a large amount of data must be encoded in a relatively small amount of space. A Code 128 barcode consists of a leading "quiet zone", one of three start codes, the data itself, a check character, a stop character, and a trailing quiet zone as shown in Fig. 3. The Code 128 data is encoded in strips of bars and spaces. The sequences of zeros or ones simply appear as thicker bars or spaces. The checksum is included in the barcode, and is a digit that verifies that the data just read in was correct. The checksum digit is based on a modulo 103 calculation based on the weighted sum of the values of each of the digits in the message that is being encoded, including the start character.

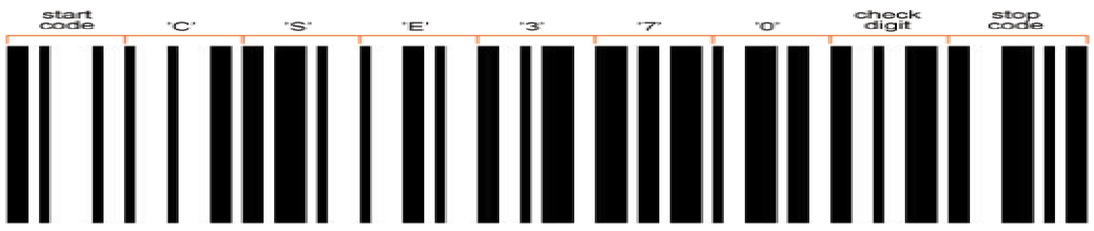


Fig. 3. Code-128 Barcode

Similar structure exists for Code-39 Barcode.

II. Literature Survey

In literature to date, many GA based encryption algorithms have been proposed. A. Tragha et.al [2] have describe a new symmetric block cipher system namely, ICIGA (Improved Cryptographic Inspired by Genetic Algorithm) which generates a one time session key in a random process. The block size and key length are variables and can be fixed by the end user in the beginning of the cipher process. ICIGA is an enhancement of the system GIC (Genetic Algorithm inspired Cryptography) [3]. There are various proposed methods for image encryption such as quad tree approach, cellular automata [4, 5]. There are wide applications of GA in solving non-linear optimization problems in various domains [6,7]. But very few papers exist which exploit the randomness in the algorithm for implementation of security. Chaos theory and entropy have large application in secure data communication and the desired disorder is provided by inherent nature of genetic algorithm [8, 10]. Mohammad Sazzadul Hoque et.al [11] have presented an intrusion detection system by applying GA to efficiently detect different types of network intrusions. They have used evolutionary theory to filter the traffic data thereby reducing the complexity [12]. There are several papers related to IDS all of which use GA in deriving classification rules [13, 15]. The authors have proposed a symmetric key and asymmetric key encryption/decryption algorithm based on Genetic Algorithm which is implemented in Java for encryption and decryption of a Word document. In their work they have employed four cross-over points and three mutation points, a random factor and a permutation factor to generate a 36-bit robust key [16,17]. But to the best of our knowledge very few papers exist which exploit randomness in generating barcode for authentication purpose.

In this paper we have made an attempt, to device a new approach to data encryption based on symmetric key. The randomness involved in crossover and mutation is exploited in generation of a onetime symmetric key. Instead of storing the authentication information on a single server, it is distributed over geographically separated multiple authentication servers. Finally, the algorithm is implemented in Java. In our current work, we have incorporated three different authentication levels in three different authentication servers which store

- encrypted textual password
- encrypted graphical password and
- encoded barcode information.

Different e-commerce applications demand different levels of security based on the financial transactions and the confidential data they deal with. Depending on the need of security as dictated by the underlying e-commerce application one or more levels of authentication can be employed by an end user.

III. Proposed Method

This paper proposes the IND-OCPA-P model to analyze the security of the proposed EOB and the encryption schemes supporting an efficient range query over encrypted data.

PROPOSED METHOD

ENCRYPTION OF A TEXT PHRASE

Step 1: Extract two 8-byte blocks from the text phrase to be encrypted. Let the two blocks be represented by

b0	b1	b2	b3	b4	b5	b6	b7

c0	c1	c2	c3	c4	c5	c6	c7

where each bi and ci is a character in a textual phrase.

Step 2 : Perform crossover operation.

Generate two random numbers in the range [0-7]. Let the two random numbers generated be 2 and 5.

Hence, Crossover Point1 = 1

Crossover Point1 = 6

Perform the crossover between two crossover points generated above.

b0	b1	b2	b3	b4	b5	b6	b7
c0	c1	c2	c3	c4	c5	c6	c7

The blocks after performing crossover operation are

b0	b1	c2	c3	c4	c5	c6	b7
c0	c1	b2	b3	b4	b5	b6	c7

Step 3: Perform mutation operation.

Generate two random numbers in the range 0 to 7. Let the two random numbers generated be 1 and 7.

Hence, Mutation Point1 = 3

Mutation Point2 = 4

Perform mutation operation on two blocks obtained in Step 2.

b0	b1	c2	128-c3	128-c4	c5	c6	b7
c0	c1	b2	128-b3	128-b4	b5	b6	c7

Step 4: Generate a random key based on crossover points, mutation points and crossover factor generated above.

Hence the symmetric key in an octal form is

1	6	3	4

Each octal digit in a symmetric key can be represented using 3 bits. Hence a symmetric key in a binary format is given by

In the above example, c = m = 2.

Hence Key Length = 12.

Hence the length of the key is 15 bits in our case which depends on the number of crossover points and mutation points. The length of the symmetric key can be computed using a general formula 3 * (c + m)

ENCRYPTION OF GRAPHICAL PASSWORD

Step 1 :

Let (x_i, y_i) denote the x and y coordinates of the end points of a pattern generated by the user. Generate two vectors consisting of x-coordinates and y-coordinates of all the points on the pattern as shown below:

These vectors become the parent chromosomes.

x1	x2	x3	x4	x5	. . . .	xi
y1	y2	y3	y4	y5	. . . .	yi

Step 2:

Select mid point as a single cross over point.

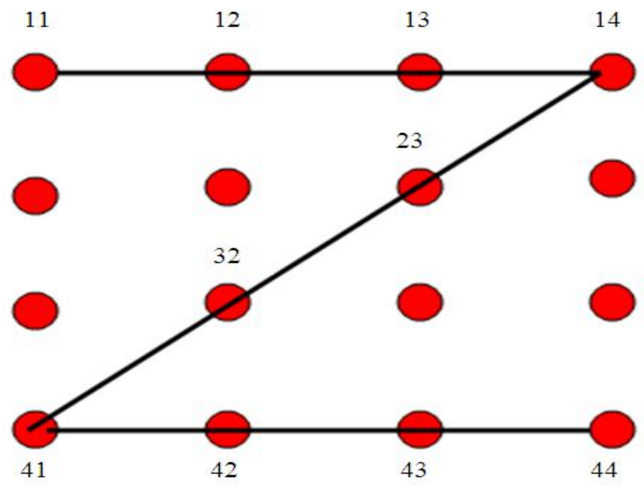
x1	x2	x3	x4	. . . .	xi/2	. . . .	xi
y1	y2	y3	y4	. . . .	yi/2	. . . .	Yi

Step 3:

Perform mutation operation at the two extreme points.

10-x1	x2	x3	x4	. . . .	xi/2	. . . .	10-xi
10-y1	y2	y3	y4	. . . .	yi/2	. . . .	10-yi

The entire procedure is illustrated in the following example. Let the pattern generated be as follows.



Step 1 : Generate two parent chromosomes from the given pattern as shown below.

1	1	1	1	2	3	4	4	4	4
1	2	3	4	3	2	1	2	3	4

Step 2 : Perform cross over operation at the mid point to obtain new chromosomes.

1	2	3	4	3	3	4	4	4	4
1	1	1	1	2	2	1	2	3	4

Step 3 : Perform mutation operation at the two extreme points to yield the final chromosomes.

9	1	1	1	2	3	4	4	4	6
9	2	3	4	3	2	1	2	3	6

These two transformed chromosomes are stored in a database on a remote server.

The decoding procedure to generate the above pattern is depicted below:

Step 1 : Perform reverse mutation operation at the extreme points to yield the following chromosomes.

1	2	3	4	3	3	4	4	4	4

1	1	1	1	2	2	1	2	3	4

Step 2 : Perform crossover at a single crossover point to yield the original chromosomes.

1	1	1	1	2	3	4	4	4	4

1	2	3	4	3	2	1	2	3	4

The x and y coordinates of the points on the pattern are given by

(1, 1), (1, 2), (1, 3), (1,4), (2,3), (3,2), (4,1), (4,2), (4,3), (4,4), which restores the original pattern.

PSEUDOCODE FOR ENCRYPTION OF GRAPHICAL DATA

C-style pseudo code for encryption of graphical data is depicted below:

```
function encryptGraphicalData()
{
   /* Read x and y coordinates of points on the patter */
   for (int i=0;i<n;i++)
   {
       Read x[i],y[i];
    }
 /* Construct two parent chromosomes, cr1 and cr2 */
   cr1=0;
   for (int i=0;i<n;i++)
   {
     cr1=cr1*10+x[i];
   }
   cr2=0;
   for (int i=0;i<n;i++)
   {
     cr2=cr2*10+y[i];
   }
/* Determine crossover- point, pt */
   pt=n/2;
/* Perform cross-over operation */
   for (int i=0;i<pt;i++)
   {
        temp=x[i];
        x[i]=y[i];
        y[i]=temp;
    }
/* Perform mutation at extreme points */
    x[0]=10-x[0];
    y[0]=10-y[0];
    x[n-1]=10-x[n-1];
    y[n-1]=10-x[n-1];
}
```

C-style pseudo code for decryption of graphical data is depicted below:

```
function decryptGraphicalData()
{
/* Perform mutation at extreme points */
    x[0]=10-(10-x[0]);
    y[0]=10-(10-y[0]);
    x[n-1]=10-(10-x[n-1]);
    y[n-1]=10-(10-x[n-1]);
/* Perform cross-over operation */
    for (int i=0;i<pt;i++)
    {
        temp=x[i];
        x[i]=y[i];
        y[i]=temp;
    }
}
```

The significance of the above algorithm is that the crossover point is uniquely determined by the size of the vector and need not be explicitly determined and stored anywhere which makes the technique more robust and further secure. Hence the key can be represented in terms of the size n of the vector as

$$\boxed{n/2\ 1\ n}$$

For example, for the vector of size 10, the key is

$$\boxed{5110}$$

The maximum value that n can take is 16.

GENERATION AND ENCRYPTION OF BARCODE DATA

We have used Code-39 and Code-128 encoding techniques for generating a barcode. The barcode data comprises of 12 randomly generated decimal digits. Each decimal digit is represented using 4 bits. Hence the length of the barcode data is 36 bits. The randomly generated data is transformed into encoded form by applying crossover, mutation and XOR operations before generating a bar code. The application architecture is shown in Fig. 4.

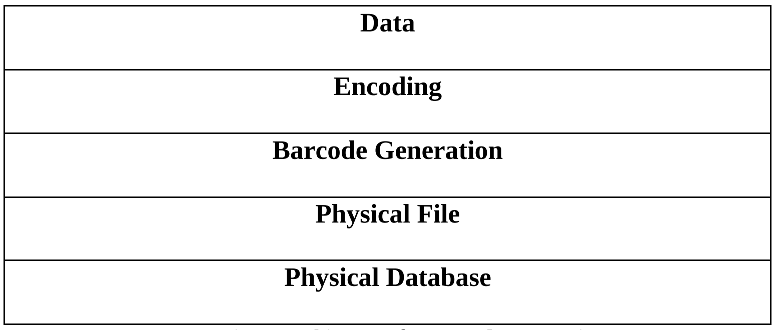


Fig. 4. Architecture for Barcode Encryption

The pseudo code and the mathematical formulation for generation and encryption of barcode data is given by the authors [18] which is summarized below.

MATHEMATICAL FORMULATION

Let the original vector be represented by $V_{Original}$. Let H be the hash constructed as follows.

H= Σ’ Hi where 1 <= i <= 12 and Σ’ is the string concatenation operator.

Hi = 0000, for i = 8 or 9

= 0101, otherwise.

H is the generated hash of length 48 bits.

Compute the hash of $V_{Original}$ as shown below:

$$V_{Original}\ \theta\ H = V_{Hash}$$

Split the hash into two vectors of size six each. Let the two parts be represented by, V^1_{Hash} and V^2_{Hash}, respectively.

$$V_{Hash} = V^1_{Hash} + V^2_{Hash}$$

Compute 10’s complement of each digit. Let the two parts be represented by V^1_{Hash} and V^2_{Hash} respectively.

Perform the crossover operation at the midpoint. Let the two new parts now be represented by $C(V^1_{Hash})$ and $C(V^2_{Hash})$, respectively, where C is the crossover operator.

Perform the mutation at the extreme positions of the vector. Let the two parts now be represented by $MC(V^1_{Hash})$ and $MC(V^2_{Hash})$ respectively where M is the crossover operator. Combine the vectors to reconstruct a 12-digit vector.

Perform the XOR operation between the data and a 48-bit hash, H computed above to generate a final vector. Let it be $V_{Transformed}$. We get,

$$V_{Transformed} = [\ MC(V1_{Hash})' + MC(V2_{Hash})'\ ]\ \theta\ H$$

DECODING THE VECTOR INTO ORIGINAL VECTOR

Perform XOR operation between H and VTransformed given by equ(1) to get,

$$[\ MC(V^1_{Hash})' + MC(V^2_{Hash})'\ ].$$

Split the hash into two vectors of size six each. Let the two parts be represented by, MC(V^1_{Hash}) and MC(V^2_{Hash})’, respectively.

Perform reverse mutation operation and then reverse cross0ver operation on two individual parts to get, V^1_{Hash} and V^2_{Hash} respectively.

Take 10’s complement of each digit in the two vectors to get (V^1_{Hash}) and (V^2_{Hash}), respectively.

Combine the two vectors to get V_{Hash}, where

$$V_{Hash} = V_{Original}\ \theta\ H$$

Perform XOR operation between H and V_{Hash} to get the original vector.

Fig. 5 summarized the distribution of data on multiple servers.

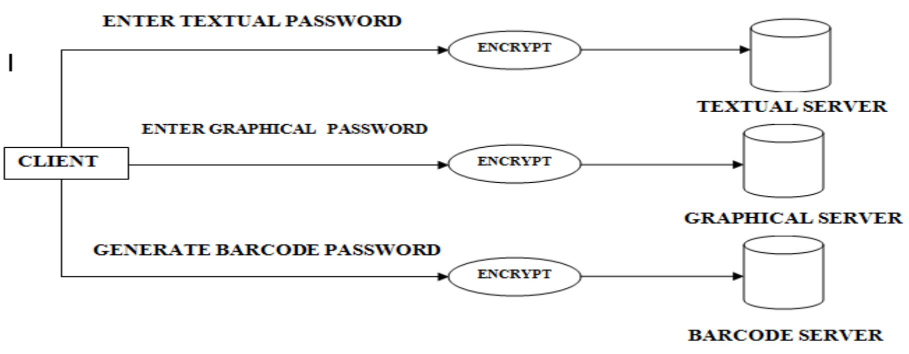


Fig. 5. Distribution of Authentication Data on Multiple Servers

Fig. 6 and 7 depict distributed authentication mechanism and layered architecture of distributed password, respectively.

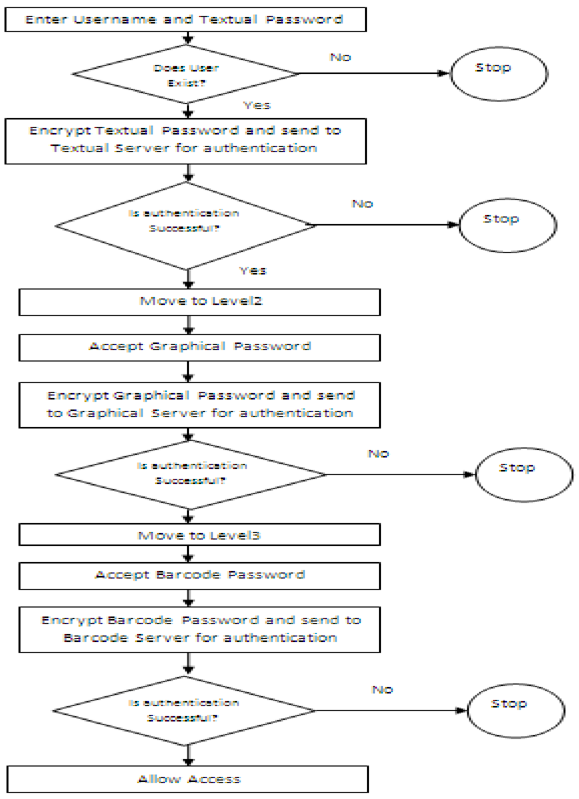

Enter Username and Textual Password
Does User Exist?
No
Stop
Yes
Encrypt Textual Password and send to Textual Server for authentication
Is authentication Successful?
No
Stop
Yes
Move to Level2
Accept Graphical Password
Encrypt Graphical Password and send to Graphical Server for authentication
Is authentication Successful?
No
Stop
Move to Level3
Accept Barcode Password
Encrypt Barcode Password and send to Barcode Server for authentication
Is authentication Successful?
No
Stop
Allow Access

Fig. 6. 3D Distributed Authentication Process

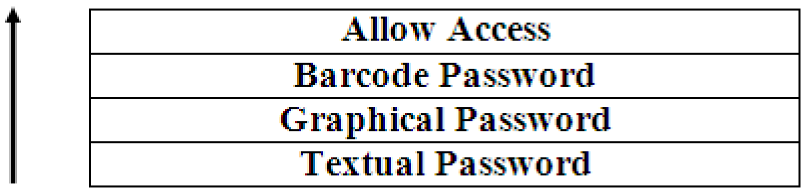
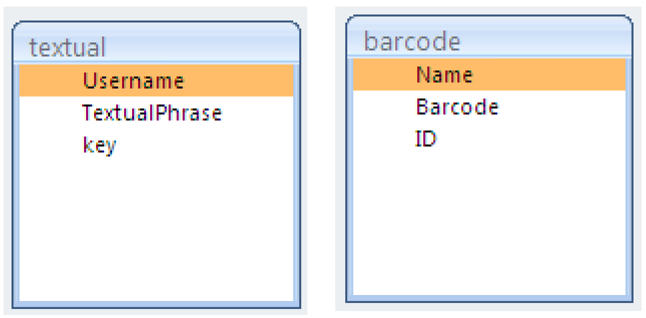


Fig. 7. Layered Architecture of Distributed Password

IV. RESULTS AND ANALYSIS

IMPLEMENTATION IN JAVA

The results presented above are implemented in Java with MS-Access as backend for storing authentication information. The structure of the database is shown in the following Fig. 8.


textual
Username
TextualPhrase
key

barcode
Name
Barcode
ID

Fig. 8 Database Structure for storing Authentication Information

Fig. 9 depicts the partial class diagram for implementation of distributed framework in Java.

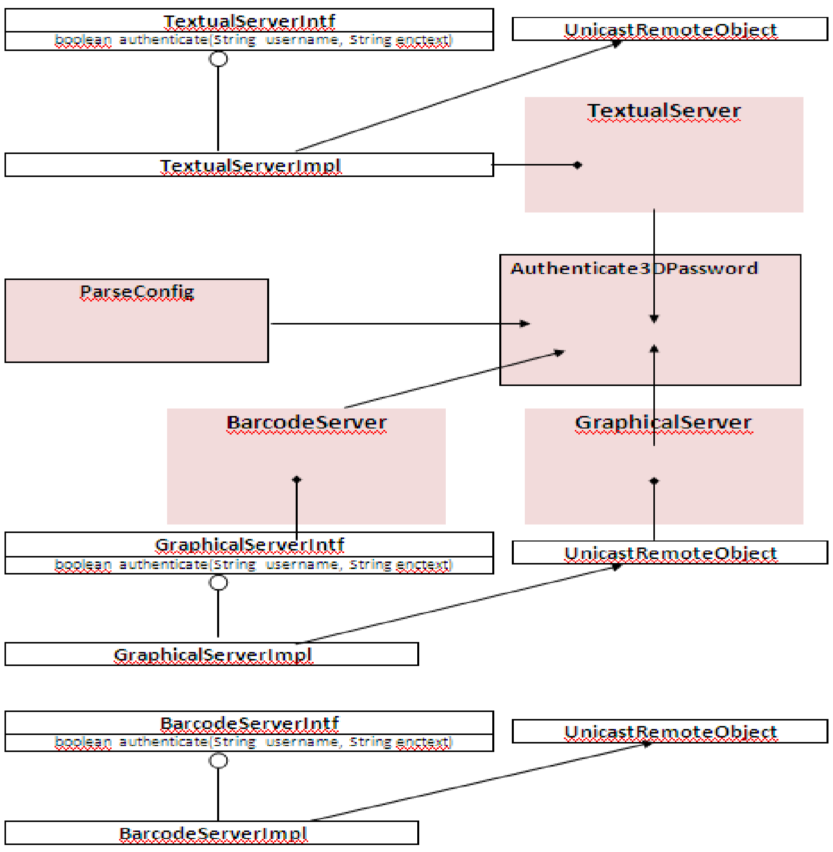
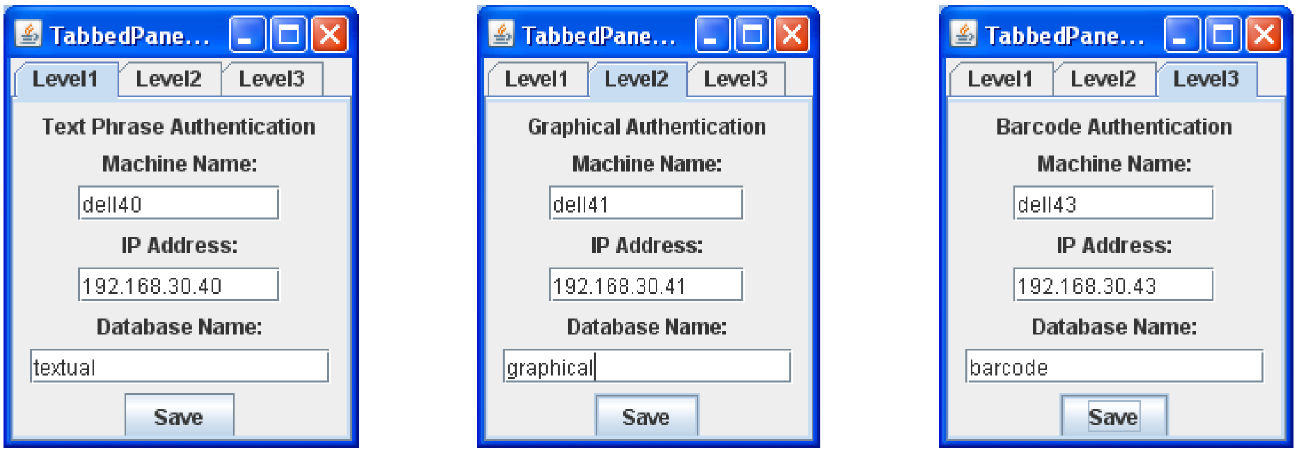


Fig. 9. Class Diagram for implementation of Distributed Framework in Java

Fig. 10 (a)-(c) depict the user interface in Java swing for storing machine configuration information in an XML file, passwordconfig.xml.



TabbedPane... | Level1 | Level2 | Level3

Text Phrase Authentication
Machine Name: dell40
IP Address: 192.168.30.40
Database Name: textual
Save

TabbedPane... | Level1 | Level2 | Level3

Graphical Authentication
Machine Name: dell41
IP Address: 192.168.30.41
Database Name: graphical
Save

TabbedPane... | Level1 | Level2 | Level3

Barcode Authentication
Machine Name: dell43
IP Address: 192.168.30.43
Database Name: barcode
Save

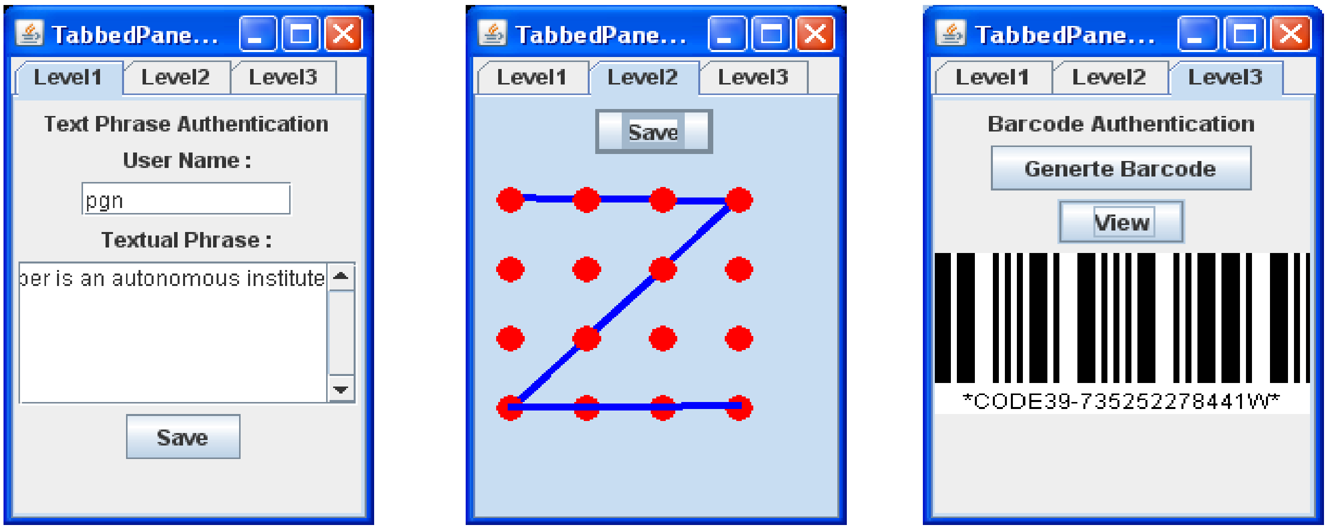
Fig. 10 (a)-(c) Storing Machine Configuration information in XML Format

On saving machine configuration information the following XML file is generated.

```
<config>
  <level>
        <name>Textual</name>
        <machineName>dell40</machineName>
        <ipAddress>192.168.30.40</ipAddress>
        <databaseName>textual</databaseName>
  </level>
  <level>
        <name>Graphical</name>
        <machineName> dell41</machineName>
        <ipAddress>192.168.30.41</ipAddress>
        <databaseName>graphical</databaseName>
  </level>
  <level>
        <name>Barcode</name>
```

```
            <machineName> dell42</machineName>

            <ipAddress>192.168.30.42</ipAddress>

            <databaseName>barcode</databaseName>

       </level>

    </config>
```

Fig. 11 a)-c) display the user interface for storing authentication information in distributed database servers conforming to the data stored in an XML file generated above.



pgn

ber is an autonomous institute

CODE39-735252278441W

Fig. 11 a) -c) User Interface for Storing Authentication Information.

The sample code to parse the XML configuration file is given below.

```
File config = new File("passwordconfig.xml");

DocumentBuilderFactory dbFactory = DocumentBuilderFactory.newInstance();

DocumentBuilder dBuilder = dbFactory.newDocumentBuilder();

Document doc = dBuilder.parse(config);
```

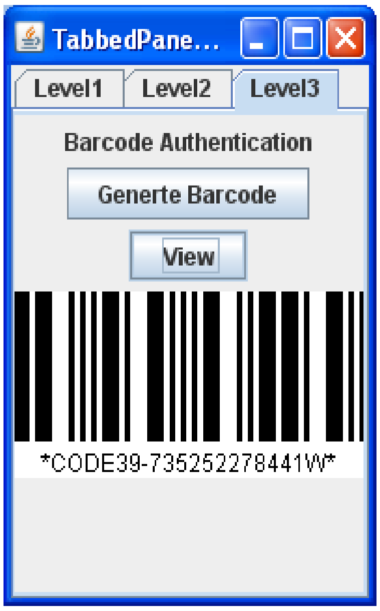
The following code snippet depicts the generation of barchar in Java.

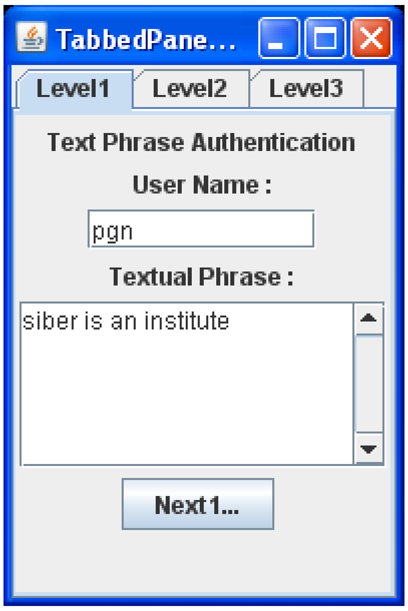
```
import com.barcodelib.barcode.Linear;

Linear barcode = new Linear();

barcode.setType(Linear.CODE39);

// barcode data to encode

barcode.setData(finalstring);

// wide bar width vs narrow bar width ratio

barcode.setN(3.0f);

// unit of measure for X, Y, LeftMargin, RightMargin,
TopMargin, BottomMargin

barcode.setUOM(Linear.UOM_PIXEL);

// barcode module width in pixel

barcode.setX(3f);

// barcode module height in pixel

barcode.setY(75f);

barcode.setLeftMargin(0f);

barcode.setRightMargin(0f);

barcode.setTopMargin(0f);

barcode.setBottomMargin(0f);

// barcode image resolution in dpi

barcode.setResolution(72);

// disply human readable text under the barcode

Secure Barcode Authentication using Genetic Algorithm

www.iosrjournals.org 140 | Page

barcode.setShowText(true);

// human reable text font style

barcode.setTextFont(new Font("Arial", 0, 12));

// ANGLE_0, ANGLE_90, ANGLE_180, ANGLE_270

barcode.setRotate(Linear.ANGLE_0);

barcode.setAddCheckSum(true);

bfilename="Noname";

if (!t1.getText().equals(""))

bfilename="C:\\"+t1.getText()+".gif";

barcode.renderBarcode(bfilename);
```

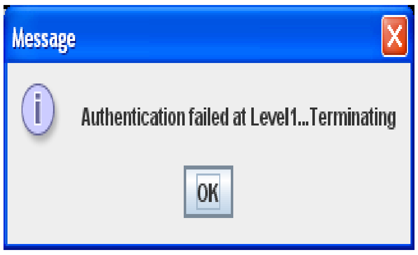
The code snippet for accessing RMI Textual Server is shown below. Similar code exists for accessing Graphical and Barcode servers.

```
String textualServerURL="rmi://"+machineName[0]+"/TextualServer";

TextualServerIntf textualserver=(TextualServerIntf)

Naming.lookup(textualServerURL);
```

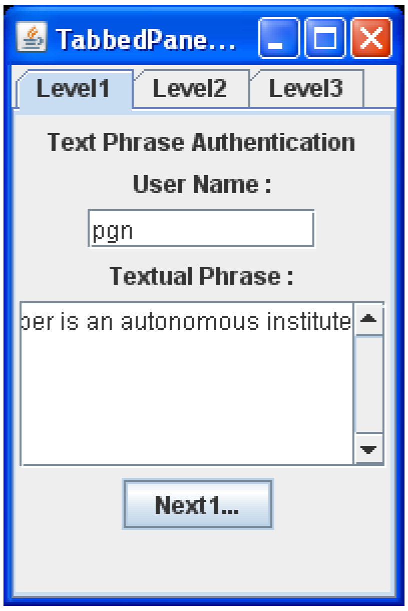
Fig. 12 a) -i) depict user interface for distributed authentication process.

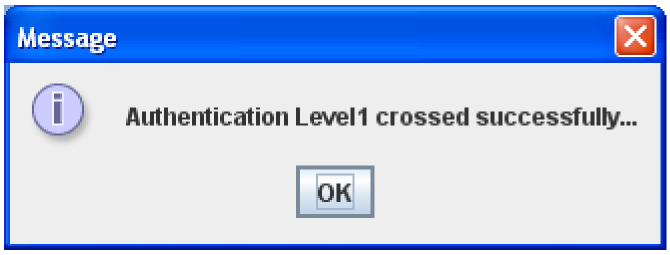




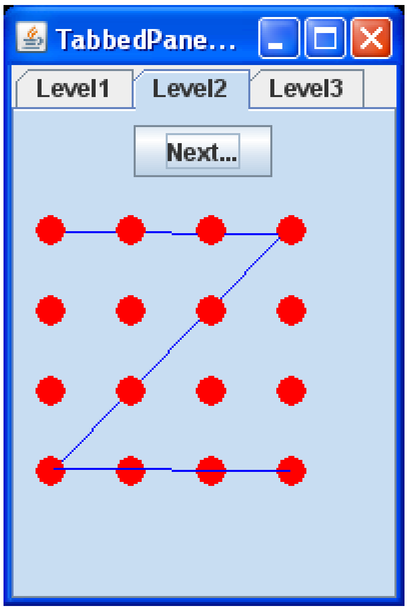


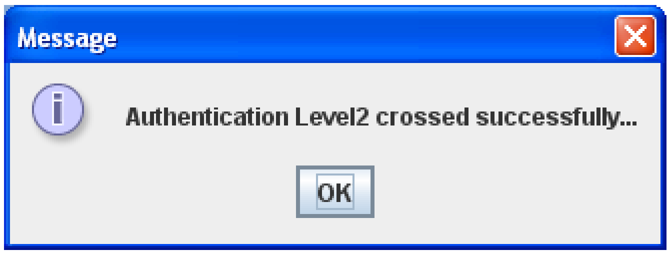
Authentication failed at Level1...Terminating



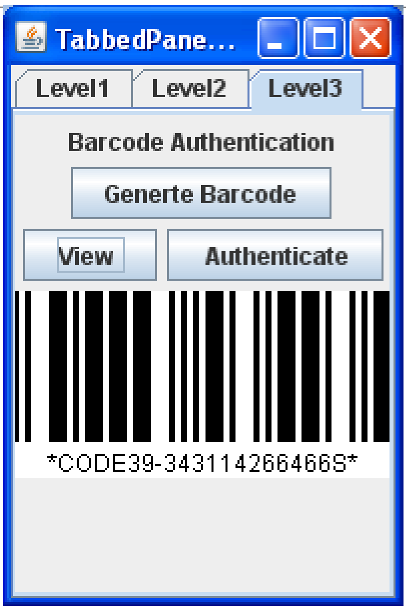


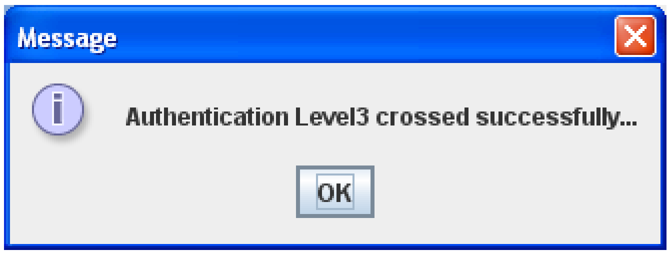
Authentication Level1 crossed successfully...





Authentication Level2 crossed successfully...




Message

Authentication Level3 crossed successfully...

Fig. 12 a) - i) User Interface for Distributed Authentication Process.

V. Conclusion

In this paper we have generated a distributed 3D password incorporating textual, graphical, and barcode authentication where the authentication information is distributed over multiple authentication servers. The client application communicates with the authentication servers using Java RMI technology. We have adopted two-tier architecture where RMI client and RMI server reside on two different logical as well as physical tiers. RMI server is integrated with the data tier. The application can be rendered more maintainable by switching to 3-tier architecture by separating database server from the RMI server and using RMI security manager for remote connection. Based on the level of security and infrastructure availability the end user can make selection between 1 to 3 different levels and authentication methods. Our future work focuses on incorporating biometric authentication as one of the authentication methods in multi layered security framework and employing more robust backend database management system such as Oracle 12C or MySQL Sever where backend information will be stored in a configuration file. This feature will enable the end user flexibility in back end selection. Further, the distributed database can be replaced very efficiently with the distributed fie system using the upcoming technology such as Hadoop. Hadoop Distributed File System (HDFS) offers several advantages such as location transparency, location independency, scalability and caching.

References

1. David. E. Goldberg, “Genetic Algorithms in Search, Optimization, and Machine Learning”, Pearson Education, 1989, ISBN-13: 978-020115767.
2. X. F. Liao, S. Y.Lai and Q. Zhou. Signal Processing. 90 (2010) 2714–2722.
3. H. Cheng and X. Li. IEEE Transactions on Signal Processive. 48 (8) (2000) 2439–2451.
4. O. Lafe. Engineering Applications of Artificial Intelligence. 10 (6) (1998) 581–591.
5. R. J. Chen and J. L. Lai. Pattern Recognition. 40 (2007) 1621–1631
6. Dr.Poornima G. Naik, Girish R. Naik, Application of Genetic Algorithm to Mass Production Line for Productivity Improvement, International Journal of Latest Trends in Engineering and Technology (IJLTET) Special Issue – IDEAS-2013 ISSN:2278-621X.
7. S. Li, G. Chen and X. Zheng. Multimedia security handbook. LLC, Boca Raton, FL, USA: CRC Press; (2004) [chapter 4].
8. Y. Mao and G. Chen. Handbook of computational geometry for pattern recognition, computer vision, neural computing and robotics. Springer; (2003).
9. H. S. Kwok, W. K. S. Tang, Chaos Solitons and Fractals, (2007) 1518–1529.
10. Mohammad Sazzadul Hoque, Md. Abdul Mukit and Md. Abu Naser Bikas, An Implementation of Intrusion Detection System Using Genetic Algorithm, International Journal of Network Security & Its Applications (IJNSA), Vol.4, No.2, March 2012
11. L.M.R.J Lobo, Suhas B. Chavan, Use of Genetic Algorithm in Network Security, International Journal of Computer Applications (0975 – 8887)Volume 53– No.8, September 2012
12. W. Lu, I. Traore, “Detecting New Forms of Network Intrusion Using Genetic Programming”. Computational Intelligence, vol. 20, pp. 3, Blackwell Publishing, Malden, pp. 475-494, 2004.

13. M. M. Pillai, J. H. P. Eloff, H. S. Venter, "An Approach to Implement a Network Intrusion Detection System using Genetic Algorithms", Proceedings of SAICSIT, pp:221-228, 2004.

14. S. M. Bridges, R. B. Vaughn, "Fuzzy Data Mining And Genetic Algorithms Applied To Intrusion Detection", Proceedings of 12th Annual Canadian Information Technology Security Symposium, pp. 109-122, 2000.

15. M. Middlemiss, G. Dick, "Feature selection of intrusion detection data using a hybrid geneticalgorithm/KNN approach", Design and application of hybrid intelligent systems, IOS Press Amsterdam, pp.519-527, 2003.

16. Poornima G. Naik and Girish R. Naik, Symmetric Key Encryption using Genetic Algorithm, International Journal of Information systems (A Journal of SIMCA), Vol IV, Issue II, Jan-May 2014, ISSN:2229-5429

17. Poornima G. Naik and Girish R. Naik, Asymmetric Key Encryption using Genetic Algorithm, International Journal of Latest Trends in Engineering and Technology (IJLTET), Vol. 3 Issue 3 January 2014, 118-128, ISSN: 2278-621X.

18. Poornima G. Naik and Girish R. Naik, Secure Barcode Authentication using Genetic Algorithm, IOSR Journal of Computer Engineering (IOSR-JCE), 8727Volume 16, Issue 2, PP 134-142, e-ISSN: 2278-0661, p- ISSN: 2278-8727.

AUTHOR(S) PROFILE

Dr. Poornima G. Naik, received M.Sc. degree in Physics and Mathematics and Ph.D. degree in physics from Karnataka University, Dharwad. She received MCA degree from IGNOU with first class Distinction. Currently, she is working as Assistant Professor in the Department of Computer Studies, SIBER, Kolhapur since 1998. Her areas of interest are network security, soft computing and cloud computing. She has participated in several national and international conferences and has published more than 20 papers in International and national journals of repute.

Mr. Girish R. Naik, received B.E. in Industrial and Production Engineering from Karnataka University and M.E. in Production Engineering from Walchand college, Sangli affiliated to Shivaji University. Currently, he is working as Associate Professor in Department of Production at KIT's college of Engineering, Kolhapur since 1989. He has participated in several national and international conferences and has published more than 30 research papers in International and national journals of repute.