KNOWLEDGE BASED MULTICASTING SYSTEM

PROJECT REPORT

Submitted in partial fulfillment of the requirements for the award of the degree of the
Bachelor of Engineering in Information Technology
of **Bharathiyar University, Coimbatore**

Submitted by

R.LAKSHMAN
G.PRADEEP
S.PRADEEPKUMAR
A.PRIYARAJA

Guided by

Ms. N.RAJATHI B.E.,

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

KUMARAGURU COLLEGE OF TECHNOLOGY,
COIMBATORE 641 006.

MARCH 2002

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

KUMARAGURU COLLEGE OF TECHNOLOGY
(Affiliated to Bharathiyar University, Coimbatore)

CERTIFICATE

This to certify that project entitled

KNOWLEDGE BASED MULTICASTING SYSTEM

done by

R.LAKSHMAN (9827S0201)
G.PRADEEP (9827S0019)
S.PRADEEP KUMAR (9827S0202)
A.PRIYARAJA (9827S0203)

in partial fulfillment of the requirements for the award of the **Degree of Bachelor of Engineering** *in the* **Information Technology** *of the* **Bharathiyar University**, *Coimbatore - 641 046 during the academic year 2001-2002.*

16/3/02
Professor & Head of the department
Dr.S.THANGASAMY

Guide
Ms.N.RAJATHI

Certified that the candidate was examined by us in the project work viva-voce examination held on 18·3·02
the university register number is ____________

Rajini
Internal Examiner

18/3/2002
External Examiner

ACKNOWLEDGEMENT

ACKNOWLEDGEMENT

We express our deep sense of gratitude to **Dr.K.K.Padmanaban Ph.D.,** Principal, Kumaraguru College of Technology, Coimbatore, for providing innumerable facilities to carryout this project work.

With profound sense of gratitude and regards, we acknowledge with great pleasure for the guidance and support extended by **Prof.S.Thangasamy Ph.D.,** Head of the Department of Computer Science and Engineering, for his valuable and continuous guidance and persistent encouragement.

We express our deep sense of respectful gratitude to our guide **Ms.N.Rajathi B.E.,** Senior Lecturer, Department of Computer Science and Engineering for her valuable guidance, keen suggestions, innovative ideas, inspiration, helpful criticisms, enthusiasm and kind encouragement in all the phases of this project work. It had been indeed a great pleasure to work under her guidance.

We express our heartiest gratitude **Ms.S.Rajini B.E., Mrs.A.U.Amutha B.E.,** for all the help which they have provided for us without which we would not have been able to complete our project successfully.

We like to express our special thanks to all the staffs and lab technicians in the Department of Computer Science and Engineering who helped us for the successful completion of the project.

Finally, we express our deep sense of gratitude to our parents, friends, roommates and all other persons who directly or indirectly involved with this project for their invaluable help and consideration towards us.

Dedicated
to our
beloved
Parents

SYNOPSIS

SYNOPSIS

In the modern communication world as the usage of computers in the communication field has increased, the same way need of security also increases for the data's which are being transmitted.

Our project Knowledge Based Multicasting System is being done to meet the challenges of the modern world. We have done the process of multicasting data in an secured and efficient way, we have specially designed some of the encryption techniques to provide secured data transfer, our intelligent system here is that it keeps on changing the encryption algorithms in an random order which makes eavesdropper's work more tedious.

The additional feature of the project is that we have very effectively utilized the bandwidth of the network (i.e.) we have designed our own compression technique to abide this job.

CONTENTS

CONTENTS

INTRODUCTION

INTRODUCTION

Various modes of communication which are available on the network are

- Point to point.
- Broadcasting.
- Multicasting.

In point to point the message transfer can only between two hosts in the network. In Broadcasting the server can send data's to all those who have logged in and here there is no provision for message transfer between the group of nodes in the network.

Multicasting provides the exact solution for the above problem we can communicate with a group of computer nodes in the network which are registered to the network as authorized users.

PROPOSED SYSTEM AND ITS ADVANTAGES

1.1 PROPOSED SYSTEM AND ITS ADVANTAGES

In our project knowledge based multicasting system we have implemented the idea of multicasting data's in the network, our major criteria and the biggest advantage is that we provide secured means of data transfer in an effective way by compressing it and sending through the network. Advantages of our system are

- Random encryption algorithm selection which provides more security for the information which is being transmitted.
- Compression technique provides an effective utilization of the bandwidth.

SYSTEM REQUIREMENTS

PRODUCT DEFINITION

2.1 PRODUCT DEFINITION

WHAT IS **KBMS?**

This is intelligent software which obtains the plain text from the user's and encrypts it using any of our encryption techniques & compresses the data sends it in the network. If it finds any intruder the software changes the encryption technique for the next set of data's.

2.1 Multicasting:

Definition:

The process of sending data's to a particular group of user's in a large network is known as multicasting.

The places where do we need multicasting are when you are serving in the large network and you need a means communication with your group users then you can go for multicasting.

2.1.2 Cryptography:

The process of converting the obtained text (Plaintext) into Cipher text by means of using any encryption key is known as encryption, the encrypted text is known as cipher text and the reverse process is known as decryption.

Encryption provides a perfect security schema for your data's which are being transmitted in the network. Various encryption techniques designed by us are

- Transposition cipher.
- Substitution cipher.
- Swapping cipher.
- Classical product cipher.
- DES.

For the process of encryption we obtain the key from the world famous Tamil literature "**THIRUKURAL**".

Algorithms for Encryption and Decryption:

2.1.2.1 Transposition Cipher:

- Get the plain text and obtain the random number (n) from the random number generation algorithm which acts the key for encryption.
- Split the plain text into number of number of words with n length each.
- Reverse each word.
- Store the words in an "n*m" matrix in column wise manner.
- Retrieve the characters in row wise manner and concatenate all the rows.
- The final string which is obtained is the cipher text.
- Do the same steps in the reverse order we would the plaintext back.

2.1.2.1.1 Random Number Generation Algorithm:

- Obtain the system current date and add the date month and year.

- Take the modulo of 1330 for the above and store it in a variable.
- Obtain the current time and convert fully into seconds and then to milliseconds.
- Take the modulo of 1330 for that milliseconds and store it in variable.
- Add both the results and the obtained result is the random number.

2.1.2.2 Substitution Cipher:

- Get the plain text and pass it to the transposition cipher algorithm.
- Obtain the cipher text of transposition cipher and treat it as the plain text of substitution.
- Using the below formula, obtain the cipher text of the substitution cipher.

  $Ci=Prlcode\ [Pi+KEYi]$

  Where KEY=Random number,

  i=Range from 1 to length of plain text,

  C=Cipher text,

  P=Plaintext,

  Prlcode={A-Z,a-z,0-9," ",''$"}.
- Repeat the above steps in the reverse order to obtain the plain text back.

2.1.2.3 Swapping Cipher:

- Obtain the plain text, count the number of words in and store them in an array.
- Now divide the word count value by 2 and reverse each word which is stored in the array.
- In the output (Cipher Text) place the nth word in the first position and the (n-1) word in the second position and so on.
- In the decryption process do the same steps in the reverse order.

2.1.2.4 Classicalproductcipher:

- Obtain the plain text, split each character of the plain text and store it in an array.
- Now obtain a random number from the random number generation algorithm.
- Then obtain the appropriate kural from the "THIRUKURAL" database which acts as the key.
- Next get the first character of the plaintext and first character of the selected Thirukural.
- Then obtain the equivalent code of these letters from the Prl code and add them and find the modulo 64 value and we will get a number.
- Obtain the appropriate character from Prl code according to the number obtained.

- Repeat the process until all the characters in plain text are decrypted.
- In the reverse process of the same steps we would get back the plain text again.

2.1.2.5 Des:

- Obtain the plain text and encrypt using the transposition cipher.
- Again encrypt 16 times using Classical product cipher Algorithm.
- The final output is the cipher text.
- For the process of decryption do the above steps in the reverse order.

2.1.3 Compression:

This compression technique converts the cipher text into six bit code which can be transmitted in the network this technique is being is designed for alphabets (upper/lower case), numerals, special Characters (“, $).It provides a two bit compression when compared to conventional ASCII code.

Algorithm for Compression and Decompression

- Obtain the cipher text from the encryption algorithm.
- Now split them into single characters and store them in an array.
- Find the equivalent code for each character which is stored in the array and store them in an output array.
- So we can store five characters in a single integer value.
- For the process of decompression we are then splitting back the data into six bits and finding the equivalent plaintext.

2.1.4 Intelligent Software:

It is an intelligent software which provides a proper security for the data's which are transmitted in the network; this software changes the encryption algorithm for each set of data.

This type of swapping of the algorithms provide an immense security to the data if one set of data is being trapped also we would be able to provide proper security to the next set of data's.

PROJECT PLAN

PROJECT PLAN

We have formulated a perfect strategy to do process of multicasting the data's through the network we have also used encrypted mode of transmission of the data.

We are first checking the authenticity for each of the clients who are logging into our communication network we give them dedicated user name and password when they register them in new user part; we also assign them new passwords for their next login. In the next step we obtain the data's from user and then encrypt it, compress it and multicast it through the network to particular group of users.

In the receiving end we obtain the data and then decompress it, decrypt it by using proper decryption algorithms and retrieve back the plain text.

SOFTWARE REQUIREMENTS SPECIFICATIONS

SOFTWARE REQUIREMENTS SPECIFICATIONS

Purpose:

The purpose of this document is to describe all unambiguous and specific software requirements, interface requirements, performance requirements, design constraints and other specific attributes.

Specific Requirements:

Hardware requirements:

- ❖ LAN with Pentium PC's.
- ❖ 64 MB RAM.
- ❖ 10 GB HDD.

Software requirements:

- ❖ Windows 9x/2000 (Server/Client) OS with Java.

Performance Constraints:

Depends on the speed of the intranet connection, network traffic and client system.

Design Constraints:

Software Constraints

1. JDK 1.2
2. MS-ACCESS.

Description of software packages used

Java:

Java was developed by Sun Microsystems under interesting circumstances. Working on java originally began with the goal of creating a platform independent language OS for consumer electronics. The original intention was to use C++ but as the work progressed the java developers came to an idea of creating language that would serve them in a better way.

Java is both a programming language and an environment for executing programs written in java. Unlike traditional compilers that convert the source code into machine level instructions, the java compiler converts Java source code into byte code which runs in java virtual machine, which, has been developed for every operating system in the world. Java is suitable for many general programming tasks and in fact many of the tools are written in java. It is a compiler can be written in the same language.

The multithreaded concept of java has enhanced parallel processing environments to greater heights. For instance we can take the example of banking tasks which are simpler; it is due to its secure and robust nature.

AWT (Abstract Windowing Toolkit)

Awt acts as perfect tool for the GUI (Graphical User Interface) in java language. Awt provides a wide variety of components for building interactive graphical user environment. Here the top level menu acts as a container class and it has various sublevel components such as container, panel, window, frame, canvas.

Various Awt components which are available are

- Applets
- Frames
- Window
- Panel

Since applets do not support multicasting we use frames which are also a top level container which is capable of having all the components in it.

Frame is also a light weight component which can run any where; frames also support all the designer components such as buttons, checkboxes, text area, radio buttons, list boxes etc.

Awt also provides many functions which are built in the graphics class it has provisions to draw auto shapes such as rectangle, square, circle etc. We can use these shapes and functions for the process of building our graphical user environment.

JDBC

JDBC is a java API that documents a standard framework for dealing with tabular and relational data. While JDBC 2.0 begins a move to make SQL semi permanent to the programmer SQL is still the *lingua franca* of the database engines and represents a major industry victory in the effort to separate data from code.

SQL is a standardized language used to create, manipulate examine and manage relational database. A database is a container for tables which contains rows columns for holding the data's.

A database approximates a file system to appreciable level, a table approximates a file, a row approximates, and a record or structure a column approximates a field or variable. Because SQL is an application specific language a single statement can be very expressive and can initiate high level actions such as sorting and merging on an entire set of data. However we must connect to a database before sending SQL commands and each database vendor has different extensions of SQL.

M.S.ACCESS XP

MS ACCESS is a user friendly RDBMS. The various records are stored in an understandable way and the changes can be made to the tables very easily. The referential constraints and the data types available which make it very easily to do the restrictions to the data entry in the tables.

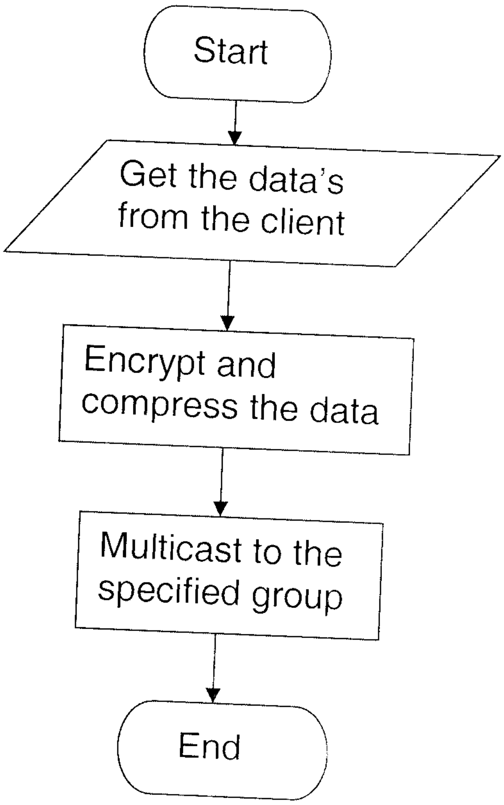
MS ACCESS coding with SQL provides a very user friendly environment to the user. The storing and retrieving of data's is easier.

DESIGN DOCUMENT

DATAFLOW DIAGRAM

4.1 Dataflow Diagram

At each session


Start
Get the data's from the client
Encrypt and compress the data
Multicast to the specified group
End

At login

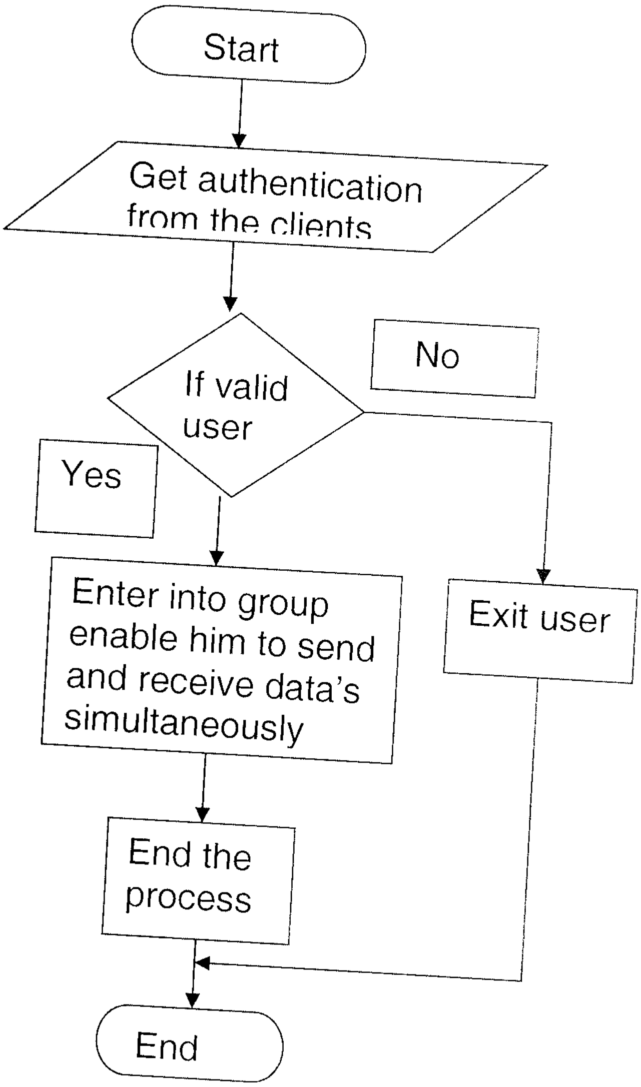

Start
Get authentication from the clients
If valid user
No
Yes
Enter into group enable him to send and receive data's simultaneously
Exit user
End the process
End

TABLE DESIGN

4.2 DESIGN OF TABLES

Kural table:

Serial number(number)	Kural (text)

Group data:

Port name (number)	IPaddress (text)

KBMSTab:

Username (text)	Password (text)	Group name (text)

PRODUCT TESTING

PRODUCT TESTING

Unit testing

Any product which is being developed has to be tested in each every phase of development. Unit testing provides the means for testing the product at each phase.

We have tested our software in each and every phase of development, it provides perfect requirement satisfaction. The product also provides requirement satisfaction even after integration of all the modules.

FUTURE ENHANCEMENT

FUTURE ENHANCEMENT

The project has been designed and developed flexibly according to the current objectives any new additional requirements may come in the future; this project is modifiable to the future enhancements.

Future development may be made in the direction of implementing the same in the internet and providing still more complex encryption techniques.

CONCLUSION

CONCLUSION

The Knowledge Based Multicasting has significantly provided secure and effective means of multicasting the data's in the network. It also uses the network bandwidth effectively. The project has emphasized greatly on our objectives and the result obtained is fruitful and convincing.

BIBLIOGRAPHY

BIBLIOGRAPHY

1. Special edition Using java2 PHI Dec-2000
- Joseph L. Weber

2. Java2 Complete Reference TMH Apr-1999
- Patrick Naughton

3. Java programming Bible Ver-2 IDG Jan-2000
- Aaron Walsh & John Fronckowiak

APPENDIX

SOURCE CODE

```
                    // Classical Product cipher //
import java.io.*;
import PrPack.*;

public class ClsicPrdCyp
{
    public String Classic, Cipher, Plain, Key;
    public int Len, KeyArrVal[];
    public char PrlCode[] = {'A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'J', 'K', 'L', 'M',
                             'N','O', 'P', 'Q', 'R', 'S', 'T', 'U', 'V', 'W', 'X', 'Y', 'Z',
                             ' ', '0', '1', '2', '3', '4', '5', '6', '7', '8', '9', 'a', 'b', 'c',
                             'd', 'e', 'f', 'g', 'h', 'i', 'j', 'k', 'l', 'm', 'n', 'o', 'p', 'q', 'r',
                             's', 't', 'u', 'v', 'w', 'x', 'y', 'z'};
    public ClsicPrdCyp(int a, String C) throws IOException
    {
        Classic = C;
        Cipher = C;
        Len = a;

    //Getting the Key from Database
        PrPack.SelectRow SRow = new PrPack.SelectRow(Len);
        Key = SRow.GetRow();

    //Finding CodeValue for each Character in the Key
        char PrArr[] = new char[512];
        KeyArrVal = new int[Key.length()];
        PrArr = Key.toCharArray();
        for (int i=0; i<Key.length(); i++)
        {
            for(int j=0; j<63; j++)
                if (PrArr[i] == PrlCode[j])          KeyArrVal[i] = j;
        }
    }

    public String EnCrypt()
    {
        char PrArr[] = new char[512];

    //Adding Key Value
        PrArr = Classic.toCharArray();
        int k=0;
        for (int i=0; i<Classic.length(); i++)
        {
            if (i > Key.length())  k=0;
            for(int j=0; j<63; j++)
            {
                if (PrArr[i] == PrlCode[j])
                {
                PrArr[i] = PrlCode[((j+KeyArrVal[k++])%63)];     break;
                }
            }
        }
        Cipher = new String(PrArr);
        return (Cipher);
    }

    public String DeCrypt()
    {
        char PrArr[] = new char[512];

    //Subtracting the Key Value
        PrArr = Cipher.toCharArray();
        int k=0;
        for(int i=0; i<Cipher.length(); i++)
        {
            if (i > Key.length()) k=0;
            for(int j=0; j<63; j++)
            {
                if(PrArr[i] == PrlCode[j])
                {
                    if (j >= KeyArrVal[k])
                    PrArr[i] = PrlCode[((j-KeyArrVal[k++])%63)];
                    else
                    PrArr[i] = PrlCode[63-((KeyArrVal[k++]-j)%63)];
                    break;
                }
            }
        }
        Plain = new String(PrArr);
        return (Plain);
    }

    public static void main(String Ar[]) throws IOException
    {
        PrPack.RandGenNo R = new PrPack.RandGenNo();
        int p = R.GetRnd()%1330;

        ClsicPrdCyp c = new ClsicPrdCyp(p, "Hai Sweety how are u");
        System.out.println("  -->" + c.EnCrypt());
    }
}
//-----------------------------------------------------------------------------------------------//
```

```
        //      Program for Swapping Encryption         //
import java.io.*;

public class EncrypSwap
{
    String SwapStr, Cipher, Plain;

    public EncrypSwap(String C) throws IOException
    {
        SwapStr = C;
        Cipher = C;
    }

    public String EnCrypt()
    {
        char PrArr[] = new char[512];
        String WrdsArr[] = new String[33];

        PrArr = SwapStr.toCharArray();
        int Start = 0, j = 0;

        // For Splitting the String into Words
        try
        {
            for (int i=0; i<SwapStr.length();)
            {
                while(PrArr[i++] != ' ' ){}
                WrdsArr[j++] = SwapStr.substring(Start, --i);
                Start = ++i;
            }
        }
        catch(ArrayIndexOutOfBoundsException ArrExcp)
        {
            WrdsArr[j] = SwapStr.substring(Start);
        }
  //Words Splitting End

  //Code for Swapping
      // For Reversing
      for (int i=0; i<=j; i++)
      {
          StringBuffer WordArr = new StringBuffer(WrdsArr[i]);
          WordArr.reverse();
          WrdsArr[i] = WordArr.toString();
      }
```

```
        //For Swapping
        int k=j;
        for(int i=0; i<=(j/2); i++)
        {
            String Temp = WrdsArr[i];
            WrdsArr[i] = WrdsArr[k];
            WrdsArr[k--] = Temp;
        }
    //End of Swapping

        Cipher = new String(WrdsArr[0]);
        for (int i=1; i<=j; i++)
            Cipher = Cipher.concat(" " + WrdsArr[i]);

        //System.out.println("Cipher Text is : " + Cipher);
        return(Cipher);
    }

    public String DeCrypt()
    {
        char PrArr[] = new char[512];
        String WrdsArr[] = new String[33];

        PrArr = Cipher.toCharArray();
        int Start = 0, j = 0;

    // For Splitting the String into Words
        try
        {
            for (int i=0; i<Cipher.length();)
            {
                while(PrArr[i++] != ' ' ){}
                WrdsArr[j++] = Cipher.substring(Start, --i);
                Start = ++i;
            }
        }
        catch(ArrayIndexOutOfBoundsException ArrExcp)
        {
            WrdsArr[j] = Cipher.substring(Start);
        }
    //Words Splitting End

    //Code for Swapping

        //For Reversing
        for (int i=0; i<=j; i++)
```

```
                {
                        StringBuffer WordArr = new StringBuffer(WrdsArr[i]);
                        WordArr.reverse();
                        WrdsArr[i] = WordArr.toString();
                }
                //For Swapping
                int k=j;
                for(int i=0; i<=(j/2); i++)
                {
                        String Temp = WrdsArr[i];
                        WrdsArr[i] = WrdsArr[k];
                        WrdsArr[k--] = Temp;
                }
        //End of Swapping

                Plain = new String(WrdsArr[0]);
                for (int i=1; i<=j; i++)
                        Plain = Plain.concat(" " + WrdsArr[i]);

                //System.out.println("Actual Text is : " + Plain);
                return (Plain);
        }
}

//--------------------------------------------------------------------------------------------------//

        //        Program for Sustitutional Cipher         //

import java.io.*;
import java.lang.*;

public class SubSt
{
        public int b, SubRndNo, Len;
        public char PrlCode[] = {'A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'J', 'K', 'L', 'M',
                                 'N','O', 'P', 'Q', 'R', 'S', 'T', 'U', 'V', 'W', 'X', 'Y', 'Z'
                                  ' ', '0', '1', '2', '3', '4', '5', '6', '7', '8', '9', 'a', 'b',
                                 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j', 'k', 'l', 'm', 'n', 'o',
                                 'p', 'q', 'r', 's', 't', 'u', 'v', 'w', 'x', 'y', 'z'};
        public String Plain, Cipher, Pr;

        public SubSt(String C) throws IOException
        {
                Pr = C;
                PrPack.RandGenNo Rgn = new PrPack.RandGenNo();
                Len = (int)(Rgn.GetRnd()%16);
        }

        public SubSt(String C, int a)
        {
                Pr = C;
                Len = a;
        }

    //--------------------------------------------------------------
        public String EnCrypt()
        {
                PrPack.CTransPos Ctp = new PrPack.CTransPos(Pr, Len);
                Plain = Ctp.EnCrypt();

                char SplitChars[] = new char[Plain.length()];
                SplitChars = Plain.toCharArray();

                PrPack.RandGenNo RNo = new PrPack.RandGenNo();
                SubRndNo = (int)((RNo.GetRnd())%Plain.length());
                int i,j,Temp;
                Temp = SubRndNo;

                for (i=0; i<Plain.length(); i++)
                {
                        if (SplitChars[i] == '$')
                        {
                                SubRndNo += Temp;
                                continue;
                        }

                        for(j=0; j<63; j++)
                                if(SplitChars[i] == PrlCode[j])        break;

                        if ((j+SubRndNo) > 62)
                                SplitChars[i] = PrlCode[((j+SubRndNo)%63)];
                        else
                                SplitChars[i] = PrlCode[j+SubRndNo];

                        SubRndNo += Temp;
                }
                SubRndNo = Temp;

                Cipher = new String(SplitChars);
                return (Cipher);
        }
//--------------------------------------------------------------
```

```
        public String DeCrypt()
        {
                int i, j, Temp;
                char SplitChars[] = new char[Cipher.length()];

                SplitChars = Cipher.toCharArray();
                Temp = SubRndNo;

                for (i=0; i<Cipher.length(); i++)
                {
                        //System.out.print("" + SplitChars[i]);
                        if (SplitChars[i] == '$')
                        {
                                SubRndNo += Temp;
                                continue;
                        }

                        for(j=0; j<63; j++)
                                if(SplitChars[i] == PrlCode[j])         break;

                        if (j > SubRndNo)
                                SplitChars[i] = PrlCode[j-SubRndNo];
                        else
                                SplitChars[i] = PrlCode[63-((SubRndNo-j)%63)];

                        //System.out.println(" ----> " + SplitChars[i]);
                        SubRndNo += Temp;
                }
                SubRndNo = Temp;

                Cipher = new String(SplitChars);
                PrPack.CTransPos Ctp = new PrPack.CTransPos(Cipher, Len);
                Plain = Ctp.DeCrypt();

                return (Plain);
        }
}

//------------------------------------------------------------------------------------------------//
```

```
                          // Program for transposition cipher //
import java.io.*;
import java.lang.*;

public class TransPos
{
        public String Pr;
        public int i, Len;
        private String Cipher, Plain;

//-----------------------------------------------------------------------
        public TransPos(int a, String c) throws IOException
        {
                Pr = c;
                Cipher = c;
                Len = a;
        }

//-----------------------------------------------------------------------
        public String EnCrypt()
        {
                String Prarr[] = new String[(Pr.length()/Len)+1];
                i = 0;

        // For Splitting the Strings into Sub(Small)Strings
                int j;
                for(j=0; j<Pr.length()-Len; j+=Len)
                        Prarr[i++] = Pr.substring(j, j+Len);

                String Tmp = new String();
                Tmp = Pr.substring(j, Pr.length());
                if (Tmp.length() <= Len)
                {
                        for(int k=Tmp.length(); k<Len; k++)
                                        Tmp = Tmp.concat(" ");
                        Prarr[i] = Tmp.replace(' ', '$');
                }

        // For Reversing the SubString(s)
                for(int k=0; k<=i; k++)
                {
                        if ((k%2) == 0)
                        {
                                char Temp[] = new char[Len];
                                char PrPt[] = new char[Len];        int v=Len-1;
                                Temp = Prarr[k].toCharArray();
                for(j=0; j<Len; j++)
                    PrPt[v--] = Temp[j];

                Prarr[k] = new String(PrPt);
            }
        }

    // For Matrix Coversion
        char Temp[][] = new char[Len][i+1];
        for(int k=0; k<=i; k++)
        {
            char TempArr[] = new char[Len];
            TempArr = Prarr[k].toCharArray();
            for(j=0; j<Len; j++)
                Temp[j][k] = TempArr[j];
        }

    // For getting Final Cipher Text
        Cipher = new String(Temp[0]);
        for(int k=1; k<Len; k++)
            Cipher = Cipher.concat(new String(Temp[k]));

        return (Cipher);
    }
/*-----------------------------------------------------------------------*/
    public int Geti()
    {
        return (i);
    }

    public void Seti(int p)
    {
        i = p;
    }

/*-----------------------------------------------------------------------*/
    public String DeCrypt()
    {
        String Prarr[] = new String[Len];
        int k, j=0;

    // For Converting to Matrix Format
        for (k=0; k<Len-1; k++)
        {
            Prarr[k] = Cipher.substring(j, j+i+1);
            j += (i+1);
        }
        Prarr[k] = Cipher.substring(j, Cipher.length());

        char Tmp[] = new char[i];
        char Temp[][] = new char[Len][i+1];
        for(k=0; k<Len; k++)
        {
            Tmp = Prarr[k].toCharArray();
            for(int h=0; h<i+1; h++)
                Temp[k][h] = Tmp[h];
        }

    // For getting Plain Text in Reverse order of SubStrings
        int m=0;
        char[] Tmp1 = new char[Len];
        for(k=Len-1; k>=0; k--)
            Tmp1[m++] = Temp[k][0];
        Plain = new String(Tmp1);

        for(j=1; j<=i; j++)
        {
            m=0;
            if ((j%2)==0)
                for(k=Len-1; k>=0; k--)
                    Tmp1[m++] = Temp[k][j];
            else
                for(k=0; k<Len; k++)
                    Tmp1[m++] = Temp[k][j];

            Plain = Plain.concat(new String(Tmp1));
        }
        Plain = Plain.replace('$', ' ');

        return (Plain);
    }
}

//-----------------------------------------------------------------------------------------------------//
```

```
                        //  Program for Data Encryption Std    //
import java.io.*;
import java.sql.*;

public class DES
{
        public String Plain, Cipher, Pr;
        public int TRndNo, CRndNo, iVal, c=0;
        public int KeyArr[] = new int[2];

        public DES(String C) throws Exception
        {
                PrPack.RandGenNo Rgn = new PrPack.RandGenNo();
                TRndNo = (int)(Rgn.GetRnd()%16);
                if (TRndNo == 0) TRndNo = 1;
                KeyArr[c++] = TRndNo;

                Rgn = new PrPack.RandGenNo();
                CRndNo = (int)(Rgn.GetRnd()%1330);
                KeyArr[c++] = CRndNo;

                Pr = C;
        }

        public int[] GetArr()
        {
                return(KeyArr);
        }

        public void SetArr(int[] a)
        {
                TRndNo = a[0];
                CRndNo = a[1];
        }

        public String EnCrypt() throws IOException
        {
                Ciphers.TransPos Tp = new Ciphers.TransPos(TRndNo, Pr);
                String Temp = Tp.EnCrypt();
                iVal = Tp.Geti();

   Ciphers.ClsicPrdCyp Cpc;
                for (int i=1; i<=16; i++)
                {
                        Cpc = new Ciphers.ClsicPrdCyp(((CRndNo*i)%1330),
Temp);
```

```
                        Temp = Cpc.EnCrypt();
                }

    Ciphers.EncrypSwap Es = new Ciphers.EncrypSwap(Temp);
                Cipher = Es.EnCrypt();
                return(Cipher);
        }

        public String DeCrypt() throws IOException
        {
                Ciphers.EncrypSwap Es = new Ciphers.EncrypSwap(Cipher);
                String Temp = Es.DeCrypt();

                Ciphers.ClsicPrdCyp Cpc;
                for (int i=16; i>=1; i--)
                {
                        Cpc = new Ciphers.ClsicPrdCyp(((CRndNo*i)%1330),
 Temp);

                        Temp = Cpc.DeCrypt();
                }

                Ciphers.TransPos Tp = new Ciphers.TransPos(TRndNo, Temp);
                Tp.Seti(iVal);
                Plain = Tp.DeCrypt();
                return(Plain.trim());
 }
public static void main (String h[])
        {
                DES s=new DES();
                s.EnCrypt();
                s.DeCrypt();
        }
}

//--------------------------------------------------------------------------------------------//
```

```
          ///*...  PROGRAM FOR COMPRESSION AND DECOMPRESSION  ...*///

package Ciphers;

import java.io.*;
import java.lang.*;

public class Comp
{

  public int i=0,j=0,k,m,s=0,kk=0,ss=0,length,C,laks=0,s1=0,rem,c;
  public String h, Dec;
  public int Comps[]=new int [200],h1[]=new int[250],b[]=new int [250];
  public int E[]={0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22.
            23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,
            43,44,45,46,47,48,49,50,51,52,53,54,55,56,57,58,59,60,61,62,63};

  public char S[]={' ','A','B','C','D','E','F','G','H','I','J','K','L','M','N','O','P',
             'Q','R','S','T','U','V','W','X','Y','Z','0','1','2','3','4','5','6',
             '7','8','9','a','b','c','d','e','f','g','h','i','j','k','l','m','n',
             'o','p','q','r','s','t','u','v','w','x','y','z','$'};

        public Comp()
        {       }
 public Comp(String pr)
        {
                h = pr;
                Dec = new String();
        }

 public int[] Split()throws Exception
        {
        //SPLIT THE TEXT INTO CHARACTERS
                k=h.length();
                m=k;
                length=(m/5);
                for(int v=0;v<200;v++)
                {
                        Comps[v]=0;
                }

                char[]l=h.toCharArray();
                do
                {
                        if(l[i]==S[j])
                        {
```

```
                Comps[i]=E[j];
                i=i+1;
                j=0;
                k=k-1;
        }
        else    j=j+1;
}while(k!=0);

//COMPRESS THE TEXT
j=0;    i=0;

for (int y=0;y<250;y++)
{
        b[y]=0;
}
laks=length;

if (length!=0)
{
        do
        {
                b[kk]=(Comps[i]<<25);i++;
        b[kk]=b[kk]|(Comps[i]<<19);i++;
        b[kk]=b[kk]|(Comps[i]<<13);i++;
                b[kk]=b[kk]|(Comps[i]<<7);i++;
                b[kk]=b[kk]|(Comps[i]<<1);i++;
                kk=kk+1;
                laks=laks-1;
        }while(laks!=0);
}
rem=(m % 5);
if(rem>0)
{
        switch(rem)
        {
        case 1:
                b[kk]=Comps[i];
                b[kk]=(b[kk]<<25);                      break;
        case 2:
                b[kk]=b[kk]|(Comps[i]<<25);i++;
                b[kk]=b[kk]|(Comps[i]<<19);                             break;
        case 3:
                b[kk]=Comps[i];
        b[kk]=(b[kk]<<25);i++;
                b[kk]=b[kk]|(Comps[i]<<19);i++;
                b[kk]=b[kk]|(Comps[i]<<13);                 break;
```

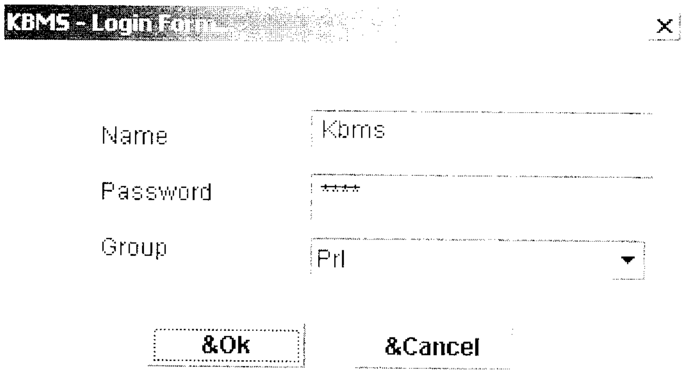
```
            case 4:
                b[kk]=Comps[i];
                b[kk]=(b[kk]<<25);i++;
                b[kk]=b[kk]|(Comps[i]<<19);i++;
                b[kk]=b[kk]|(Comps[i]<<13);i++;
                b[kk]=b[kk]|(Comps[i]<<7);                    break;
            }
        }

        return (b);
    }

//DECOMPRESS THE TEXT
    public String decompress(int[] dec)
    {
        while(dec[i] != 0)      b[i++] = dec[i];

        if(length !=0)
        {
            for( s1=0;s1<(length);s1++)
            {
                h1[ss]=b[s1]&2113929216;
                h1[ss]=h1[ss]>>25;ss++;
            h1[ss]=b[s1]&33030144;
                h1[ss]=h1[ss]>>19;ss++;
                h1[ss]=b[s1]&516096;
                h1[ss]=h1[ss]>>13;ss++;
                h1[ss]=b[s1]&8064;
                h1[ss]=h1[ss]>>7;ss++;
                h1[ss]=b[s1]&126;
                h1[ss]=h1[ss]>>1;ss++;
            }
        }
        if (rem !=0 )
        {
            h1[ss]=b[s1]&2113929216;
            h1[ss]=h1[ss]>>25;ss++;
            h1[ss]=b[s1]&33030144;
            h1[ss]=h1[ss]>>19;ss++;
            h1[ss]=b[s1]&516096;
            h1[ss]=h1[ss]>>13;ss++;
            h1[ss]=b[s1]&8064;
            h1[ss]=h1[ss]>>7;ss++;
            h1[ss]=b[s1]&126;
            h1[ss]=h1[ss]>>1;ss++;
        }
        if (length !=0)
        {
            for( c=0;c<(length*5);c++)
            {
                for(int ii=0;ii<63;ii++)
                {
                    if( h1[c]==E[ii])        Dec += S[ii];
            }
            }
        }
        //REGROUP THE CHARACTERS
        C=length*5;
        if (rem !=0)
        {
            for(c=C ;c<(C+5);c++)
            {
                for(int jj=0;jj<63;jj++)
                {
                    if(h1[c]==E[jj])        Dec += S[jj];
            }
            }
        }
        return(Dec);
    }
}
```

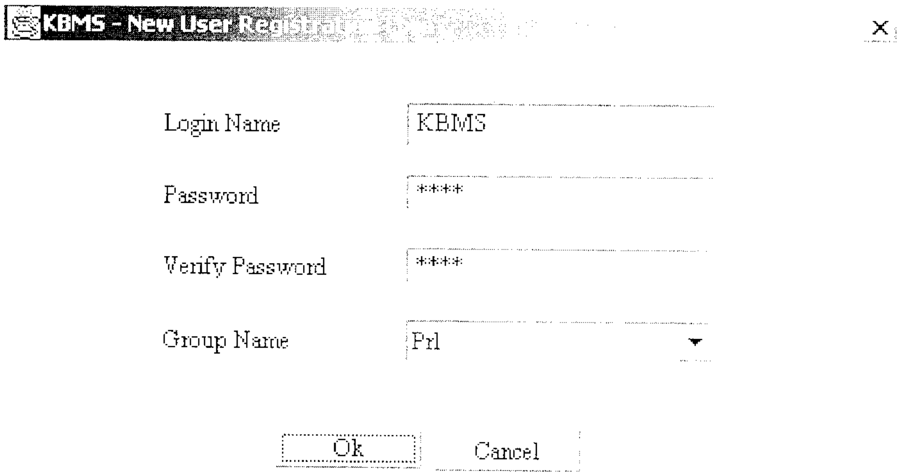
SAMPLE RESULTS



Name Kbms

Password ****

Group Prl



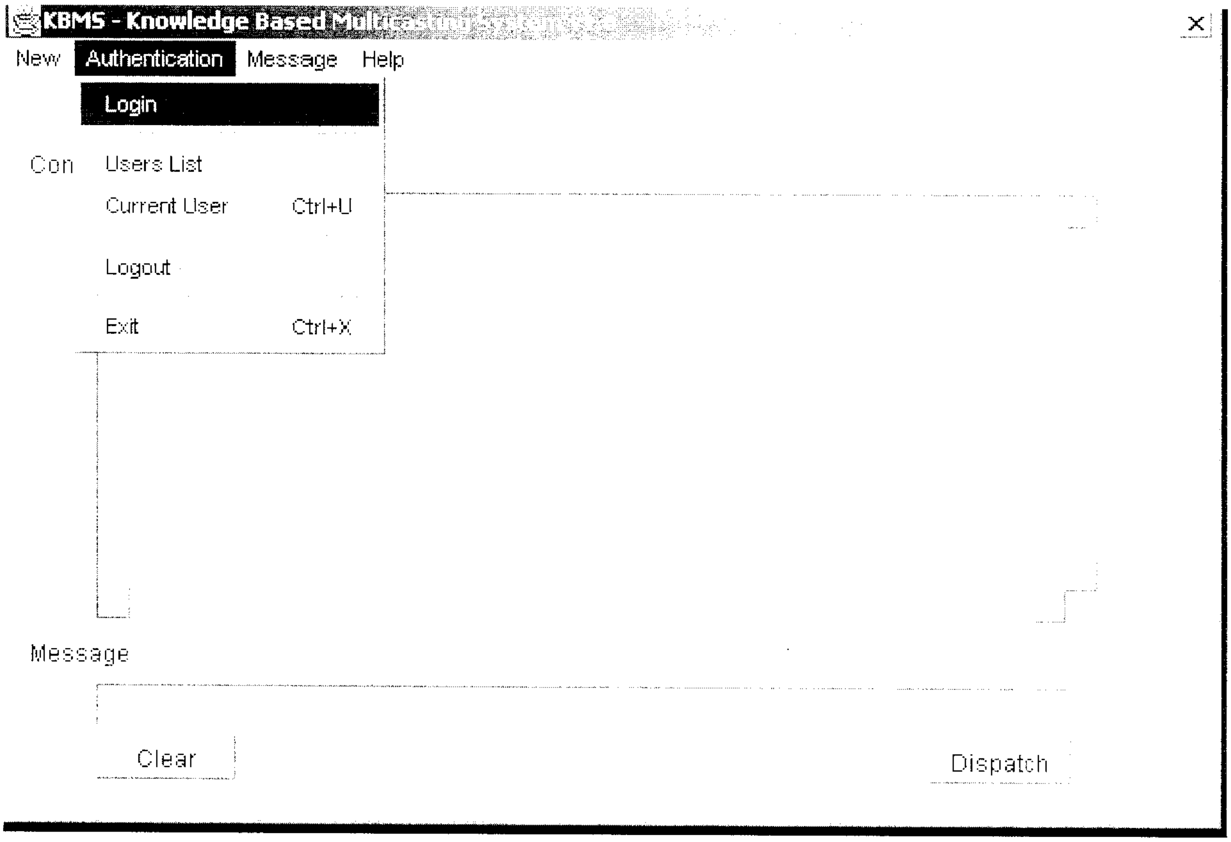
KBMS - New User Registra

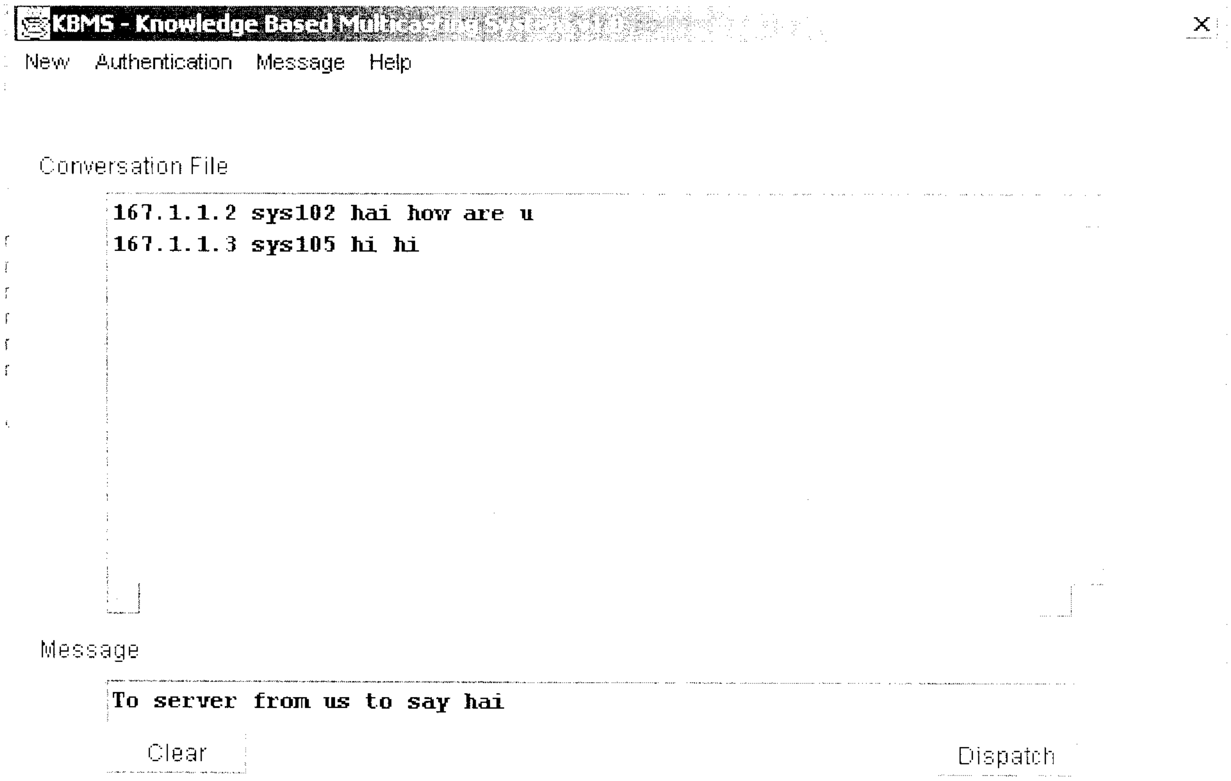
Login Name KBMS

Password ****

Verify Password ****

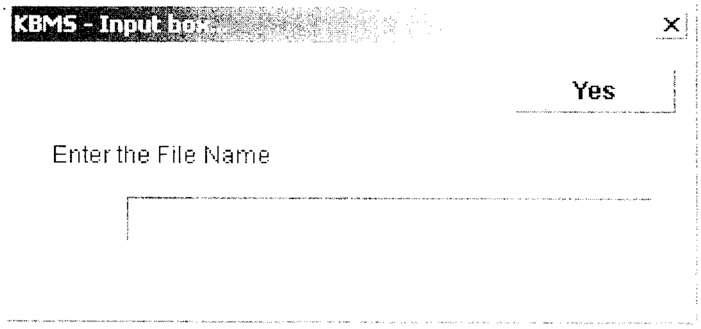
Group Name Prl

KBMS - Knowledge Based Multicasting System
New
Authentication
Message
Help
Login
Users List
Current User Ctrl+U
Logout
Exit Ctrl+X
Con
Message
Clear
Dispatch



```
167.1.1.2 sys102 hai how are u
167.1.1.3 sys105 hi hi
```

To server from us to say hai



Enter the File Name