IMPLEMENTING SYMMETRIC CRYPTOGRAPHIC TECHNIQUES IN ONLINE FEE PAYMENT SYSTEM OF NIT ROURKELA

Kaustav Nag
Mihir Birua

Department of Computer Science and Engineering
National Institute of Technology, Rourkela
Rourkela – 769 008, Odisha, India

IMPLEMENTING SYMMETRIC CRYPTOGRAPHIC TECHNIQUES IN ONLINE FEE PAYMENT SYSTEM OF NIT ROURKELA

A Thesis Submitted on
14th May, 2012
in partial fulfilment of the requirements for the degree of

Bachelor of Technology

in

Computer Science and Engineering

by

Kaustav Nag
(Roll No-108CS008)
&
Mihir Birua
(Roll No-108CS042)

Under the Guidance of
Prof. Sanjay Kumar Jena

Department of Computer Science and Engineering
National Institute of Technology, Rourkela
Rourkela – 769008, Odisha, India

Computer Science and Engineering
National Institute of Technology, Rourkela
Rourkela – 769008, Odisha, India

Certificate

This is to certify that the work in the thesis entitled '***Implementing Symmetric Cryptographic Techniques in Online fee Payment System of NIT Rourkela***' submitted by Kaustav Nag and Mihir Birua is an original research work carried out by them under my supervision and guidance for partial fulfilment of the requirements for the award of the degree of Bachelor of Technology in Computer Science and Engineering during the session 2011-12 in Department of Computer Science and Engineering, NIT Rourkela.

To the best of my knowledge, the matter embodied in this report has not been submitted to any other University/Institute.

Date -14/05/2012

Rourkela

(Dr. S. K. Jena)

Department of Computer Science and Engineering

NIT Rourkela

Acknowledgments

We express our profound gratitude and indebtedness to **Dr. S. K. Jena**, Department of Computer Science and Engineering, National Institute of Technology, Rourkela, for introducing the present topic and for his inspiring intellectual guidance, constructive criticism and valuable suggestions throughout our project.

We are also thankful to Prof. S. K. Panigrahy, Department of Computer Science and Engineering for guiding and motivating us throughout the project

We would also like to thank our parents and colleagues for their support and motivation for completion of this project.

Date-14/05/2012
Rourkela

Kaustav Nag
(Roll No-108CS008)

Mihir Birua
(Roll No-108CS042)

Abstract

Cryptography protects the information in network and reduces the risk of security breaches from hackers. One of its most important applications is in the financial sector and the e-commerce over the Internet which requires secure handling of data during transactions. The online fee payment system of NIT Rourkela has serious vulnerabilities which makes it prone to attacks by hackers. In this project our objective is to highlight the various drawbacks in the system and to suggest ways to eliminate those flaws using symmetric cryptographic techniques which encrypts data at sender side and decrypts ciphertext at receiver side using the same shared key.

Contents

List of Figures

1. Introduction

Over the last decade the world has seen an astounding growth of information technology that has resulted in significant advances in cryptography to protect the integrity and confidentiality of data.

1.1 Problem Statement

To secure Online Fee Payment Website of NIT Rourkela by using symmetric key cryptographic techniques.

1.2 Drawbacks of current payment system

The online fee payment system of NIT Rourkela has serious vulnerabilities which can be exposed easily. There is no user authentication due to absence of login page during the fee payment. Also the fee amount to be paid is directly visible in the address bar when one is directed from NIT website to Online SBI. This raises various security concerns as data can be easily manipulated. Attacker can change the fee amount resulting in loss of data integrity. Then the bank accepts this changed amount and redirects to NIT website which displays a receipt acknowledging that the fee payment is successful and that the original amount has been received while it receives the changed amount. Our primary goal is to eliminate this flaw in the online payment process.

1.3 Challenges

When a student selects the fees, the fee amount is going to be encrypted and the cipher text will be sent to the bank website where it needs to be decrypted. Since we cannot code for

decryption on the Online State Bank of India website as we do not have necessary privileges, we will be creating a virtual bank to simulate the payment.

1.4 Overview of Symmetric-Key Cryptography

Symmetric-key algorithms are a class of cryptographic algorithms that use shared key for both encryption of plaintext and decryption of ciphertext. The encryption key is trivially related to the decryption key. They might be identical or there is a simple transformation to go between the two keys. Both the sender and receiver know the secret key is advance.

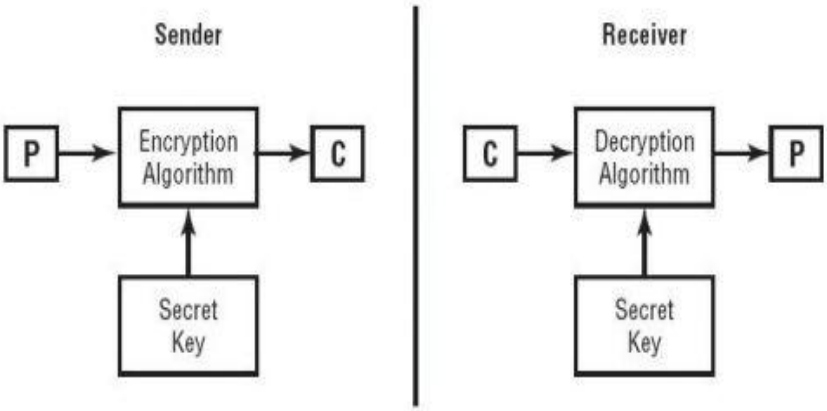

Fig 1.1 Symmetric key Cryptosystem

The symmetric encryption scheme has five ingredients:

- Plaintext: This is the intelligible message which needs to be encrypted and is fed to the algorithm as input.
- Encryption algorithm: It performs various permutations and substitutions on the plaintext.
- Secret Key: The exact substitutions and permutations performed depend on the key used which is fed to the algorithm as input.

- Ciphertext: It is the scrambled message produced by the encryption algorithm as output. The ciphertext depends on the plaintext and the key and is unintelligible.
- Decryption Algorithm: This is essentially the reverse of encryption algorithm. It takes the secret key and the ciphertext and produces the original plaintext.

There are two requirements for a symmetric key cryptosystem

1. It is assumed impractical to decrypt a message on the basis of the ciphertext and knowledge of the encryption/decryption algorithm. Only the key is to be kept secret and the algorithm is in public domain
2. Sender and the receiver must have obtained the secret key in a secure way. If the key is discovered by the attacker then he can read all communications.

1.5 Block Ciphers

In case of block ciphers; a block of input is encrypted together. The block size is fixed and the message is partitioned into blocks. In Electronic Codebook (ECB) each block is encrypted independently and there is no chaining and error propagation.

2. Data Encryption Standard (DES)

2.1 Overview

DES [1] is a block cipher which accepts a 64 bit plaintext and transforms it into 64 bit ciphertext. This ciphertext is then decrypted at the decryption side into 64-bit plaintext. Both encryption and decryption takes place with the help of same 56-bit key.

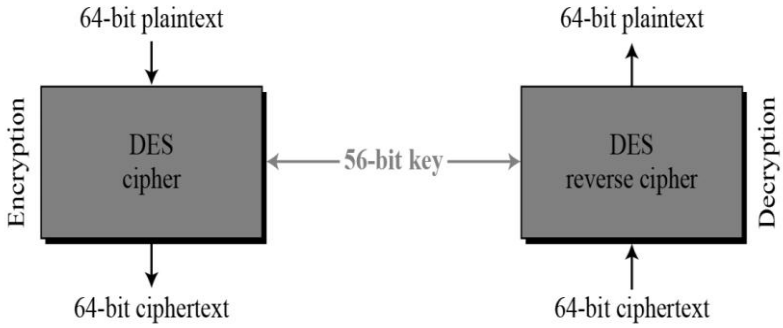

Fig 2.1: Encryption and Decryption with DES

2.2 Structure

The encryption process consists of two permutations which are known as initial and final permutation along with 16 feistal rounds whose each round uses a different 48-bit key generated from the same cipher key.

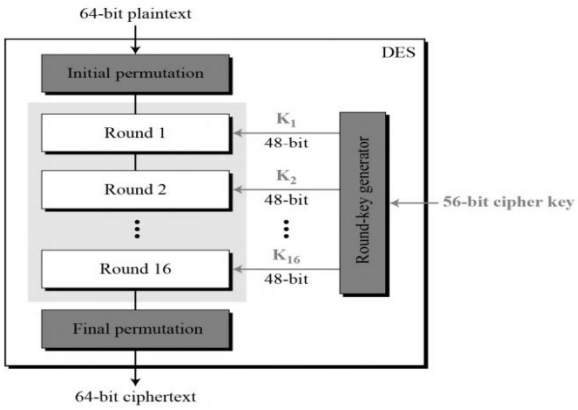

Fig 2.2: General Structure of DES

2.3 Initial Permutation

The block to be encrypted is first subjected to an initial permutation IP followed by complex key dependent computation process then subjected to inverse initial permutation IP^{-1}. The two permutations are inverses of each other and have no cryptographic significance. These complex permutations will thwart any software simulation when DES is implemented in hardware.

2.4 Rounds

There are 16 rounds in DES. Every DES round is a fiestal cipher consisting of a mixer and swapper.

L_{I-1} : Leftmost 32 bits after initial permutation.

R_{I-1}: Rightmost 32-bits after initial permutation.

K_I : 48–bit Key

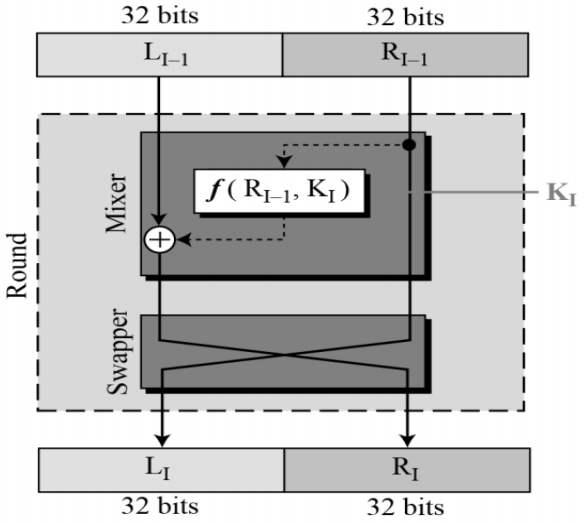

Fig 2.4: A round in DES

2.4.1 Expansion

The 32-bit R_{I-1} is expanded to 48 as key length K_I is 48 bits. So R_{I-1} is divided into 8 sections each of 4 bit. The output consists of eight sections each of 6-bits, each of which contains a copy of 4 corresponding input bits. In addition to that, it contains a copy of the immediately adjacent bit from each of the input sections to either side.

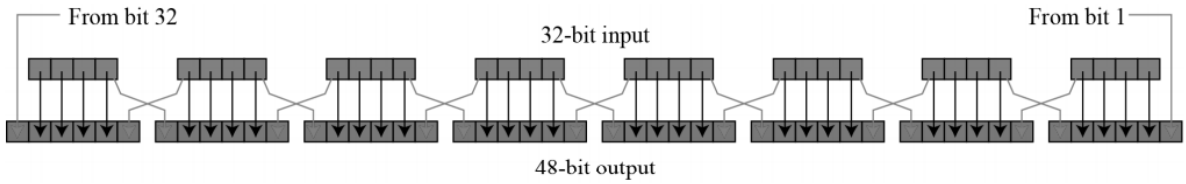

Fig 2.4.1: Expansion Permutation

2.4.2 Whitener (XOR)

After expansion permutation is done, DES does the XOR operation on the expanded right section with the 48-bit round key. The output of L'R' of an iteration with input LR is given by

$L' = R$

$R' = L \oplus f(R_{n-1}, K_n)$

2.4.3 Substitution

The 48-bit key obtained after the second operation is again divided into 8 sections each consisting of 6-bits. The output of this operation is 8 sections of 4-bit. These S-boxes do the real mixing. Each S-box has its own table

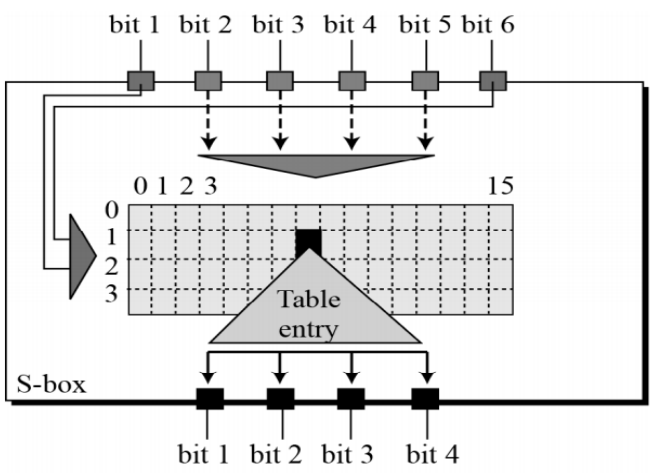

Fig 2.4.3: S Box 1 in detail

2.4.5 Permutation

According to a fixed permutation, the P-Box, the 32 outputs from the S-boxes are rearranged. This is designed so that, each S-box's output bits are spread across 6 different S boxes in the next round, after the expansion.

2.5 Key Generation

Each round of DES uses a different 48-bit round key. These 48-bit keys are generated from 56-bit cipher text. The cipher key length is 64-bits in which extra 8-bit are the parity bits.

2.5.1 Parity Drop

From the 64-bit cipher key, it drops the parity bits (bits 8,16,24,…64) and permutes the rest of the bit according to the table.

57	49	41	33	25	17	09	01
58	50	42	34	26	18	10	02
59	51	43	35	27	19	11	03
60	52	44	36	63	55	47	39
31	23	15	07	62	54	46	38
30	22	14	06	61	53	45	37
29	21	13	05	28	20	12	04

Fig 2.5.1: Parity Bit Drop table

2.5.2 Shift Left

Following the permutation, the 56-bit key is divided into two blocks each of 28-bit. Each part is shifted left one or two bits depending on the table.

Round	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Bit shifts	1	1	2	2	2	2	2	2	1	2	2	2	2	2	2	1

Fig 2.5.2: Number of Bit Shifts

2.5.3 Compression D-Box

The 56-bit key is then compressed into 48-bit key to be used in each feistal round.

2.6 Cipher and reverse Cipher

Each encryption in DES consists of 16 feistal rounds using the round keys K_1 to K_{16}. The last round in DES is a bit different from the previous 15 rounds. It has only mixer and no swapper. At decryption site the keys are applied in reverse order.

3. Advanced Encryption Standard (AES)

3.1 Overview

AES [2] is a symmetric key block cipher published by NIST as FIPS 197. It encrypts as well as decrypts a plaintext blocks of size 128-bits. The number of rounds varies with the key size.

Number of rounds	Key size
10	128
12	192
14	256

Fig 3.1: Relation between key length and number of rounds

3.2 High-level description of the algorithm

1. KeyExpansion

   128-bit round keys are derived from the cipher key using Rijndael's key schedule

2. Initial Round

   1. AddRoundKey—each byte of the state is XORed with the corresponding round key

3. Rounds

   1. SubBytes—It is a non-linear substitution step where each byte is replaced with another byte according to a lookup table.
   2. ShiftRows—It is a transposition step where each row of the state is shifted a certain number of steps cyclically.
   3. MixColumns—It is a mixing operation which combines the four bytes in each column in a state.

4. AddRoundKey

4. Final Round (no MixColumns)
    1. SubBytes
    2. ShiftRows
    3. AddRoundKey

To provide security, AES uses four types of transformations:

- Substitution
- Permutation
- Mixing
- Key-adding

3.3 Substitution

AES uses substitution which is done for each byte. It means if the two bytes are same then the resulting transformation will also be same. two invertible transformations.

3.3.1 SubBytes

The first transformation used at the encryption site is SubBytes. We interpret the byte as two hexadecimal digits whose left digit gives the row and right digit gives the column of the substitution table. The contents of each byte in a state change but the arrangement of the bytes remains the same. It is an intra-byte transformation

3.3.2 InvSubBytes

It is the inverse of SubBytes.

3.4 Permutation

In AES shifting is done to permute the bytes. In this case shifting transformation is done at Byte level and not at bit level.

3.4.1 ShiftRows

This transformation is done at encryption site.

3.4.2 InvShiftRows

This transformation is inverse of ShiftRow and is done at decryption site.

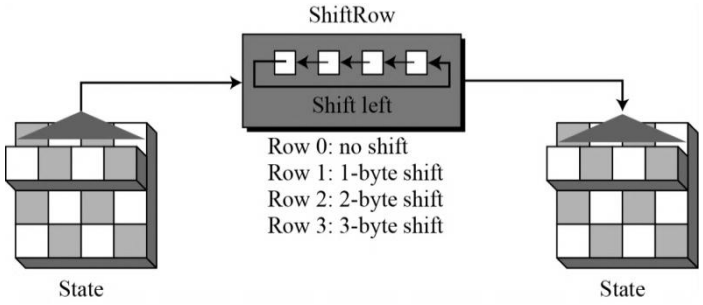

Fig 3.4: Shift rows

3.5 Mixing

Mixing is an interbyte transformation. It takes care of neighboring bites while doing transformation. The bits inside a byte are changed based on the bits in the neighboring bytes. To provide diffusion bytes should be mixed at the bit level. In order to make sure that each new byte generated is different from the old byte even if all four bytes are same, the old bytes are multiplied with a different constant and then mixing is done

3.5.1 MixColumns

It operates at the column level and each column of the state is transformed into a new column by performing a matrix multiplication of a constant matrix with the state column.

3.5.2 InvMixColumns

In this case the constant matrix is inverse of the one used in MixColumns

3.6 Key Adding

All the previous transformations are invertible. In this transformation, 128-bit key round key generated from cipher key is used.

AddRoundKey

In this transformation each column of the state is XORed with the corresponding key word.

3.7 Key Expansion

AES uses a key-expansion process to create round keys for each round. If the number of rounds is N_r, the key-expansion routine creates N_{r+1} 128-bit round keys from one single 128-bit cipher key.

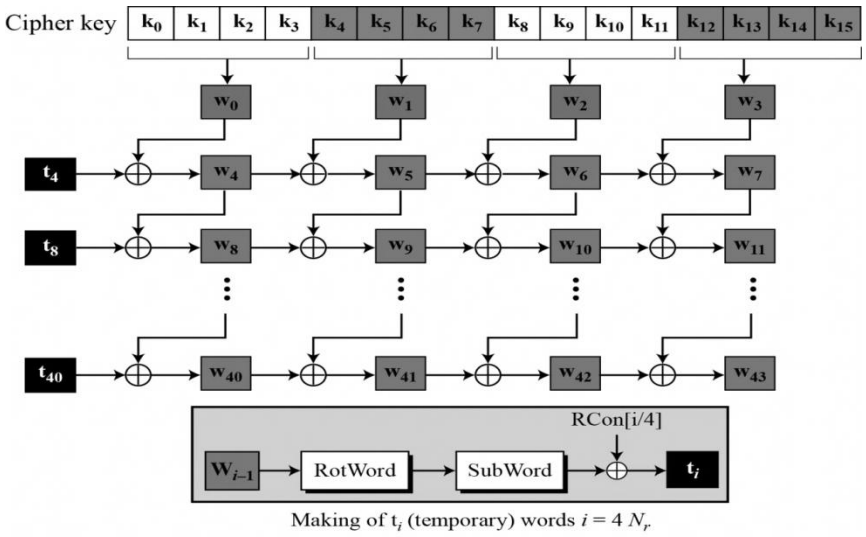

Fig 3.7: Key expansion in AES

3.7.1 RotWord

Rotate word is quite similar to ShiftRows transformation. The only difference is that it is applied to one row only. It takes an array of four byte and shifts each byte to the left

3.7.2 SubWord

Substitute word is quite similar to SubBytes transformations. The only difference is that it is applied to four bytes only.

3.7.3 Round Constants

It is a 4-byte value in which the rightmost three bytes are always zero

Round	*Constant (RCon)*	*Round*	*Constant (RCon)*
1	$(\underline{\mathbf{01}}\ 00\ 00\ 00)_{16}$	6	$(\underline{\mathbf{20}}\ 00\ 00\ 00)_{16}$
2	$(\underline{\mathbf{02}}\ 00\ 00\ 00)_{16}$	7	$(\underline{\mathbf{40}}\ 00\ 00\ 00)_{16}$
3	$(\underline{\mathbf{04}}\ 00\ 00\ 00)_{16}$	8	$(\underline{\mathbf{80}}\ 00\ 00\ 00)_{16}$
4	$(\underline{\mathbf{08}}\ 00\ 00\ 00)_{16}$	9	$(\underline{\mathbf{1B}}\ 00\ 00\ 00)_{16}$
5	$(\underline{\mathbf{10}}\ 00\ 00\ 00)_{16}$	10	$(\underline{\mathbf{36}}\ 00\ 00\ 00)_{16}$

Fig 3.7.3: Rcon constants

3.8 The AES Cipher

Since AES is a non Feistal cipher, each transformation or group of transformation has to be reversible. The round keys have to used in reverse order at the decryption site.

4. Proposed Approach

The website we developed uses Java Server Pages (JSP) on the client side and java Servlets on the server side. The server we used for our web application is Apache Tomcat. MySQL database was selected for storing data at the backend. We are encrypting the message using 64-bit DES and 128-bit AES and then transmitting the ciphertext through the unsecure channel.

4.1 Components of a Java web application

A servlet/ JSP engine is the software that allows the web servlet to work with servlets and JSPs.

The Java Enterprise Edition (Java EE) specifications describe how web servers interact with servlet/JSP engines. Tomcat is one such popular servlet/JSP engines.

For a servlet/JSP to work, it must have access to Java Development Kit (JDK) which comes as part of the Java Standard Edition (Java SE). Among other things, it contains the core Java class libraries, the Java compiler and the Java Runtime Environment (JRE).

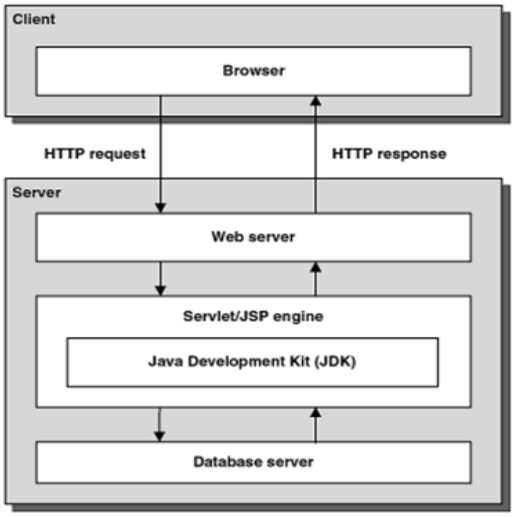

Fig 4.1: Components of Java Web Application

4.2 Tomcat Application Server

Tomcat is a free, open-source implementation of Java Servlet and Java Server Pages technologies developed under the Jakarta project at the Apache Software Foundation.

It is an application server from the Apache Software Foundation that executes Java servlets and renders Web pages that comprise Java Server Page coding.

Advantages

- Active open-source development effort
- Very current in terms of servlet API compliance
- Works on multiple Operating system

4.3 Databases

MySQL is an open source relational database management system which is fast, stable and easy to use and install. Two databases were created one for NITR online fee payment website and another for bank website. Studentdb database stores information about student as well as the fee details. Bank database stores data about account holders.

4.3.1 Tables on NITR Online Fee Payment Website

- Studentinfo

```
mysql> describe studentinfo;
+------------------+-------------+------+-----+---------+-------+
| Field            | Type        | Null | Key | Default | Extra |
+------------------+-------------+------+-----+---------+-------+
| rollno           | varchar(50) | NO   | PRI | NULL    |       |
| name             | varchar(50) | NO   |     | NULL    |       |
| password         | varchar(50) | NO   |     | NULL    |       |
| program          | varchar(10) | NO   |     | NULL    |       |
| department       | varchar(50) | NO   |     | NULL    |       |
| admissionYear    | varchar(8)  | NO   |     | NULL    |       |
| semester         | int(11)     | NO   |     | NULL    |       |
| hallnno          | varchar(50) | NO   |     | NULL    |       |
| permittedOutside | char(1)     | NO   |     | NULL    |       |
| lastLogon        | varchar(30) | NO   |     | NULL    |       |
+------------------+-------------+------+-----+---------+-------+
```

Fig 4.3.1a: Table studentinfo

- Fee_detail

```
mysql> describe fee_detail;
+-------------+-------------+------+-----+-------------------+-----------------------------+
| Field       | Type        | Null | Key | Default           | Extra                       |
+-------------+-------------+------+-----+-------------------+-----------------------------+
| ref_no      | int(11)     | NO   | PRI | NULL              | auto_increment              |
| fee_amount  | double(20,3) | NO   |     | 0.000             |                             |
| sem_reg     | double(20,3) | NO   |     | 0.000             |                             |
| mess_dues   | double(20,3) | NO   |     | 0.000             |                             |
| summer_reg  | double(20,3) | NO   |     | 0.000             |                             |
| library_fine | double(20,3) | NO   |     | 0.000             |                             |
| misc        | double(20,3) | NO   |     | 0.000             |                             |
| time        | timestamp   | NO   |     | CURRENT_TIMESTAMP | on update CURRENT_TIMESTAMP |
+-------------+-------------+------+-----+-------------------+-----------------------------+
```

Fig 4.3.1b: Table fee_detail

- . Fee_transaction

```
mysql> describe fee_transaction;
+------------+-------------+------+-----+-------------------+-----------------------------+
| Field      | Type        | Null | Key | Default           | Extra                       |
+------------+-------------+------+-----+-------------------+-----------------------------+
| ref_no     | int(11)     | NO   | PRI | NULL              | auto_increment              |
| rollno     | varchar(50) | NO   |     | NULL              |                             |
| fee_amount | double(20,3) | NO   |     | 0.000             |                             |
| time       | timestamp   | NO   |     | CURRENT_TIMESTAMP | on update CURRENT_TIMESTAMP |
| status     | tinyint(1)  | NO   |     | 0                 |                             |
+------------+-------------+------+-----+-------------------+-----------------------------+
```

Fig 4.3.1c: Table Fee_transaction

4.3.2 Tables on Bank website

- User

```
mysql> describe user;
+-----------+--------------+------+-----+-----------+-------+
| Field     | Type         | Null | Key | Default   | Extra |
+-----------+--------------+------+-----+-----------+-------+
| acc_no    | int(11)      | NO   | PRI | NULL      |       |
| name      | varchar(50)  | NO   |     | NULL      |       |
| password  | varchar(50)  | NO   |     | NULL      |       |
| balance   | double(20,3) | NO   |     | 10000.000 |       |
| lastLogon | varchar(30)  | YES  |     | NULL      |       |
| username  | varchar(30)  | YES  |     | NULL      |       |
+-----------+--------------+------+-----+-----------+-------+
```

Fig 4.3.2a: Table User

- Transaction_bank

```
mysql> describe transaction_bank;
+------------+-------------+------+-----+-------------------+-----------------------------+
| Field      | Type        | Null | Key | Default           | Extra                       |
+------------+-------------+------+-----+-------------------+-----------------------------+
| tran_id    | int(11)     | NO   | PRI | NULL              | auto_increment              |
| ref_no     | int(11)     | NO   |     | NULL              |                             |
| acc_no     | int(11)     | NO   |     | NULL              |                             |
| fee_amount | double(20,3) | NO   |     | NULL              |                             |
| time       | timestamp   | NO   |     | CURRENT_TIMESTAMP | on update CURRENT_TIMESTAMP |
| status     | tinyint(1)  | NO   |     | 0                 |                             |
+------------+-------------+------+-----+-------------------+-----------------------------+
```

Fig 4.3.2b: Table transaction_bank

4.4 Web Pages Created

- Login Page

  User enters username and password which is encrypted using 64 bit DES/ 128 bit AES. Then the cipher text is posted to server for authentication

- Menu page

  After successful authentication a menu is displayed. User can select options like display user data, view previously successful transactions, change password and log out.

- Display User Data

  Displays user information such as name, roll, semester etc.

- Login Error Page

  Displayed when user authentication fails as a result of incorrect username or password.

- Fee Select Page

  User selects the fees which he wants to pay. The fees is then encrypted using 64 bit DES/128-bit AES. Then the cipher text is posted to server.

- Confirm fees

  It shows the user the fees he has selected and asks for confirmation and then redirects to the bank website

- Bank login

  It asks the user for username and password. It also has a hidden field which stores the encrypted total amount and reference number.

- Fee Payment

  It shows the account balance of the user and total fee amount and asks user for confirmation.

- Payment status

  If the account balance of the user is more than the total amount then this page displays transaction successful else transaction failure. It then redirects to NIT online Fee Payment website with status as successful or failure depending upon above condition.

- Fee Status

  It displays whether fee transaction as successful or failure depending upon the status it receives from bank website.

- Set password

  It asks the user to enter his old password and the new password he wishes to set. The new password must be atleast 6 characters long and has to contain at least one capital letter, one small letter and one numeric value.

4.5 Servlets

A Servlet is a Java class in Java EE that conforms to the Java Servlet API, a protocol by which a Java class may respond to HTTP requests.

The following servlets were coded to handle data on server side.

- LoginValidate

  Decrypts the user entered username and password using DES and shared key and then checks against that stored in database. It refers to studentinfo table in MySQL database and cross checks the decrypted username and password with those stored in the table. If user is authenticated then it displays menu page and creates a session else login failure page is displayed.

- Change Password

  Decrypts the ciphertext to get the old password and the new password of the user. It then updates password field of the studentinfo table and displays password change successful page if old password matches that stored in database. Otherwise it shows that password change has failed and redirects to menu page.

- Previous fee

  It shows all the previous fees the user has paid till date in a tabular format. It also shows the details such as fee type and date of payment.

- Payment Summary

  Decrypts the fees entered using the shared key and displays the fees selected by the user.

- Redirect To Bank

  After the user has confirmed to pay fees, this servlet creates a new record in fee_detail table and stores the amount of different fee type amount as entered by user. It also creates a record in fee_transaction table generating a unique reference number which

is forwarded to bank website along with the total amount. Before forwarding it encrypts them. Also the status attribute of fee_transaction table is set to zero to indicate that fees has not been paid yet.

- Validate bank user

  It validates the user by first decrypting the username and password and matching them against those in user table. If validation is successful then it decrypts the total fee amount and reference number sent by NIT fee payment website. It then displays them along with account balance which it retrieves from user table.

- Process fees

  After receiving confirmation to pay the selected fees, this servlet checks whether the account balance is more than fee amount or not. If it's more then it debits the fee amount from account balance and updates the user table to reflect the current balance and displays a payment status page showing transaction successful. If it's not then it shows payment status page with transaction failure.

- Payment status

  On receiving payment status from bank website, it updates fee_transaction table with the status. If payment is successful it updates the status attribute having the reference no as given by the bank as one. Otherwise it does nit update the fee_transaction table.

4.6 Session Tracking

HTTP is a stateless protocol: it provides no way for a server to recognize that sequences of requests are all from the same client. Privacy advocates may consider this a feature, but it

causes problems because many web applications aren't stateless. We implemented session tracking so as to recognize if the client requesting login protected pages is the one who is already logged in or if it's coming from a new client.

5. Results and Discussion

User enters username and password in Login page. Then JavaScript code encrypts the entered username and password using 128 bit AES or 64 bit DES and then posts the encrypted text to server. Server decrypts the cipher text and checks if the user is authentic by cross checking the deciphered text against those stored in database. If user is verified a session is created and a menu is displayed else login failed page is displayed. Verified user can either proceed to fee select page where he checks the fees and the amount he wants to pay or change his account password or view previous fees. In fee select page, JavaScript code encrypts the fee type and amount using 128 bit AES or 64 bit DES. It then Posts the cipher text to server which decrypts the cipher text and a page is displayed showing the fees he has selected to pay. User is redirected to bank website. Logout button is there in all webpages which invalidates the sessions. In bank login page, JavaScript code again encrypts name and password. The cipher text is then posted to server for authentication. If authentication is successful, a page is displayed which shows his current balance along with the fees to be paid. If user confirms then server checks whether his account balance is greater than the fees to be paid. If so, then a page is displayed confirming fee payment and the fee amount is debited from his balance. If his account balance is less then a transaction failure page is displayed. On successful transaction bank redirects to NIT Rourkela Online fee payment website with status as successful. In case of a transaction failure the bank website redirects with status as failure

5.1 Performance Analysis

We compared the program execution time of 64-bit DES and 128 bit AES on client as well as server side. On client side 64-bit DES was faster than 128 bit AES. As the number of characters increased so did the execution time but a steep jump was noticed when the character length was multiple of 128 bit (16 characters). It is also clearly visible form the

graph that execution time was much less in Google Chrome (almost half). The execution time in Microsoft Internet Explorer and Mozilla Firefox was almost same.

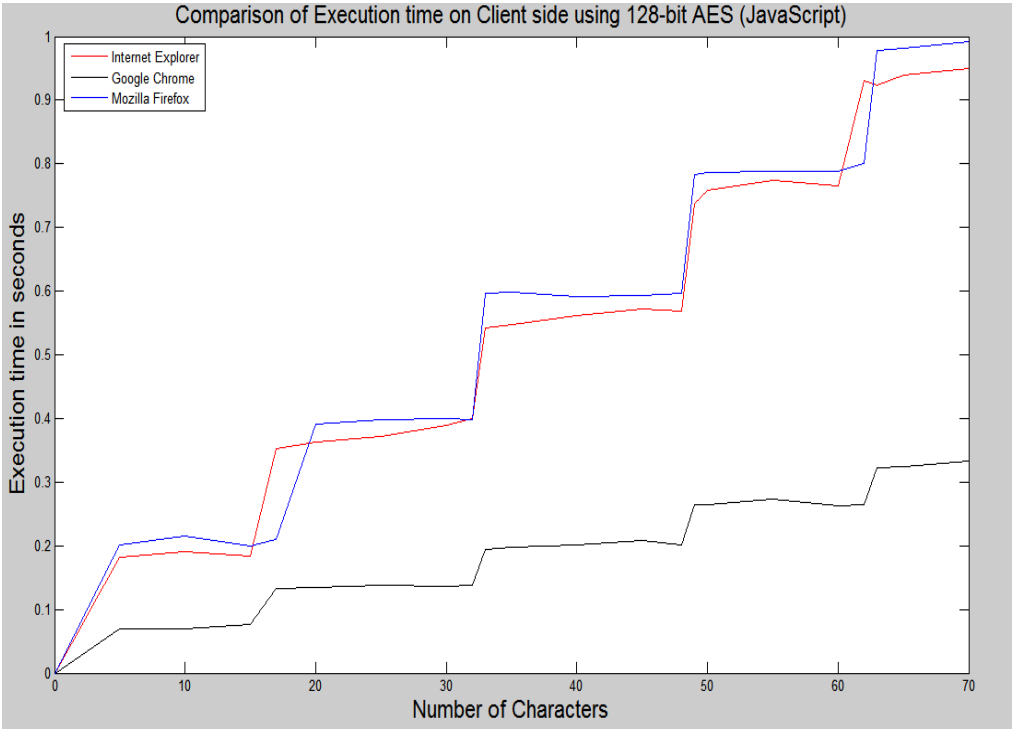

Fig 5.1a: Execution time of AES on client side

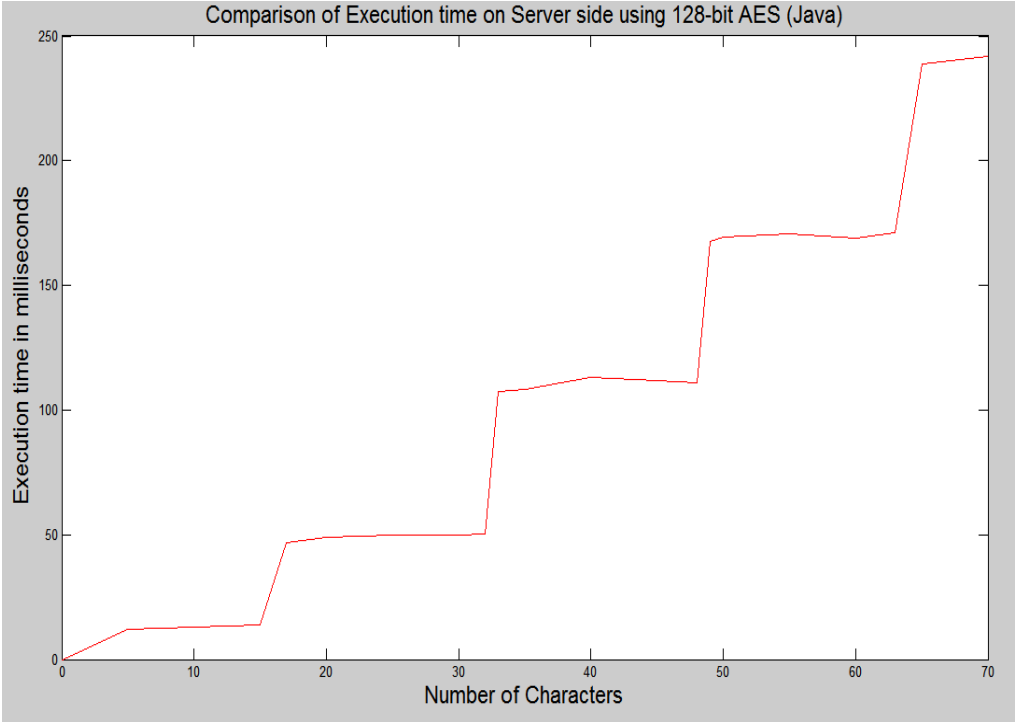

Fig 5.1b: Execution time of AES on server side

In case of 64-bit DES a steep jump is noticed at steps of 8 characters (64-bit) as DES processes encrypts blocks of size 64-bit. On client side, execution was fastest on Google Chrome. Also DES took less time to encrypt data than 128-bit AES.

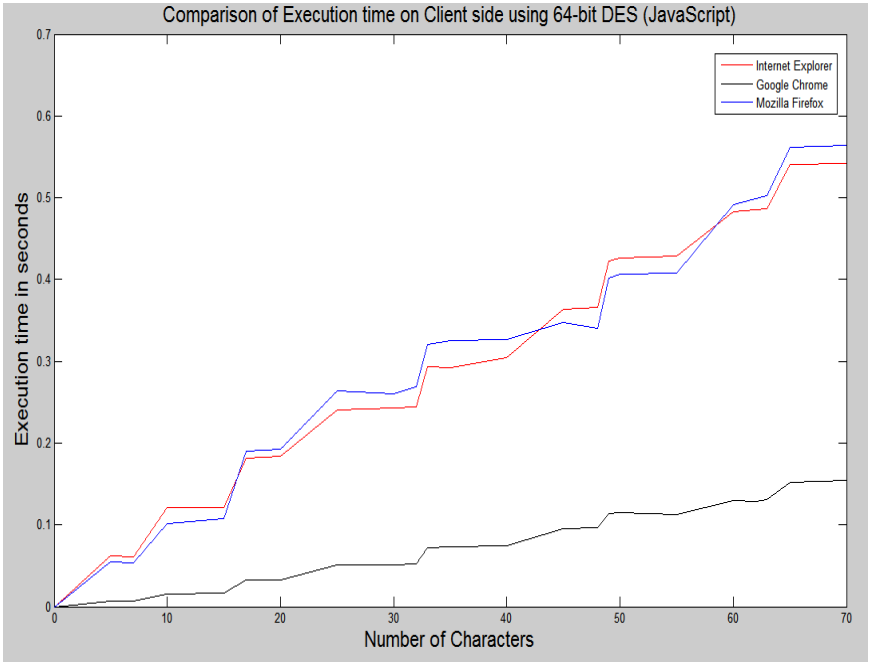

Fig 5.1c: Execution time of DES on client side

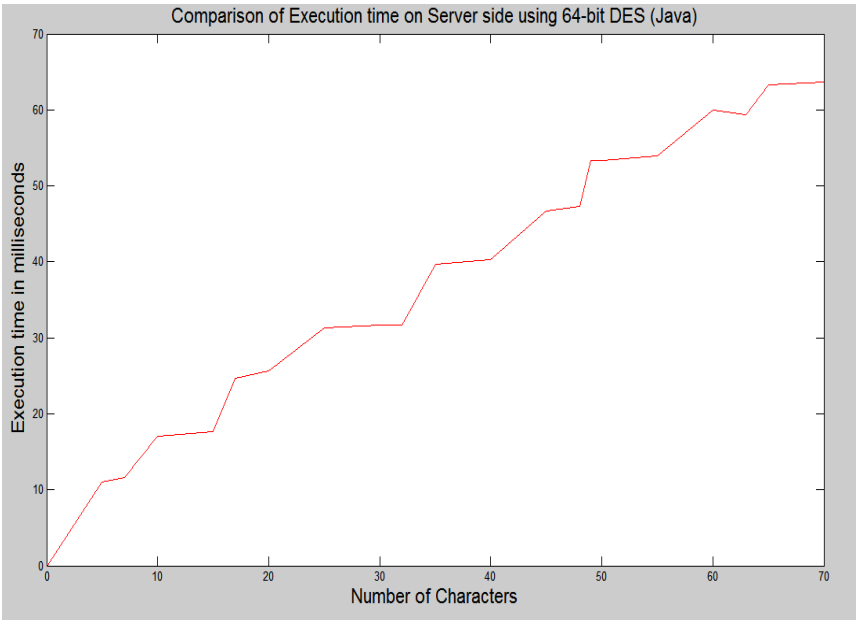

Fig 5.1d: Execution time of DES on Server side

All the results were obtained on a system with the following configuration:-

Operating system: Microsoft Windows 7 (32-bit)

CPU: Intel Mobile Core 2 Duo T5800 2.00GHz

RAM: 3.0GB Dual-Channel DDR2

Graphics: 512MB GeForce 9600M GT

6. Conclusion

6.1 Achievements and Limitations of Work

The website we have developed takes care of the flaws of the current online fee payment system. Since we are encrypting the data, it is very difficult for the attacker to decode it. Also from the graphs we can see that DES is much faster when it comes to encrypting data since it uses shorter keys and data blocks. But AES is much more secure as it uses longer keys and is currently the de facto standard of encryption. In our fee payment system, authentication is done using login page which is lacking in present fee system. We are using post method to send data from client to server as a result of which no data is visible on address bar. But since we are using symmetric key encryption, the key is shared using unsecure channel, so attacker can get access to the key if he detects network packets.

6.2 Future Work

In this project we have implemented symmetric-key cryptographic techniques to encrypt and decrypt data. So the next logical step would be to implement public key cryptographic techniques like RSA to encrypt data on client side and decrypt it on server side. Since time to encrypt using public key cryptography is much higher than those of symmetric key cryptography, it would be better if we encrypt the secret key of AES, DES using RSA and then encrypt the data using the secret key

7. References

[1] FIPS, “Data encryption standard,” Federal Information Processing Standards, 1977, National Bureau of Standard

[2] FIPS, “Advanced encryption standard,” Federal Information Processing Standards, Nov 2001, publication 197

[3] Behrouz A. Forouzan, “Cryptography & Network Security”, Special Indian Edition, Tata McGraw Hill

[4] Joel Murach, Andrea Steelman , “Murach’s Java Servlets and JSP”, 2nd edition

[5] William Stallings ,”Cryptography And Network Security”,4th Edition, Prentice Hall