Defacement of Colluding Attack Using Blowfish Algorithm

[1]Palak Jain, [2]Nikhil Kumar Singh, [3]Dr. Deepak Singh Tomar
[1]PG Scholar, Dept. of CSE, [2]PhD Scholar, Dept. of CSE, [3]Professor, Dept. of CSE
[1, 2, 3] Maulana Azad National Institute of Technology, Bhopal, Madhya Pradesh, India
[1]palak.152112210@manit.ac.in, [2]nikhil.133112004@manit.ac.in, [3]deepaktomar@manit.ac.in

Abstract - **In web environment, browser extension extends its functionality by retrieving, presenting and traversing the information through web browser. Browser extensions run with 'high' privileges which consequences, vulnerable web browser extensions to steal user's credentials and trap users into leaking sensitive information to unauthorized parties. One of the attack known as Colluding browser extension attack causes privacy leakage of share data in web browser through extensions. This paper, proposed Defacement of colluding Attack (DCA) mechanism to secure user credentials and confidential information over web browser extension. DCA mechanism encapsulate padding with blowfish algorithm to encrypt sensitive information before sharing it over common memory location. Finally the comparison evaluation of proposed mechanism is carried out with twofish, threefish, 3DES and DES on standard parameters such as encryption time, decryption time, key-length, throughput, attacks and level of security.**

Keywords - Browser, Browser Extension, Colluding Attack, Blowfish, Twofish, Threefish, 3DES, DES.

I. INTRODUCTION

Web Browser is often a user's window to the world, providing them an interface to perform a wide range of activity including social networking, shopping, personal finance management and professional business. Browser extensions become an integral part of Web browsers like Mozilla Firefox, Google Chrome, etc., to provide various features and functionalities [2], [6]. Nowadays extensions are the main presentation point for all the web contents which add more features on the top of the standard functionalities of a browser. Extensions are authored by using web technologies such as JavaScript, HTML and CSS. Browser extension changes the user interface of the web browser without directly affecting the viewable content of a webpage. It is used for improving security, browser's user interface, blocking advertisements and many other features to make browsing the internet more pleasant and easier [7]-[18].

Browser extensions have an access to every activity which is performed by end users, and can do things like injecting ads into web pages or make background HTTP requests to third-party servers. Webpages are constrained by the security model of the web browser but extensions are not. As a result, a malicious browser extension may take action against the interest of the user that installed it. Such browser extensions are a type of malware. SQL injections and Cross-Site Scripting (XSS) attacks are launched by an attacker on browser extension to steal user's personal information [3], [4].

Modern web browsers support an architecture which allows third party extensions to enhance the core functionality of the browser. For example, Firefox provides millions of free extensions to customize and enhance the look and feel of the browser [19]-[22]. Firefox executes the code of an extension with full privileges including access to all browser components, OS resources such as file system and network services, browser DOM (Document Object Model), and all web page elements. Therefore, malicious and benign vulnerable extensions are serious security threats. Researchers have shown that a malicious extension could spy on users and install malware

The communication between browser and their extensions is carried out through message passing techniques by using interfaces: i.e., APIs that can send information to the local and global environment. In Colluding Browser Extension attack [15], [24], one vulnerable extension can steal information from another extension through message passing technique. The communication between extensions allows two extensions to collude with each other, and share objects that are allocated in the same address space. So there is an Inter-Extension Collusion (IEC) which conclude the object sharing and communication channels in the browser. So for restricting the object communication or sharing of information between extensions, this paper proposed the Defacement of colluding attack (DCA) by using Blowfish Algorithm. In DCA algorithm, padding of extra bits is applied in message to make it feasible for blowfish algorithm. By using proposed algorithm, objects can be placed in a

memory in an encrypted form and if one extension uses the object of another extension, then encrypted object will be communicated from one extension to another. So that encrypted object can't be recognized by vulnerable extension or an attacker.

The paper is organized as follows. In section 2, discussion of Blowfish algorithm is done. In section 3, survey of related work is summarized. In section 4, proposed work is described. In section 5, Experimental setup and results are shown. In section 6, conclusion of paper is carried out.

Browser extension needs an encryption which would be light-weighted, highly secure and public domain. So blowfish encryption algorithm is used to fulfill all the parameters.

II. BLOWFISH ALGORITHM

Blowfish is a 64-bit block cipher which uses symmetric key encryption algorithm of variable key-length ranging from 32-bits to 448-bits for providing security and protection of data [25],[26].

Blowfish algorithm is based on 16-iteration **Feistel Network** for encryption. It is suitable for applications where key remains same, like an automatic file encryptor or a communication link. This algorithm provides better encryption and decryption mechanism for user's data.

Feistel Network: Blowfish is a 16-round Feistel Cipher in which each and every round is made up of a key and data dependent substitution and a key dependent permutation. Feistel network is a general method of transforming any function (usually called an F-function) into a permutation [27], [30].

In Blowfish algorithm, F-function splits 32-bit input data into four 8-bit quarters and uses that quarters as an input to the S-boxes i.e. S-box 1, S-box 2, S-box 3 and S-box 4 respectively as shown in Figure 1. The output of first 2 boxes i.e. 'p' and 'q' is added and subsequently modulo 232 is taken which produces output i.e. X. Then X is XOR-ed with S-box 3 output i.e. 'r' and produces another output i.e. Y. Then Y is added with S-box 4 output i.e. 's' and subsequently modulo 232 is taken which produces final output of 32-bits i.e. Z as shown in equation (1)

$$F(Z) = ((S1,p + S2,q \bmod 232) \text{ XOR } S3,r) + S4,s \bmod 232 \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (1)$$

Where p, q, r & s are output of S-boxes

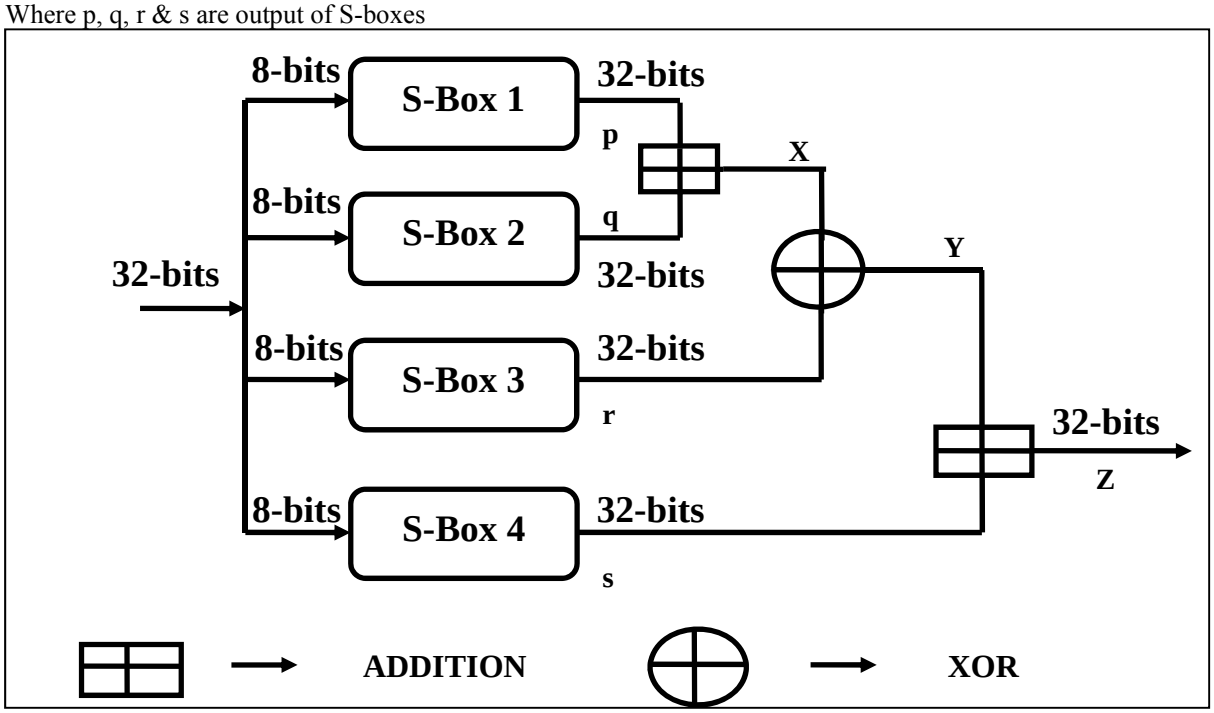

Fig. 1- Feistel Network [27]

III. RELATED WORK

In recent years, a number of research efforts have been made for securing browsers and browser extensions from malicious and vulnerable activities. In this section, there is a review on related work with a particular focus on

credentials stealing attack, storage mechanism of browser, protection of browser extensions and performance related to encryption algorithm.

A. Credentials stealing attack

In this attack, attacker steals the user credentials or important information through vulnerable extensions or through some type of infected files or softwares which is discussed below.

Anil Saini et.al. [15], [24], extend the concept of colluding extension and present the concept of attacks through collusion among browser extensions in Firefox. The Author also provided a proof-of-concept in explaining how multiple extensions can collude with each other for negotiating the browser for data leakage. Finally, they have discussed some possible mitigation techniques to address the proposed colluding attack.

Sampsa Rauti et.al. [23], explains that the problem is raised by the powerful browser extensions and viable attack surface of internet applications. The Browser extension is not only the way to realize man-in-the-browser attack. Man-in-the-Browser is a Trojan horse that infects a web browser and has the ability to tamper the contents of web pages and transactions. This attack is a serious threat to online services. Techniques like Modifying payload, Modifying DOM tree, Modifying Ajax transmission mechanism, Modifying Ajax application functionality have flaws as well because these are implemented on the target site in javascript which can be overwritten by the attacker.

B. Storage Mechanism

In this mechanism, web storage area is discussed for different browsers which store different artifacts such as cookies, history, etc.

Abner Mendoza et.al. [1], presented a brief overview of the evolution of persistent storage mechanisms on websites and describe the new web storage features wrap with the new HTML5 specifications. The main contribution of this paper is to identify the means by which different browsers implement web storage, and to show that further information can be obtained from web storage artifacts that may not be present in other browser artifacts, such as Cookies and History. They designed and implemented a tool, BrowStEx, through which one can analyze web storage artifacts on Windows platform. It parses both SQLite files and XML files in web storage used by the five major web browsers.

C. Protection of Browser Extensions

In this mechanism, the basic focus is on the protection of extensions through different techniques and different tools is used to track the flow of objects from source to sink.

Anton et.al. [5], presents a runtime protection mechanism which is based on code randomization technique and apply static analysis technique to protect browser extensions from javascript attacks. The protection is applied during runtime by separating malicious code from the randomization extension code. The protection mechanism is evaluated on the set of vulnerable and non-vulnerable firefox extensions. Their results indicated that the approach would be a viable extension. Their approach is able to reduce false positives and attain maximum compatibility with existing extensions.

SABRE [11] tracks the flow of JavaScript objects from sensitive sources to sinks inside the Mozilla Firefox browser by employing a dynamic taint analysis technique. White listing is used to separate benign extension flows from malicious ones. However, the whitelist approach essentially delegates the responsibility of deciding the maliciousness of an extension to a user. Similarly, a dynamic taint analysis based approach detects vulnerable extensions. This approach attempts to prevent unprivileged data from being compiled into privileged bytecode. It also identifies and prevents privileged caller functions from accidentally calling unprivileged code.

D. Performance related to Encryption Algorithm

In this section, comparison of different encryption algorithms is analyzed on the basis of different parameters such as block size, key-length, number of rounds, execution time, etc.

A.Ramesh et.al. [25], analyzed the performance of AES, DES and Blowfish encryption algorithms. Their performances were compared by varying block size, key size and number of round of the encryption input file. The performances are analyzed by computing certain performance parameters such as memory required, execution time and throughput. The result shows blowfish algorithm consumes less memory usage, execution time and produces more throughputs. Blowfish performed approximately 4 times faster than AES and 2 times faster than DES. AES showed poor performance results compared to other algorithms, since it required more power for processing.

A.E. Diaa et.al. [26], evaluated the common encryption algorithms such as DES, 3ES, AES, RC2, Blowfish, and RC6. There were some basic parameters of performance such as battery power consumption, encryption or decryption speed compared. The results showed that blowfish had better performance than other algorithms when changing packet size. 3DES still had low performance compared to DES algorithms. RC2 showed the poorest performance among all.

So on the basis of the related work, main concern is on the communication of object from one extension to another without user's permission which is also known as Colluding Browser Extension attack. So in this paper, algorithm is applied on data which restricts the attacker to read or detect the user's information or credentials.

IV. PROPOSED WORK

For restricting the communication of object from one extension to another, a mechanism is needed for data by which attacker cannot be able to identify the user's credential or personal information. So this paper proposed Defacement of Colluding Attack (DCA) algorithm which is implemented on message bits by sending pre-processed data as an input of Blowfish Algorithm. In proposed algorithm, data is in the form of bits and on that bits, logarithmic function is applied to minimize the value of data. Then those logarithmic values are compared with each other and among them the bigger value is selected for message bits and padding of extra bits is applied on smaller value. Then proposed algorithm adds both the values i.e. bigger one and smaller one with padded bits with each other and apply Blowfish encryption algorithm on it. After this process, output is generated in the form of ciphertext. So for decrypting the ciphertext, Blowfish decryption algorithm is applied and message bits are generated as an output of it containing padded bits with it. For removing the padded bits, divide the output of decryption algorithm into two equal halves and compare the bits one by one with padded symbol i.e. '₹'. When bit is equal to '₹' then discard that bit and when bit is not equal to '₹' then from that bit to the last bit it is going to be called complete message bits.

A. Defacement of Colluding Attack (DCA) Encryption Algorithm

DCA encryption algorithm is applied, to pre-process the data before going in the input of Blowfish Algorithm. This encryption algorithm is having two phases as shown in Figure 2. Phase-I is for padding and Phase-II is for encryption.

1) Padding:

In this phase, firstly take a card number which is equal to X-bits and Pin number which is equal to Y-bits as shown in Figure 2,

$$\boldsymbol{Card\ No. = X\ bits}$$

$$\boldsymbol{Pin\ No. = Y\ bits}$$

Then apply logarithmic function on X and Y bits because logarithms are a convenient way to express large numbers. So, we take new variable M and N to represent the value of X and Y after taking log. Therefore,

$$\boldsymbol{M = \log_2 X}$$

$$\boldsymbol{N = \log_2 Y}$$

Now the value of M & N is compared with each other to find the bigger value from the among two values i.e.

- **Case 1: If M is greater than equal to N (M >= N)**

If card bits (M) is greater than pin bits (N) then take the exponential of M for message bits i.e.

$$A = 2^M \text{..... (1)}$$

and add padding bits with 2^N. Here we are using '₹' symbol for padding extra bits. For calculating padding value (P), we have to calculate the value of M-N. Then we take the exponential of the difference value which will be taken as padding bits, therefore P is equal to

$$P = 2^{M-N} \text{..... (2)}$$

$$B = 2^N \text{..... (3)}$$

Now multiply equation (2) & (3) i.e.

$$P * B = 2^{M-N} * 2^N \text{..... (4)}$$

Then add equation (1) & (4) which will make complete message bit i.e. D

$$D = A + (P * B)$$

$$D = 2^M + (2^{M-N} * 2^N)$$

Now divide 'D' into two equal halves i.e. D_L (left data) and D_R (right data) and apply Blowfish encryption algorithm on it.

- **Case 2: If N is greater than M (M < N)**

If pin bits i.e. N is greater than card bits i.e. M then take the exponential of N for message bits i.e.,

$$A = 2^N \text{..... (5)}$$

and add padding bits with 2^M. Here we are using '₹' symbol for padding extra bits. For calculating padding value (P), we have to calculate the value of M-N. Then we take the exponential of the difference value which will be taken as padding bits, therefore P is equal to

$$P = 2^{N-M} \text{..... (6)}$$

$$B = 2^M \text{..... (7)}$$

Now multiply equation (6) & (7) i.e.

$$P * B = 2^{N-M} * 2^M \text{..... (8)}$$

Then add equation (5) & (8) which will make complete message bit i.e. D

$$D = A + (P * B)$$

$$D = 2^N + (2^{N-M} * 2^M)$$

Now divide 'D' into two equal halves i.e. D_L (left data) and D_R (right data) and apply Blowfish encryption algorithm on it.

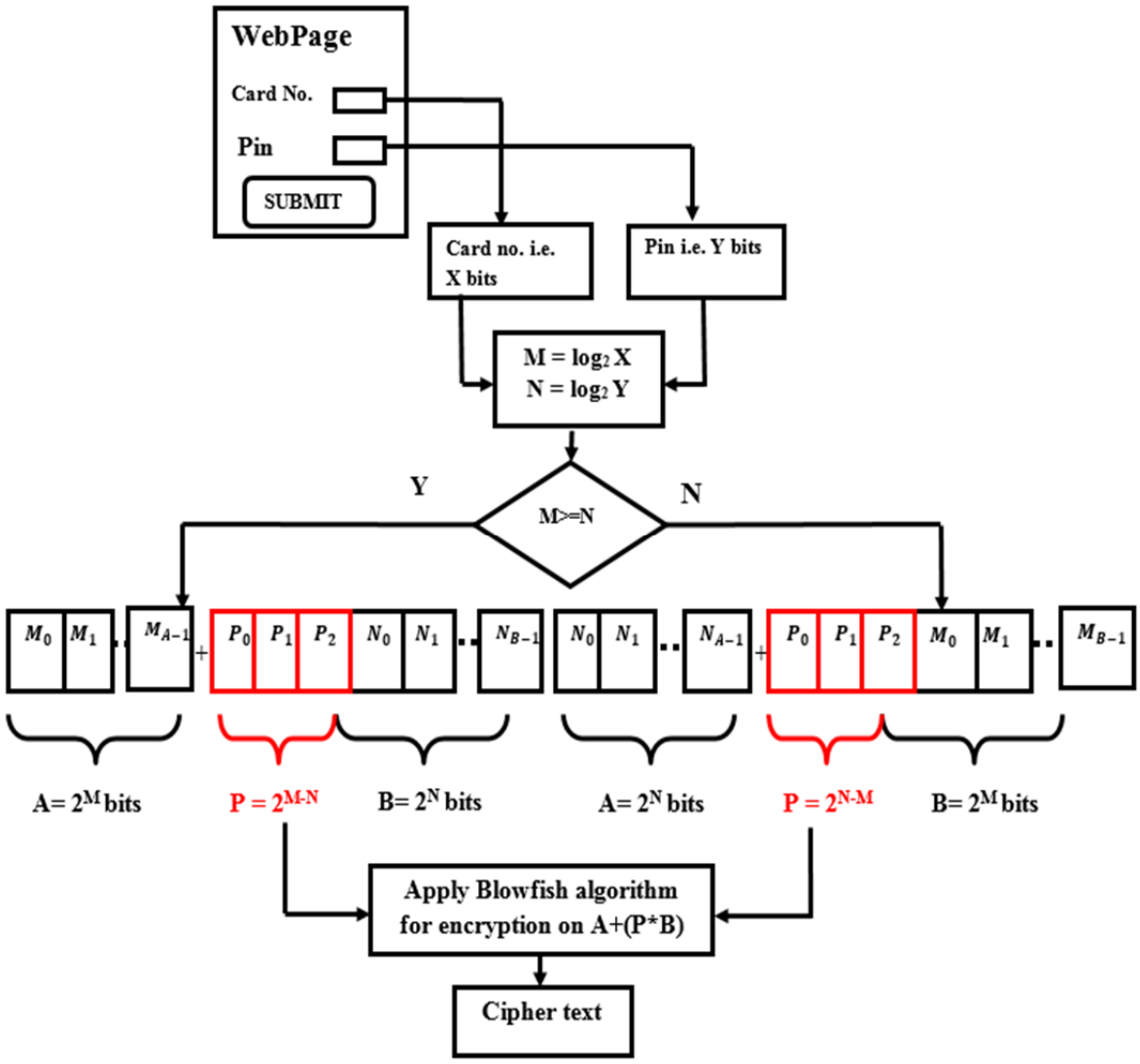

Fig. 2 – Defacement of Colluding Attack (DCA) Encryption Algorithm

2) Encryption:

Blowfish is a 64-bit block cipher encryption algorithm which can be used as a replacement of DES algorithm. It uses a variable key-length ranging from 32 bits to 448 bits having 16 rounds with input of 64-bit data [27]. This data is further divided into two equal halves as shown in Figure 3, and then apply following algorithm steps on it.

➢ Algorithm

Input is of 64-bit data element, D.
Divide D into two equal halves i.e. 32-bits each: D_L & D_R.
Then, for i = 1 to 16:
$D_L = D_L$ XOR K_i
$D_R = F(D_L)$ XOR D_R
Swap D_L and D_R
After the sixteenth round, swap D_L and D_R again to undo the last swap.
Then, $D_R = D_R$ XOR K_{17} and $D_L = D_L$ XOR K_{18}.
Finally, recombine D_L and D_R to get the ciphertext.

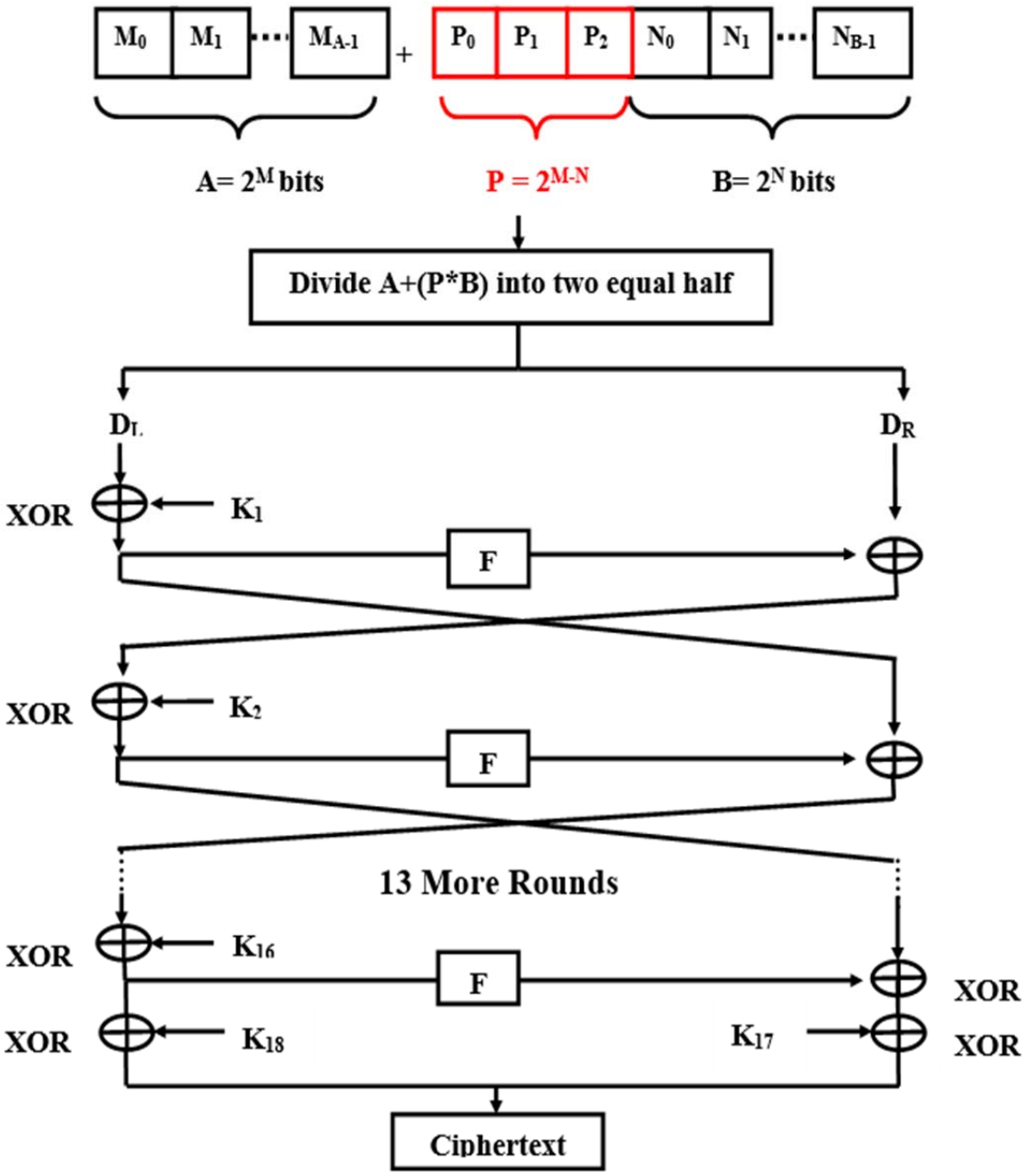

Fig. 3 – Blowfish Encryption Algorithm

- Subkeys

Blowfish uses a large number of subkeys that can be precomputed before any data encryption or decryption. Blowfish consists of an array also called as P-array which comprises of 18 sub-keys [28]. This prevents attackers from figuring out how the sub-keys were generated, and then gaining access to all the other known keys. Blowfish is solid against attacks because of the complexity of the subkey generation process. Generation of subkeys took longer time but in case of security, it is time well spent. For each key, the encryption routine runs for 522 times [29].

i. The P-array consists of 18 subkeys which is of 32-bit:

$K_1, K_2, \ldots, K_{18}$.

ii. There are four 32-bit S-boxes with 256 entries each:

S1,0, S1,1,..., S1,255;
S2,0, S2,1,…, S2,255;
S3,0, S3,1,..., S3,255;
S4,0, S4,1,…, S4,255.

- Generating subkeys

Subkeys which is used in P-array can be generated by using following steps [28]:

i. Firstly the P-array is initialized followed by four S-boxes with a fixed string that contains hexadecimal digits of pi.
ii. XOR P_1 with the key's first 32-bits, XOR P_2 with its second 32-bits, and so on upto P_{14}. This process or cycle repeated until the entire P-array has been XOR-ed with key bits.
iii. Encrypt all-zero string with the blowfish algorithm, by using the subkeys described in steps (i) and (ii).
iv. Replace P_1 and P_2 with the output of step (iii).
v. Encrypt the output of step (iii) by using the blowfish algorithm with the modified subkeys.
vi. Replace P_3 and P_4 with the output of step (v).
vii. Continue the process, replace all entries of the P-array, followed by all four S-boxes, with the output of the continuously changing the blowfish algorithm.

B. Defacement of Colluding Attack (DCA) Decryption Algorithm

DCA decryption algorithm is used for de-padding the message bits after decrypting the ciphertext by using Blowfish Algorithm. This decryption algorithm is having two phases as shown in Figure 5. Phase-I is for decryption and Phase-II is for de-padding.

1) Decryption

For Blowfish cipher, encryption algorithm is so well intended, that the decryption algorithm is same as the encryption algorithm step by step in the same order [27], only the sub-keys are applied in the reverse order as shown in Figure 4.

- *Algorithm*

Input is of 64-bit ciphertext, C.
Divide C into two equal halves i.e. 32-bits each: C_L & C_R.
Then, for i = 1 to 16:
$C_L = C_L$ XOR K_{16}
$C_R = F(C_L)$ XOR C_R
Swap C_L and C_R
After the sixteenth round, swap C_L and C_R again to undo the last swap.
Then, $C_R = C_R$ XOR K_2 and $C_L = C_L$ XOR K_1.
Finally, recombine C_L and C_R to get the data element.

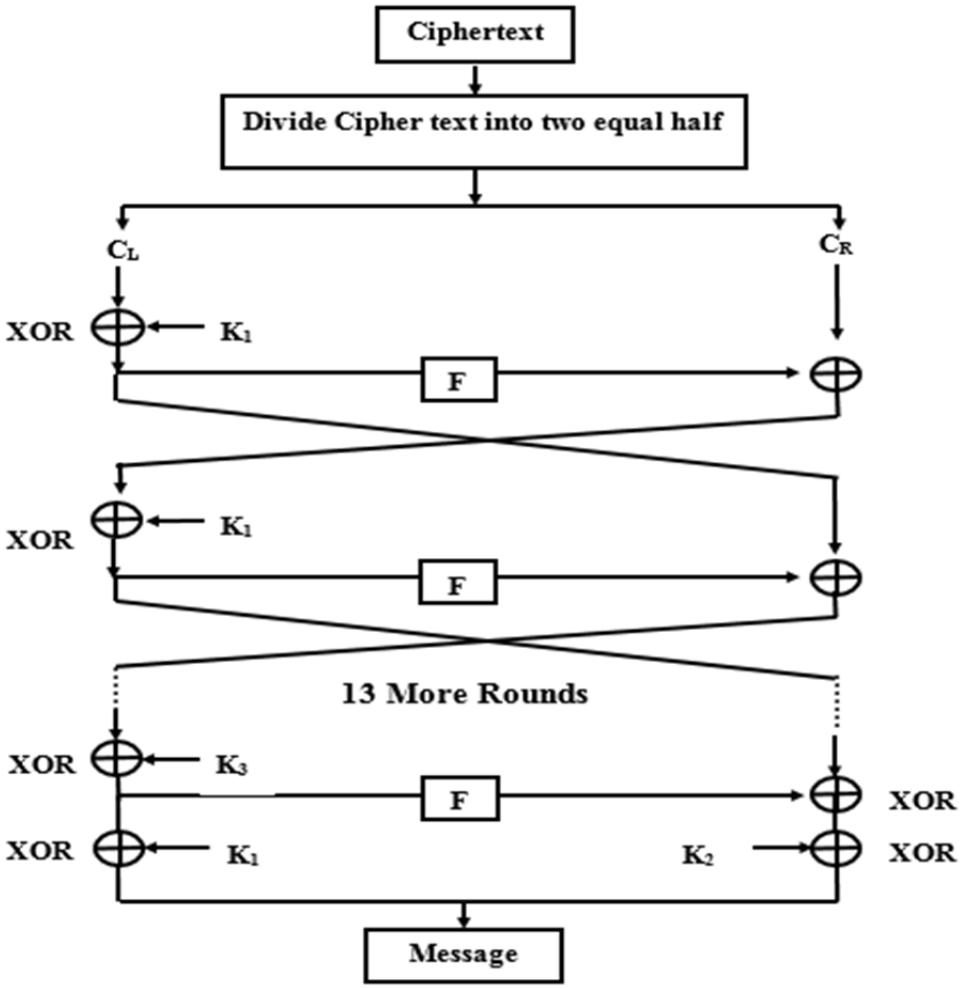

Fig. 4 – Blowfish Decryption Algorithm [27]

2) De-Padding

After applying algorithm, the output is obtained in the form of message or data element i.e. Card number & Pin number in combined form. So divide that message into two equal halves.

Suppose message is having a range from 0 to R which is divide into two halves i.e. first one is from 0 to Q-1 & second one is from Q to R as shown in Figure 5.

Now compare both halves of message bits one by one with '₹' symbol because at the time of padding we pad the extra bits with'₹' symbol.

So, compare each message bit and separate padding bits from the message as shown in Figure 5, and extract the Card no. and Pin number from it. So comparison of bits is done by using basic algorithm i.e.

```
i=0;
While (Mi == ₹)
{
 Separate the bit from whole message bits because this is padding bit
 i++;
  }
 From this bit to the last bit there is a message bits without padding bits
```

So by using this, there is an extraction of valuable bits i.e. Card number & Pin number from the complete message bits.

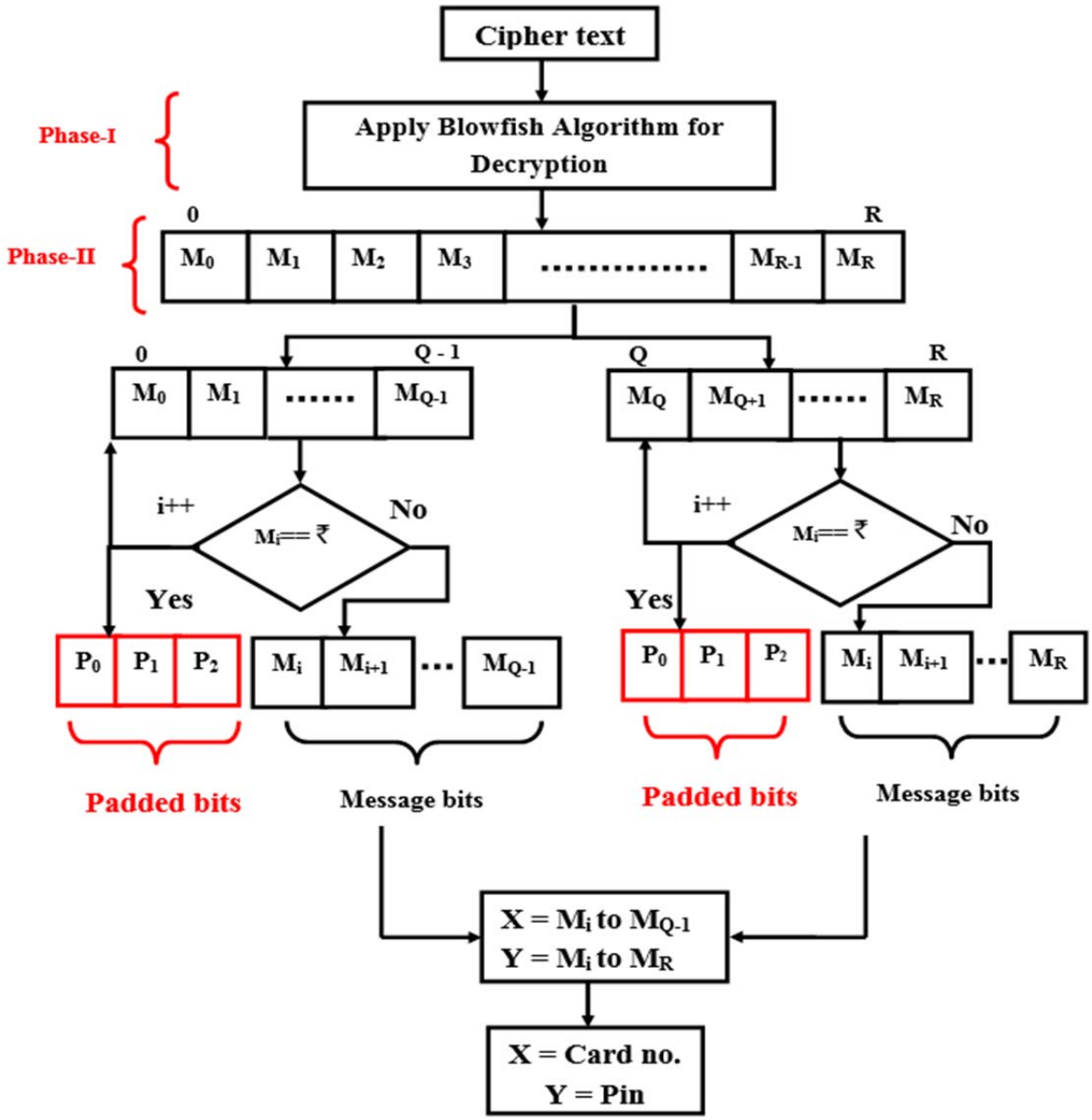

Fig. 5 – Defacement of Colluding Attack (DCA) Decryption Algorithm

V. EXPERIMENTAL SETUP AND RESULT ANALYSIS

For experiment purpose, a computer present in Network Security Lab of MANIT, Bhopal is used with Intel Core(TM) i7-3770 CPU @ 3.40 GHZ CPU with 2GB RAM.

Implementation of DCA algorithm is done by using Blowfish algorithm in python language using Linux Operating System. In the experiment, encryption and decryption is carried out on 1000 different datasets i.e. card numbers and pin numbers. For making the bits of pin number equal to card number, padding of extra bits is required and which is done by using '₹' symbol and then encryption and decryption time is calculated.

Here 10 datasets of encryption time is shown in Table I, which is ranging from $16.80 * 10^{-5}$ to $23.37 * 10^{-5}$ ms

Table I. Calculation of Encryption Time

ID Number	Card Number	Actual Pin Number	Pin Number with Padded bits	Ciphertext	Encryption Time (ms)
ID 1	5574 5372 8573 2205	0234	₹₹₹₹ ₹₹₹₹ ₹₹₹₹ 0234	xa0\xf4l\x99\xa0\xb2LA\xa3ZXJ\x058\xe4\ xae\xd6\xdd\x168\xc7\x11	16.80 * 10^{-5}
ID 2	4537 9472 6503 4087	6482	₹₹₹₹ ₹₹₹₹ ₹₹₹₹ 6482	x91\xc6\xb06\xa1\xeb#\xf8\x01\x03\x81\xe 6\xb3\xb6\x13\P6r	23.04 * 10^{-5}
ID 3	4209 7845 3269 0743	9467	₹₹₹₹ ₹₹₹₹ ₹₹₹₹ 9467	x08\x13\xc4\x0492A\xc2\xf8\x1b\x01\xe5\x cdO	21.71 * 10^{-5}
ID 4	4513 7569 3756 2064	1576	₹₹₹₹ ₹₹₹₹ ₹₹₹₹ 1576	x96\xde`X\\\xbd\xfe\x03\xf5\x94\xd9\xb6\x dcS\x99\xa1\xa8_^\x16	21.15 * 10^{-5}
ID 5	5238 3850 7385 2856	9588	₹₹₹₹ ₹₹₹₹ ₹₹₹₹ 9588	x01%\xb9\xed\x14\xdd\xbc`9\xe0\x02\x8c \xea\x9b\xc98\xa6	21.11 * 10^{-5}
ID 6	5028 7359 6502 0389	4668	₹₹₹₹ ₹₹₹₹ ₹₹₹₹ 4668	x05,\x84\xb1f\xb4w\x1b,\xab\xe6\x1d\xbb[\ xe3\x02<]	19.68 * 10^{-5}
ID 7	4009 9061 5286 0017	9267	₹₹₹₹ ₹₹₹₹ ₹₹₹₹ 9267	np\xd2G8Is\x0f\xf8&\x7f\x01\xd6\xc44z\xb 21\xa1\xb4\xd93\x11	19.08 * 10^{-5}
ID 8	4777 8443 0947 6698	2497	₹₹₹₹ ₹₹₹₹ ₹₹₹₹ 2497	xbde\xa6\xdd>\xa3~\x0e\x1d\x15\xc0u\x91\ x82\xe9$\xec'\xfc\x80h	23.37 * 10^{-5}
ID 9	5003 7296 7794 6346	7347	₹₹₹₹ ₹₹₹₹ ₹₹₹₹ 7347	x80\x85\xcbE\x84\x18\xbe:\x9c\xe2\xea6\xe d\x85\xd74+\xdb	19.34 * 10^{-5}
ID 10	5501 3281 6591 5501	8796	₹₹₹₹ ₹₹₹₹ ₹₹₹₹ 8796	x15K\x81\xf0\xbf\xaeE\xaf\xddD\x1a\xb3\x 98\xb8{\xba	17.62 * 10^{-5}

In decryption table, calculation of decryption time is done which is shown in Table II. Here 10 datasets of decryption time is shown which is ranging from 13.07 * 10^{-5} to 17.75 * 10^{-5}

Table II. Calculation of Decryption Time

ID Number	Ciphertext	Card Number	Pin Number with Padded bits	Decryption Time (ms)
ID 1	xa0\xf4l\x99\xa0\xb2LA\xa3ZXJ\x058\xe4\xae\xd6\xdd\x168\xc7\x11	5574 5372 8573 2205	₹₹₹₹ ₹₹₹₹ ₹₹₹₹ 0234	14.26
ID 2	x91\xc6\xb06\xa1\xeb#\xf8\x01\x03\x81\xe6\xb3\xb6\x13\P6r	4537 9472 6503 4087	₹₹₹₹ ₹₹₹₹ ₹₹₹₹ 6482	16.84
ID 3	x08\x13\xc4\x0492A\xc2\xf8\x1b\x01\xe5\xcdO	4209 7845 3269 0743	₹₹₹₹ ₹₹₹₹ ₹₹₹₹ 9467	14.46
ID 4	x96\xde`X\\\xbd\xfe\x03\xf5\x94\xd9\xb6\xdcS\x99\xa1\xa8_^\x16	4513 7569 3756 2064	₹₹₹₹ ₹₹₹₹ ₹₹₹₹ 1576	17.75
ID 5	x01%\xb9\xed\x14\xdd\xbc`9\xe0\x02\x8c\xea\x9b\xc98\xa6	5238 3850 7385 2856	₹₹₹₹ ₹₹₹₹ ₹₹₹₹ 9588	16.20
ID 6	x05,\x84\xb1f\xb4w\x1b,\xab\xe6\x1d\xbb[\xe3\x02<]	5028 7359 6502 0389	₹₹₹₹ ₹₹₹₹ ₹₹₹₹ 4668	14.87
ID 7	np\xd2G8Is\x0f\xf8&\x7f\x01\xd6\xc44z\xb21\xa1\xb4\xd93\x11	4009 9061 5286 0017	₹₹₹₹ ₹₹₹₹ ₹₹₹₹ 9267	13.95
ID 8	xbde\xa6\xdd>\xa3~\x0e\x1d\x15\xc0u\x91\x82\xe9$\xec'\xfc\x80h	4777 8443 0947 6698	₹₹₹₹ ₹₹₹₹ ₹₹₹₹ 2497	16.06
ID 9	x80\x85\xcbE\x84\x18\xbe:\x9c\xe2\xea6\xed\x85\xd74+\xdb	5003 7296 7794 6346	₹₹₹₹ ₹₹₹₹ ₹₹₹₹ 7347	15.60
ID 10	x15K\x81\xf0\xbf\xaeE\xaf\xddD\x1a\xb3\x98\xb8{\xba	5501 3281 6591 5501	₹₹₹₹ ₹₹₹₹ ₹₹₹₹ 8796	13.07

According to results shown in Figure 6, DCA with Blowfish algorithm is having less encryption and decryption time as compared to other techniques. So DCA with blowfish algorithm will make data more secure as compare to others with minimum encryption and decryption time.

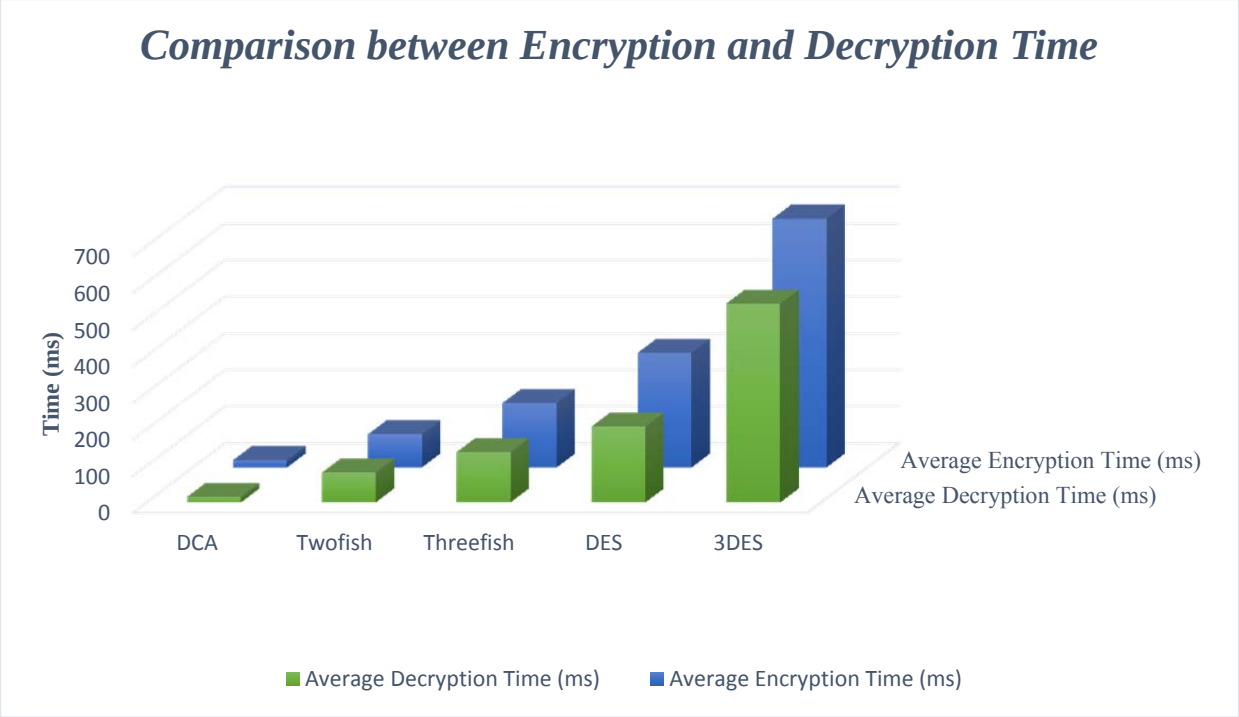

Fig. 6 – Comparison between Encryption and Decryption Time

Beside encryption and decryption time other parameters like key-length, rounds, block-size, attacks found and level of security are also important factors to compare the algorithms. In table III, comparison of algorithms is done on different parameters for different algorithms like DCA, Twofish, Threefish, 3DES and DES. For any efficient encryption algorithm it is required to have variable key-length which takes exponential time to crack the encryption. In proposed DCA algorithm, key-length is ranging from 32-448 bits whereas in twofish, threefish, 3DES and DES is having static key-length and it is very easy to crack them.

Table III –Comparison Chart

Parameters	DCA	Twofish	Threefish	DES	3DES
Key Length	Variable Key Length 32 – 448	128, 192, 256	256,512, 1024	168,112	64 (56 Usable)
Throughput	Very High	High	Average	Low	Very Low
Attacks Found	No Attack Is Found To Be Successful Against Blowfish	Differential Attack, Related Key Attack	Improved Related-Key Boomerang	Related Key Attack	Exclusive Key Search, Linear Cryptanalysis, Differential Analysis
Level Of Security	Highly Secure	Secure	Secure	Adequate Security	Adequate Security

Throughput means encryption on data executed per unit of time and for any efficient algorithm it must be high. In proposed DCA algorithm, throughput is very high as compare to twofish, threefish, 3DES and DES.

No attacks are found on DCA but attacks like differential attacks, related key attack, etc., are found on twofish, threefish, 3DES and DES.

Level of Security must be high for efficient encryption algorithm. Proposed DCA algorithm is highly secure as compare to twofish, threefish, 3DES and DES.

VI. CONCLUSION

Colluding browser extension attack is a serious threat for user credentials and confidential information over web and share confidential information between extensions. Malicious extensions deployed colluding attack to extract credential information from other extensions for malicious activity. This paper presents a blowfish algorithm based mechanism to defense colluding attack over web browser and share an object between the extensions but in an encrypted form. DCA encryption and decryption algorithm have two dependent phases one for padding/de-padding and other for encryption/decryption respectively. For performance evaluation parameters like key-length, power consumption, throughput, attacks and level of security has been taken. Here the average encryption and decryption time of DCA, twofish, threefish, 3DES and DES is compared and among them DCA Algorithm is having least encryption time and decryption time i.e. $20.29 * 10^{-5}$ ms and $15.31 * 10^{-5}$ ms respectively. In future, aim of research is to develop defense mechanism for colluding attack over android platform for securing mobile transactions.

REFERENCES

[1] Abner Mendoza, Avinash Kumar, David Midcap, Hyuk Cho, CihanVarol, "BrowStEx: A tool to aggregate browser storage artifacts for forensic analysis", Digital Investigation, Volume 14, September 2015, Pages 63-75
[2] Gaurav Aggarwal, ElieBursztein, Collin Jackson, and Dan Boneh. 2010. "An analysis of private browsing modes in modern browsers". *In Proceedings of the 19th USENIX conference on Security (USENIX Security'10)*. USENIX Association, Berkeley, CA, USA, 6-6.
[3] Hossain Shahriar, KomminstWeldemariam, Mohammad Zulkernine, ThibaudLutellier, "Effective detection of vulnerable and malicious browser extensions", Computers & Security, Volume 47, November 2014, Pages 66-84.
[4] Rezwana Karim, Mohan Dhawan, VinodGanapathy, Chung-chieh Shan, "An Analysis of the Mozilla Jetpack Extension Framework", *proceedings of the 26th European Conference on Object-Oriented Programming*, Volume 7313, June 2012, Pages 333-355.
[5] A. Barua, M. Zulkernine and K. Weldemariam, "Protecting Web Browser Extensions from JavaScript Injection Attacks," 2013 *18th International Conference on Engineering of Complex Computer Systems*, Singapore, 2013, pp. 188-197.
[6] Rui Zhao, ChuanYue, "Toward a secure and usable cloud-based password manager for web browsers", Computers & Security, Volume 46, October 2014, Pages 32-47
[7] M. Silic, J. Krolo and G. Delac, "Security vulnerabilities in modern web browser architecture," *The 33rd International Convention MIPRO*, Opatija, Croatia, 2010, pp. 1240-1245.
[8] XiuquanQiao, Guoshun Nan, YuePeng, Lei Guo, Jingwen Chen, Yunlei Sun, Junliang Chen, "NDNBrowser: An extended web browser for named data networking, Journal of Network and Computer Applications", Volume 50, April 2015, Pages 134-147
[9] AlperAksac, OrkunOzturk, ErdoganDogdu, "A novel semantic web browser for user centric information retrieval: PERSON, Expert Systems with Applications", Volume 39, Issue 15, 1 November 2012, Pages 12001-12013
[10] HengXu, Robert E. Crossler, France Bélanger, "A Value Sensitive Design Investigation of Privacy Enhancing Tools in Web Browsers", Decision Support Systems, Volume 54, Issue 1, December 2012, Pages 424-433
[11] A. Zammouri and A. A. Moussa, "SafeBrowse: A new tool for strengthening and monitoring the security configuration of web browsers," *2016 International Conference on Information Technology for Organizations Development (IT4OD)*, Fez, 2016, pp. 1-5.
[12] C. Grier, S. Tang and S. T. King, "Secure Web Browsing with the OP Web Browser," *2008 IEEE Symposium on Security and Privacy (sp 2008)*, Oakland, CA, 2008, pp. 402-416.
[13] SruthiBandhakavi, NanditTiku, Wyatt Pittman, Samuel T. King, P. Madhusudan, and Marianne Winslett. 2011. "Vetting browser extensions for security vulnerabilities with VEX. Commun". ACM 54, 9 (September 2011), 91-99.
[14] Claudio Marforio, Hubert Ritzdorf, AurélienFrancillon, and SrdjanCapkun. 2012. "Analysis of the communication between colluding applications on modern Smartphones" *in Proceedings of the 28th Annual Computer Security Applications Conference (ACSAC '12)*. ACM, New York, NY, USA, 51-60.
[15] Anil Saini, Manoj Singh Gaur, Vijay Laxmi, Mauro Conti, "Colluding browser extension attack on user privacy and its implication for web browsers", Computers & Security, Volume 63, November 2016, 14-28.
[16] A. Grosskurth and M. W. Godfrey, "A reference architecture for Web browsers," *21st IEEE International Conference on Software Maintenance (ICSM'05)*, 2005, pp. 661-664.
[17] Dormann, Will, and Jason Rafail. "Securing your web browser." CERT, 2006.
[18] Barth, A., Felt, A. P., Saxena, P., &Boodman, A. "Protecting Browsers from Extension Vulnerabilities". *In NDSS*, 2010
[19] Nicholas Carlini, Adrienne Porter Felt, and David Wagner. 2012. "An evaluation of the Google Chrome extension security architecture". *In Proceedings of the 21st USENIX conference on Security symposium (Security'12)*. USENIX Association, Berkeley, CA, USA, 7-7.
[20] Dhruwajita Devi, Dhrubajyoti Pathak, and Sukumar Nandi. "Vulnerabilities in Web Browsers", 2010
[21] A. Guha, M. Fredrikson, B. Livshits and N. Swamy, "Verified Security for Browser Extensions," *2011 IEEE Symposium on Security and Privacy*, Berkeley, CA, 2011, pp. 115-130.
[22] M. Dhawan and V. Ganapathy, "Analyzing Information Flow in JavaScript-Based Browser Extensions," *2009 Annual Computer Security Applications Conference*, Honolulu, HI, 2009, pp. 382-391.
[23] Sampsa Rauti and Ville Leppänen. 2012. "Browser extension-based man-in-the-browser attacks against Ajax applications with countermeasures". *In Proceedings of the 13th International Conference on Computer Systems and Technologies (CompSysTech '12)*, Boris Rachev and Angel Smrikarov (Eds.). ACM, New York, NY, USA, 251-258.

[24] Anil Saini, Manoj Singh Gaur, and Vijay Laxmi. 2013. "The darker side of Firefox extension". *In Proceedings of the 6th International Conference on Security of Information and Networks (SIN '13)*. ACM, New York, NY, USA, 316-320.
[25] A. Ramesh and A. Suruliandi, "Performance analysis of encryption algorithms for Information Security," *2013 International Conference on Circuits, Power and Computing Technologies (ICCPCT)*, Nagercoil, 2013, pp. 840-844.
[26] A.E.DiaaSalama, Mohamed AbdualKader.H,Mohamed Hadhoud.M, "Evaluating The Performance of Symmetric Encryption Algorithms", International Journal of Network security, PP.216-222,2010.
[27] Manju Suresh, M. Neema, "Hardware Implementation of Blowfish Algorithm for the Secure Data Transmission in Internet of Things", *Procedia Technology*, Volume 25, 2016, Pages 248-255
[28] A. Alabaichi, F. Ahmad and R. Mahmod, "Security analysis of blowfish algorithm," *2013 Second International Conference on Informatics & Applications (ICIA)*, Lodz, 2013, pp. 12-18
[29] *T. Nie, C. Song and X. Zhi,* "Performance Evaluation of DES and Blowfish Algorithms," *2010 International Conference on Biomedical Engineering and Computer Science*, Wuhan, 2010, pp. 1-4
[30] V. Poonia and N. S. Yadav, "Analysis of modified Blowfish algorithm in different cases with various parameters," *2015 International Conference on Advanced Computing and Communication Systems*, Coimbatore, 2015, pp. 1-5.

AUTHOR PROFILE

Palak Jain completed her B-Tech in Computer Science & Engineering with Honours, currently she is pursuing her M-Tech. in Computer Science & Engineering from Maulana Azad National Institute of Technology (MANIT), Bhopal, India.

Nikhil Kumar Singh completed his B. E. in (CSE), M-Tech. in Computer Science & Engineering from Maulana Azad National Institute of Technology (MANIT), Bhopal with Honours and currently pursuing PhD (CSE) from Maulana Azad National Institute of Technology, Bhopal, India. He has 4 years of teaching experience and guided 8 M-Tech Thesis and 10 B-Tech. projects. He has published more than 17 research papers in national & international journals and conferences. He is holding positions in many world renowned professional bodies. His present research interests include Data mining, Web Mining, Social Media Mining, Sentiment Analysis and digital forensics.

Prof. Deepak Singh Tomar (CSE department at NIT- Bhopal, India) completed his B. E., M-Tech. and PhD. degrees in Computer Science and Engineering. He is co-investigator of Information Security Education Awareness (ISEA) project under Govt. of India. Currently, he is chairman of cyber security center, MANIT, Bhopal. He has more than 21 years of teaching experience. He has guided 30 M-Tech and 7 PhD Thesis. Besides this he guided 70 B-Tech and 15 MCA projects. He has published more than 54 papers in national & international journals and conferences. He is holding positions in many world renowned professional bodies. His present research interests include web mining and cyber security.