Circulation in Computer Science
Vol.2, No.5, pp: (13-17), June 2017
https://doi.org/10.22632/ccs-2017-252-17

JSSecure: A Secured Encryption Strategy for Payment Gateways in E-Commerce

Ramkrishna Oruganti
Computer Engineering,
SVKM's NMIMS Mukesh Patel School of Technology Management and Engineering,
Mumbai, India.

Saurabh Shah
Computer Engineering,
SVKM's NMIMS Mukesh Patel School of Technology Management and Engineering,
Mumbai, India.

Yohan Pavri
Computer Engineering,
SVKM's NMIMS Mukesh Patel School of Technology Management and Engineering,
Mumbai, India.

Neelansh Prasad
Computer Engineering,
SVKM's NMIMS Mukesh Patel School of Technology Management and Engineering,
Mumbai, India.

Prathamesh Churi
Assistant Professor, Computer Engineering,
SVKM's NMIMS Mukesh Patel School of Technology Management and Engineering,
Mumbai, India.

ABSTRACT
JSSecure is a framework for online payment systems over e-commerce websites. Payments made online using debit/credit cards have become familiar, and the users are shifting to a higher comfort level with this method of payment. Nowadays for any online transactions, a payment gateway is used which is a service that is provided by an e-commerce or by any bank that authorizes the details of the user for the secure transaction. This paper presents a frame format of JSSecure. For any transaction, there has to be a way in which the user details needs to be protected. Cryptography is one of the methods which is used for converting the information from its standard form to encrypted form or unreadable for the attackers. Using JSSecure, each user detail is encrypted individually to provide extra security against attackers. There are umpteen number of payment gateway methods like 3D Secure, SET, and MSET Protocols. Various algorithms help user securely enter his/her card details, some of them are Jumbling Salting (JS), Data Encryption Standard (DES), Advanced Encryption Standard (AES), etc. which are used for the encrypting the details securely. All these algorithms are symmetric key. JSSecure uses double encryption strategy for more security. We will be providing a fair comparison of Data Encryption Standard (DES), Advanced Encryption Standard (AES) and Jumbling Salting (JS) algorithms. Since our major concern here is the performance of algorithms under different conditions, we will be comparing on the basis of speed, block size, and key size on the encryption time, decryption time, throughput and size of cipher text. This analysis will help in implementing the best-suited algorithm for the proposed payment gateway. It will be open source and hence it will be more cost efficient.

Keywords
JSSecure, Payment Gateway, E-Commerce, Cryptography, Jumbling Salting, Encryption

1. INTRODUCTION
"Safety first is safety always", this quote from Charles Hayes has a profound meaning in today's digital world. As technology rises security and safety of data is of paramount importance. Anyone using technology must ensure that their data is secure and safe from unauthorized use.

Payment gateway is a mediator between the bank, merchant and the user. With the rise in technology and its user base, E-payments are a common practice. The user enters his card details, CVV number and expiry date which in turn is encrypted by the payment gateway and sent to the bank. The bank decrypts the user details, matches it with its database and proceeds with the transaction. The payment gateway is a secure mediator.

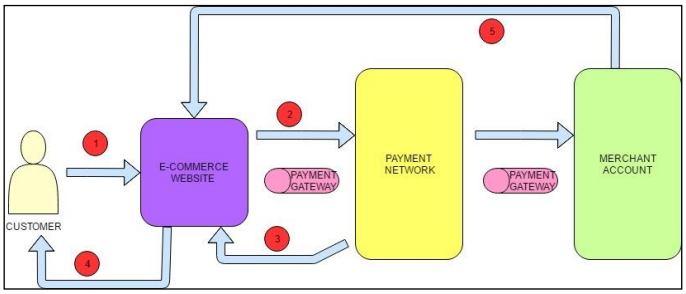

Figure 1. Working of Payment Gateway

The main idea behind securing data is encrypting it so that attackers cannot access the sensitive data for unauthorized use. Some algorithms that are used for encrypting the user details are JS (Jumbling and Salting) [2,12], DES (Data Encryption Standard) [1, 6], AES (Advanced Encryption Standard), etc. These algorithms use symmetric key encryptions [1, 5, 6] i.e. the same key is used to encrypt and decrypt the data. These algorithms require frequent update of the key used which is tedious to maintain. An overview and comparison of these algorithms is presented in this paper. We have compared the algorithms based on the following parameters: Cipher text Size, Encryption Time, Decryption Time and Throughput [2].

We have proposed a new way to ensure security of transactions by using JSSecure. JSSecure is designed in such a way that it uses double encryption for securing the user details from the attacks. It is based on the Jumbling-Salting algorithm which is a symmetric key algorithm [2]. JS is based on two steps first Jumbling and then Salting. During Jumbling process

it undergoes three sub processes i.e. addition, selection and reversing. During addition it calculates and generates a value that needs to be added to the password. In selection it randomly chooses the symbols and adds them to the string. In reversal it reverses the new string of password that is formed. In Salting process the jumbled string is taken as input and some salt is added to it. The JS Algorithm uses randomized processes which make the data even more secure by making it difficult for attackers to predict the plaintext [2].

In this paper we have proposed JSSecure, a new strategy for online transactions. We have also presented a comparison between some of the existing algorithms and the JS algorithm.

2. RELATED WORK

2.1. Existing Algorithms

Any kind of information or data that is translate into a secret code is encryption. It is one of the most effective way to achieve security. To decrypt that data one must have access to its secret key for decryption. The data that is not encrypted is Plain Text and the data which is encrypted is called Cipher Text.

2.1.1 DES

The Data Encryption Standard (DES) [1] is one of the oldest algorithms that was used for encryption. It uses symmetric key encryption strategy. For encryption and decryption it uses the same key so the sender and receiver should have that key for encrypting and decrypting.

It was introduced in the early 1970's. It is a block cipher which means that the encryption is applied to different blocks of data rather than applying to just 1 bit at a time. DES uses the block size of 64-bit. Each block is enciphered by applying mathematical function i.e. permutation and substitution permutation involves 16 rounds and though the key is 64-bit 8 keys are used for parity check so only 56 bits are effective. The decryption process is also simple in which we inverse the encryption. The steps are same as encryption but the order is reversed in which keys are applied [1, 2].

Now one of the basic attacks that is observed for any cipher text is Brute Force attack. In this kind of attack the right key is obtained by trying all the possible keys. DES uses 64–bit key [5] but out of 64, 8 are used for parity check and thus left with 56 effective bits. Hence it would take 2^56 attempts by brute force attack for finding out the correct key.

2.1.2 AES

The major problem observed with DES was its key size and how it could be cracked by repetitive attacks. Advanced Encryption Standard (AES) [1, 6] was designed to overcome this drawback. It is one of the most popular and widely used encryption strategy.

AES is an iterative algorithm. It is based on substitution, permutations and linear transformations. Each of these operations occurs in a single round. The plaintext is encrypted a number of times based on the key size. For 128-bit key it uses 10 rounds, for 192-bit 12 rounds and for 256-bit 16 rounds [1, 5, 6]. Each of the rounds uses a different key for encryption of data.

The flow chart below shows the typical processes involved in one round of the AES algorithms. Initially the plaintext is encrypted with the first round key [1, 9]. There are 4 sub processes [9] involved:

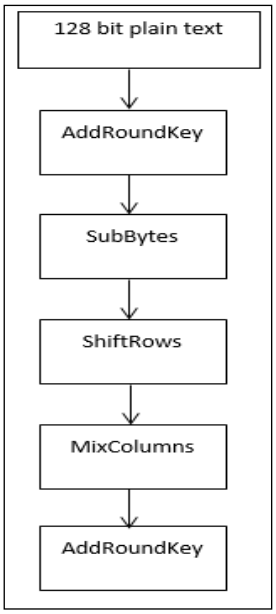

Figure 2. Block Diagram od AES Algorithm

i)SubBytes: This step performs the substitution of bytes by looking up a fixed S-Box table. The S-Box is typically is Rijndael S-Box. This provides non-linearity to the cipher. The output of this step is a 4x4 matrix.

ii)ShiftRows: Each of the rows in the matrix is shifted to the left. Each byte in the rows is shifted by some offset. The first row is unchanged. The second row is shifted one position to the left. The third row is shifted two positions and the final row is shifted three positions. The output is a matrix with same bytes but different positions with respect to each other.

iii)MixColumns: Each column is transformed using a mathematical function. The four bytes in each column are replaced by four new bytes. The output is a matrix with 16 new bytes. This step is not performed in the last round of the algorithm.

iv)AddRoundKey: The 16 bytes of the matrix are considered as 128 bits and another round is started with the same sub processes. If it is the last round then the output is the ciphertext.

2.1.3 JS Algorithm:

The Jumbling and Salting (JS) [2] is an algorithm which uses a symmetric key encryption technique. It is implemented with the AES i.e. Advanced Encryption Standard. the algorithm is divided into two blocks that is jumbling block and Salting block [2].

Jumbling block consists of three other sub blocks: Addition, Selection and Reverse sub blocks [2]. A mathematical process is performed on the process array which is given to the jumbling block. Now as the process array is given to the jumbling block the mathematical function modulus performs some operation that is the characters from the character set is chosen and jumbled. With the help of the modulus function the remainder is obtained by the division process.

During the addition process the random value is generated and the size of the process array is updated. So if the size of the process array was i then the updated size will be i+ (random value generated). In the selection process random characters are chosen from the defined character set. The selection of the character depends on the random value that was generated

during the addition process. In the reverse process the entire string or the process array is reversed. The reversing process is done on the basis of a predefined mathematical condition [2]. Here the mathematical condition is that if the generated value is even then the reversal process takes place but if the generated value is odd then the process array remains as it was after the selection process. The mathematical condition can be changed as per the requirement of the application programmer who is developing the JS algorithm [2].

Now the process array is given to the Salting block in which the random string is added to the jumbled process array. The adding of salt is based on the users sign up time stamp value [2]. The salt is added to confuse the attacks so that it becomes difficult for the attackers to obtain the original process block. Now after the Jumbling and Salting is done the process block is passed to the AES block. In this block the algorithm is predefined for encryption and decryption [2].

2.1.4 Comparison of AES and JS

Table 1. Computational Complexity values of AES and JS algorithm

Algorithm	Plain Text	Cipher Text	Encryption Time	Decryption Time
AES	16	44	86	19
			94	33
			26	21
			26	20
			24	20
JS	16	108	140	28
		154	139	28
		130	139	28
		138	138	28
		126	138	28

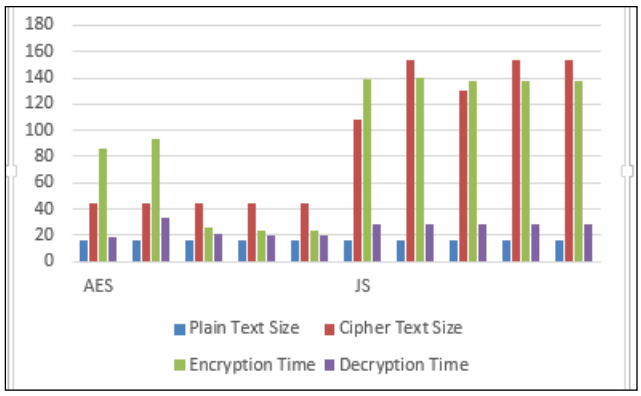

Figure 3. Comparsion Graph of JS and AES Algorithm

2.2. Protocols

2.2.1 SET Protocol:

The Secure Electronic Transaction (SET) [7, 8] Protocol is the most widely used protocol for transactions over an insecure channel. The SET Protocol consists of 3 main parties: customer, merchant and merchant's bank. Digital Certificates are employed by the SET Protocol to ensure authenticity and verification of identity of the involved participants [7].

The principle features of the SET Protocol are:

- All exchanged information between parties is encrypted
- It is mandatory for all parties to be authenticated with certificates
- The merchant will never see the customer details in plain text

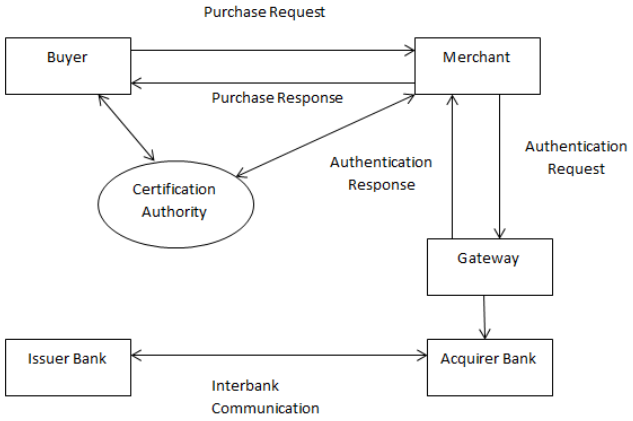

Figure 4. SET Protocol Working

Stages of SET transaction [3]:

i) Purchase initiate Request message: Wallet → Merchant: Merchant opens the envelope of the message using his private key to get the session key SK then the whole message and the signature are decrypted. The hash of the received message is compared with the hash of the originally signed message after decrypting it using the wallet public key.

ii) Purchase initiate response message Merchant → Wallet: The message contains a unique transaction identification number, challenging string, and merchant digital certificate. For the wallet to read and verify this, the public key decryption algorithm is applied twice, and the symmetric key decryption is applied once.

iii) Purchase Request message Wallet → Merchant: Purchase request is a doubly signed message. The wallet partitions the message into two sub messages. The first one contains the Order information message where an envelope is created using Merchant's public key such that it can only read by Merchant. The second message contains only the payment information message and a digital envelope is created using Payment Gateway's public key such that it can only read by Payment Gateway. Each message is signed by the wallet.

iv) Authorization Request message Merchant → Payment Gateway: Merchant sends the second message to the payment gateway from Purchase request message with envelope using the Payment Gateway's public key and adds to it, the additional information related to his financial institution and his digital certificate to the Payment Gateway. The money transfer from the issuer bank to the acquirer bank is done through Electronic Fund Transfer Network (EFTN) and Payment Gateway gets notified.

v) Authorization Response message Payment Gateway → Merchant: After Payment Gateway checks availability, it sends authorization response to the merchant.

vi) Purchase Response message Merchant → Wallet: After the merchant gets the authorization from Payment Gateway, he sends Purchase response message to W and this completes the protocol. For the wallet to read and verify the message, the

public key decryption algorithm is applied twice, and the symmetric key decryption is applied once.

2.2.2 MSET Protocol:

Using SET protocol on GSM network involves heavy computation. This slows down the network. To avoid this, MSET protocol was born. This Modified SET protocol replaces time consuming public key encryption and decryption algorithms by symmetric key cryptography. This protocol claimed about 50% reduction in computation, and 80% reduction in communication overhead [3]. The protocol uses the Transport Layer Security Protocol (TLS) and the Wireless TLS (WTLS) in lower layers to reduce the number of required signature generations within the protocol.

MSET keeps the privacy of the payment information of Wallet away from Merchant and keeps the privacy of the order information of Wallet away from the Payment Gateway. MSET enables the Payment Gateway to verify that Merchant does not alter the Purchase Request message during its processing at Merchant. The only critical assumption made is that the Payment Gateway is a Trusted Third Party.

MSET protocol replaces public key encryption by symmetric key encryption. It is a two-step process. In step 1, the three parties exchange their digital certificates. Three symmetric keys are generated and exchanged. We assume that each party has a lookup table that includes the digital certificate of all the other parties associated with their symmetric keys.

The second step is the transaction step, which like the SET protocol involves 6 steps. They perform the same task as SET, only, they utilize symmetric ley encryption.

i) Purchase initiate Request message: Wallet → Merchant
ii) Purchase initiate response message Merchant → Wallet
iii) Purchase Request message Wallet → Merchant
iv)Authorization Request message Merchant → Payment Gateway
v) Authorization Response message Payment Gateway → Merchant
vi) Purchase Response message Merchant → Wallet

The above calculations show the improvement of MSET over SET protocol.

Table 2. Literature Cited for SET and MSET protocol

Sr No.	Name	Limitations	Inferences
1	SET	• It requires heavy computation over wireless networks which slows down the overall performance. • Adapting systems to work with SET is more complicated • The customer must have a valid digital certificate	• The interacting parties must be isolated from each other • Each party must be able to authenticate itself with a valid digital certificate from certificate authority • Adaptation of the protocol to all systems is key for it to be a standard
2	MSET	• Uses Symmetric key encryption which is considered to be less secure as compared to asymmetric. • A dishonest client can buy goods from a merchant without paying the actual price of the good. • MSET protocol uses a certificate, having no use once the symmetric key is distributed. This causes communication overhead for the mobile application. As a result, the protocol will be useful only for mobiles with high processing power. This contradicts its assumption of being a more viable alternative. • The MSET protocol does not provide the non-repudiation property which is critical for any transaction. • In MSET protocol the payment gateway does not verify whether the payment message sent by the customer is genuine one or not. Hence, the customer can pay less and get away without it being a flawed transaction. • The computational time calculate for SET and MSET were shown to be highly apart with SET having a huge advantage. Since during calculations, the computational time for session key generation and distribution was ignored, the computational time of has come very close to that of SET. Here, the improvement is not as drastic as shown to be.	• Using a combination of symmetric and asymmetric keys is important to achieve a vastly secure algorithm. • Non-Repudiation is a key property to avoid fraudulent transactions. • A payment mediator must have a balance between security and backend computation to find any application. • Amongst the interacting parties, none can be considered as secure and be ignored from a proper check.

3. PROPOSED WORK

JSSecure is a payment gateway that uses Jumbling and Salting Algorithm for encrypting the user details. It is a symmetric key encryption strategy. JSSecure is a payment gateway that is more secure as it uses double encryption technique. In double encryption the user details are encrypted and after the JS Algorithm has encrypted the process array it again encrypts that jumbled process array. The JS algorithm is implemented with the RSA for better security. So that the asymmetric encryption technique will be more efficient and it will be more secure as it uses public key encryption strategy.

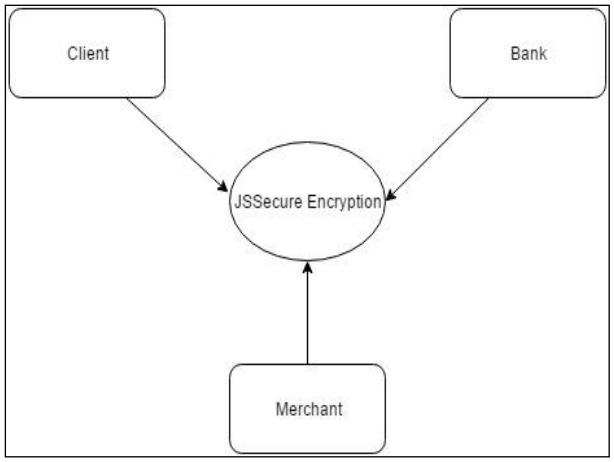

Figure 5. JSSecure Blocks

We intend to increase the cipher text to a limit where it is impossible to crack for the attackers using brute force attack or dictionary attack. The advantage of the JS algorithm is that is uses random length variable to encrypt the plain text so the cipher text varies every time. Adding to this the double encryption strategy boosts the overall security of the system.

4. CONCLUSION AND FUTURE WORK

The rise of E-commerce has led to the development and demand of sophisticated payment gateways. What we are doing with JS Secure is creating an Encryption algorithm which is better suited for the task at hand by using a symmetric double encryption algorithm. It is better at performance than the other already existing algorithms and more secure because of the large cypher text used which protects it from cyber-attacks using brute force or others. As security is our utmost priority much thought and working has been dedicated towards it. The algorithm is just a part of the payment gateway created. One of its major functions is that it can be integrated seamlessly with your website without the need for specialized coding knowledge, which is what clients look at when they specifically mean a payment gateway.

The Payment Gateway is up to the current security standards and it is implemented in such a way that in future also no problems will take place, however if such an exception takes place We can update our work, it is flexible enough to be updated up to the future standards.

5. REFERENCES

[1] Ramesh, Archana, and A. Suruliandi. "Performance analysis of encryption algorithms for Information Security." Circuits, Power and Computing Technologies (ICCPCT), 2013 International Conference on. IEEE, 2013.

[2] Churi, Prathamesh P., Vaishali Ghate, and Kranti Ghag. "Jumbling-Salting: An improvised approach for password encryption." Science and Technology (TICST), 2015 International Conference on. IEEE, 2015.

[3] Shedid, Sabrina M., and Mohamed Kouta. "Modified SET protocol for mobile payment: an empirical analysis." Software Technology and Engineering (ICSTE), 2010 2nd International Conference on. Vol. 1. IEEE, 2010.

[4] Abdel-Hamid, Ayman, Ossama Badawy, and Shreif Bahaa. "PA-SET: privacy-aware SET protocol." Computer Theory and Applications (ICCTA), 2012 22nd International Conference on. IEEE, 2012.

[5] Elminaam, Diaa Salama Abdul, Hatem Mohamed Abdul Kader, and Mohie Mohamed Hadhoud. "Performance evaluation of symmetric encryption algorithms." IJCSNS International Journal of Computer Science and Network Security 8.12 (2008): 280-286.

[6] Nadeem, Aamer, and M. Younus Javed. "A performance comparison of data encryption algorithms." Information and communication technologies, 2005. ICICT 2005. First international conference on. IEEE, 2005.

[7] Guan, Hong-Jun. "The research of SET-based electronic payment system model." E-Business and Information System Security, 2009. EBISS'09. International Conference on. IEEE, 2009.

[8] Zihao, Shen. "An improved SET protocol payment system." 2010 International Conference on Computer and Communication Technologies in Agriculture Engineering.

[9] Yang, Zi-Heng, et al. "An Improved AES Encryption Algorithm Based on Chaos Theory in Wireless Communication Networks." Robot, Vision and Signal Processing (RVSP), 2015 Third International Conference on. IEEE, 2015.

[10] Zhang, Xuewang, and Linlin Wang. "Key technologies for security enhancing of payment gateway." Electronic Commerce and Security, 2008 International Symposium on. IEEE, 2008.

[11] Kurniawan, Daniar Heri, and Rinaldi Munir. "Double Chaining Algorithm: A secure symmetric-key encryption algorithm." Advanced Informatics: Concepts, Theory And Application (ICAICTA), 2016 International Conference On. IEEE, 2016

[12] Churi, P., Kalelkar, M. and Save, B., 2014. JSH Algorithm: A Password Encryption Technique using Jumbling-Salting-Hashing. International Journal of Computer Applications, 92(2).