ISSN: 2277-5528
Impact Factor: 3.145

INTERNATIONAL JOURNAL OF ENGINEERING SCIENCES & MANAGEMENT

COOKIE STEALING ATTACK PREVENTION USING ADVANCED ENCRYPTION STANDARD

Jagdish B. Patil[1], Miss. Nisha Bhalse[2]
[1]P.G. Student, Department of Computer Science and Engineering, IES-IPS Academy, Indore
[2]Assistant Professor, Department Of Computer Science and Engineering, IES-IPS Academy, Indore
ravipatil2580@gmail.com

ABSTRACT
Internet is being used as a resource for gaining information. It is observed that maximum websites are vulnerable to different types of attacks. From security point of view, it is necessary to prevent such attacks. Cookies are sent by web server to browser for storing client information. In cookie stealing, this information is stolen by an attacker. Thus in case of internet banking account, email account and other social media applications accounts, it requires text protection. In this project, I have implemented AES-256 algorithm to protect cookies from attackers. The goal is to inform the user that how credentials can be captured by an attacker using the technique called cookie stealing and how to be alert from this issue.

***Keywords*--- AES, DES, TDES, Vulnerability, Encryption**

INTRODUCTION

To facilitate a browser-server stateful interaction, in a stateless protocol, cookies have been designed. Cookies, also known as HTTP (Hypertext Transfer Protocol) cookies, are generated by the server and get modified. These cookies are raw data which sent by server and stored on client system. Then those are transmitted between browser and server at each interaction for further retrieval to allow user. The HTTP communication protocol is designed for stateless operation between server and browser but for server using only HTTP protocol, it is not possible to correlate one request from a client with previous one by the same client. To overcome this problem, cookies are the common solution.

For the first time when the client contacts the web server, the server generates a session id which is a part of cookie. Whenever a user enters a URL (Uniform Resource Locator) in a browser, the browser searches its local memory to check whether it has any cookie associated with it. If the browser found cookie, it is then inserted in the query sent to the server. Cookies are stolen by an attacker because it contains important information of user which can be misused which may result in economical loss for the user. Following type of data can be stored in cookies.

- Credentials such as user names and passwords.
- Session data and session id
- Tracking information about the user.

This information inside the cookie is easily accessed by an attacker using which an attacker can steal session and compromise accounts. An attacker having knowledge to insert malicious JavaScript code into vulnerable website is perfect in cookie stealing.

Now a day, it is very easy for an attacker to inject vulnerable code through JavaScript. Due to this activity, scripting attacks are common security threats in web environment. To avoid these attacks, data needs to be encrypted into such a format which is unreadable for an unauthorized person. Advance version of data encryption standard is AES.

From January 1997, effort was started for the development of AES by National Institute of Standards and Technology. AES is a symmetric key encryption algorithm. 15 algorithms were selected at first but this number reduced down to 5 after detailed analysis. These final five algorithms naming MARS, RC6, Rijndael, Serpent and Two fish were iterated block ciphers. These all were in a queue to be qualified as a algorithm for AES. But after final analysis, Rijndael was selected as an algorithm for AES.

Across a variety of platforms, for both hardware and software, AES designed to be efficient. Faster performances, resistance to all known attacks, higher encryption speed are the characteristics of Rijndael AES algorithm.

LITERATURE SURVEY

According to Rodica TIrtea, any website can issue cookies. The information stored in a cookie is coded in a plain text and whenever user visits the web page, this information can be modified each time. As a result of this, retrieval of cookie is possible. In cookie stealing attack, user trying to access HTTP is get impersonated. Some of these attacks may expose cookies. For example, in case of cache sniffing, if an attacker can access the browser then cookie contents can also be obtained by an attacker. In case where a web application maliciously gathers data from the user, cross-site scripting cookie sniffing (XSS cookie sniffing) takes place. This XSS attack hijacks account, changes user setting, can steal or poison the cookie. The attacker can also capture cookie and extract data from it. To overcome such attacks, existing solutions provide identification of vulnerabities. Also session management of vulnerabilities and attack is also presented. But all the security requirements are not provided by existing solutions.

Charmie Prajapati surveyed on cookie stealing in Facebook account by using different mechanisms and concluded that HTTP cookies are helpful for storing user specific data into browser to reduce overhead on server round trips. It is observed that “datr” cookie authentication mechanism is used by Facebook. User session cookies can be used by an attacker to inject in browser. Due to this, browser will redirect an attacker to user’s account state at that time.

According to HTML 5 security group (APWG) surveyed, the cookie can steal information from the storage and global variables. It supports local storage where a developer can create local storage for an application and can store some information. From anywhere in the application, this storage can be accessed and HTML 5 offers great flexibility on the client side. Local storage can be accessed through JavaScript. This allows an attacker to steal information via XSS, if the application is vulnerable to an XSS attack.

According to Saurabh Jain, malicious JavaScript code can be inserted into a web page of vulnerable website. This activity is performed by an attacker to steal cookies. Then user cookies are collected by this script and sent to an attacker. For this purpose, an attacker uses code snippet of cookie stealing as <SCRIPT> document.write(document.cookie). An attacker can also gain access to vital portions of information like browser cache and browser history. JavaScript snippets append to web page and execute on browser. It may exploit malicious effects on client system. Some browser engines that render JavaScript differently provide results in inconsistent functionality.

Image and text encryption and decryption solution has been provided by K.R.Saraf by using AES 128 bit algorithm. But as images have large data size, similar methods not used to protect images as well as text from unauthorized access. By Shraddha Soni, comparison between AES and DES algorithms is given. Both these algorithms are useful to encrypt user data so as to prevent attack like cookie stealing. DES algorithm is further modified into TDES to improve the performance of an algorithm. However, in case of TDES as the name suggests, it performs three times as many rounds as DES. So it is much slower. Also it uses 64-bit block size and hence it does not provide more efficiency and security. For TDES algorithm larger block size is desirable.

TEXT ENCRYPTION BY USING AES ALGORITHM

AES stands for Advance Encryption Standard is mainly an advance version of DES which stands for Data Encryption Standard. Although existing methods use DES, TDES (Triple DES) algorithms for data encryption, but comparatively AES-256 is the better option than those.

A. AES algorithm rounds

In case of cookie stealing attack, we have used AES-256 algorithm. Generally 128,192 or 256 is the length of the cipher key K for the AES algorithm. AES uses repetitive symmetric block cipher method so it repeats the same steps for multiple times. It is a symmetric key encryption algorithm and operates on a fixed number of bytes. This iteration of steps is called a round. For the execution of AES algorithm, number of rounds are performed which depends on the key size.

Algorithm	Key Size (Bytes)	Block size (Bytes)	Rounds
AES -128	16(128 –Bit)	16	10
AES-192	24(192-Bit)	16	12
AES-256	32(256-Bit)	16	14

Table 1: AES Encryption Round Table

For both cipher and Inverse cipher, a round function is used by AES algorithm. Plain text is the input for AES algorithm which needs to be encrypted from the security point of view. It is converted into a 4*4 array, called a state. Four transformations namely *AddRoundKey*, *SubBytes*, *ShiftRows* and *MixColumns* are involved in this operation on a state.

B. Working of AES algorithm

With the aid of mathematical concepts, above mentioned transformations are operated. AES algorithm works in a following manner. As mentioned earlier, different number of rounds takes place in execution of AES algorithm. Since we have implemented AES-256 algorithm, it consists of 14 rounds.

1. Initial Round:

a*) AddRoundKey* - It is a simple XOR addition of round key and a portion of expanded key into plaintext. Each byte of the state is combined with a block of the round key using bitwise XOR.

2. Next Rounds:

a) *SubBytes* - Each byte in the state is operated by this *SubByte*. A non-linear substitution step takes place where each byte is replaced with another according to a lookup table. These non-linear substitutions performed in the field converts AES to non-linear cryptographic system.
b) *ShiftRows* - Throughout the AES algorithm, diffusion occurs since *ShiftRows* operates on individual rows of the state. Considerable changes occur for all three rows except first row where there is no change. It means a

transposition step where the last three rows of the state are shifted cyclically upto certain number of steps.

c) *MixColumn* - It is operated on the individual columns of the state. Throughout the AES algorithm, diffusion is provided by this step also. Thus it is a mixing operation which operates on the columns of the state, combining the four bytes in each column.

d) *AddRoundKey* - It works in the same way as discussed in case of initial round. In each round, it functions in the same way as in the initial round.

3. Final Round:

In final round all three steps excluding mix column takes place as follows.

a) *SubBytes*
b) *ShiftRows*
c) *AddRoundKey*

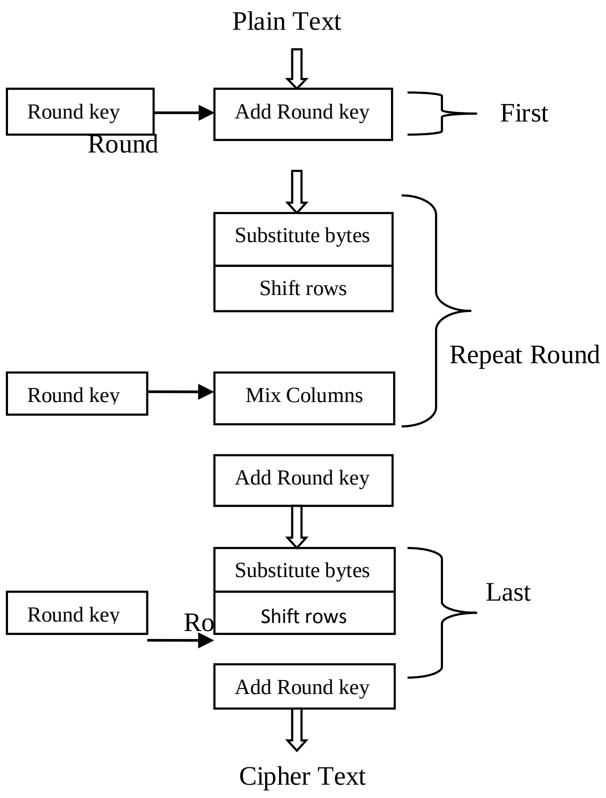

Fig 1 : AES Encryption Algorithm

WORKING OF DATA ENCRYPTION USING AES-256 ALGORITHM

Generating a Cipher object

An instance of cipher is easily obtained and for both encryption & decryption, it requires the same process.

Cipher = Cipher.getInstance("AES/CBC/DES etc");

Once we have an instance of the cipher, it becomes easy to encrypt and decrypt data according to the algorithm.

Step 1 : Generate an AES key using Key Generator.
Initialize the keysize to 256 bits.
KeyGenerator.init(256);
SecretKey secretkey = keyGenerator.generatekey();

Step 2 : Generate an Initialization Vector.

Step 3 : Initialize the cipher for encryption.

Cipher.init(Cipher.ENCRYPT_MODE,secretkey);

Step 4 : Encrypt the data by following way.
a) Initialize the data which is of type string.
b) Convert the input text to bytes.
c) Encrypt the bytes using doFinal method.

RESULT FOR TEXT ENCRYPTION USING AES COMPARED TO TDES

In this cookie stealing prevention module, data encryption using both AES-256 and TDES algorithms shows that AES algorithm is better than TDES because it takes minimum time for data encryption. Thus AES-256 improves encryption accuracy and secures data contents. It also minimizes storage data capacity into database.

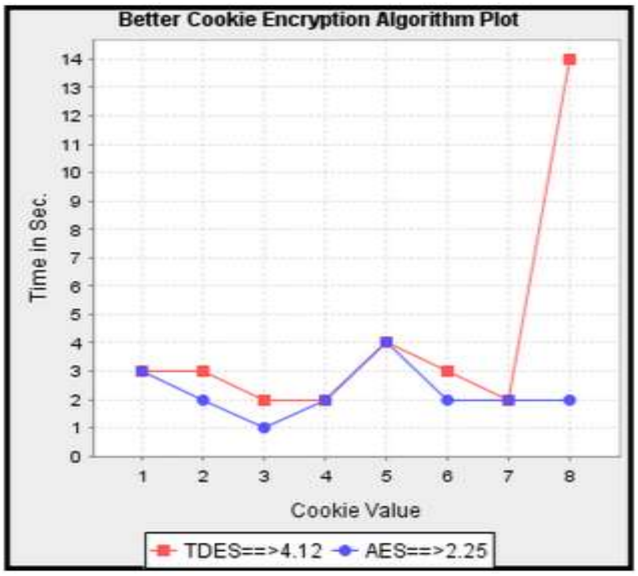

Fig 2: AES Vs TDES Algorithm

Comparison detection performance:-

The simulation result showed that compared to TDES, AES-256 has better performance. The comparative analysis of both AES-256 and TDES encryption algorithm is as follows.

Parameter	AES	TDES
Encryption Time(in sec)	2.25	4.12

Table 2: Comparative Result of AES Vs TDES

VI Conclusion

In this world of internet, it is a strong requirement to provide security in the form of encryption for data since it is the basic need for communication system, if we focus

ISSN: 2277-5528
Impact Factor: 3.145

on the seriousness of web attacks. This paper shows successful implementation of text encryption. Although previous technologies also working for the same, but comparatively this one provides main benefit by encrypting data in less time duration than older ones. Proposed methodology is used for user privacy security concerning to the user credentials so that only authorized user can decrypt the encrypted data. We have used AES-256 bits algorithm which uses block size of 32 bytes i.e. 256 bits and 14 rounds of the algorithm. Thus more encryption provides stronger security as compared to existing solutions.

FUTURE SCOPE

The proposed system can be extended for the prevention of web based attacks on runtime overhead approaches at server side centralized protection for hosted applications. It is an effective result for social media sites like Facebook, LinkedIn, WhatsApp, Google+ and Twitter. It may also be adapted for more precise analysis of JavaScript vulnerabilities, dynamically in smart phones and other operating systems for all web browsers. We also plan to develop a real time web based attack detection system for e-banking, social media applications etc., deploy it and then test the performance of the system in the real world.

REFERENCES

[1] K. R. Saraf, Vishal P. Jagtap, Amit K. Mishra, "*Text and Image Encryption Decryption Using Advance Encryption Standard*", Vol 3, Issue 3, June 2014

[2] William Roche, "*The Advanced Encryption Standard, The Process, Its Strengths and Weaknesses*", University of Colorado, CSC 7002, Final Paper May 2006.

[3] J. Daemen and V. Rijmen, "*AES Proposal: Rijndael, AES Algorithm Submission*", September, 1999

[4] Shraddha Soni, Himani Agrawal, Monisha Sharma, "*Analysis and Comparision between AES and DES Cryptographic Algorithm*", Vol 2, Issue 6, December 2012

[5] Y. Xian, B. Sun, H. Chen, S. Guizani, R. Wang, "*Performance Analysis of AES*", GLOBECOM 2006

[6] Saurabh Jain, Deepak Singh Tomar, Divya Rishi Sahu, "*Detection of JavaScript Vulnerability At Client Agen*", Vol 1, Issue 7, August 2012

[7] D. Flanagan, "*JavaScript: The Definitive Guide*", 4th Edition, December 2001

[8] Rodica Tirtea, Claude Castelluccia, Demosthenes Ikonomou, "*Bittersweet cookies. Some Security and Privacy Consideration*", enisa, June 2012

[9] David Crystol, "*HTTP Cookies: Standards, Privacy and Policies*", Vol 1, Issue 2, 2001

[10] John Park, Ravi Sandhu, "*Secure Cookies on the Web*", July-August 2000

[11] C.A. Vlsaggio, L.C. Blasio, "*Session Management Vulnerabilities in Today's Web*", Vol 8, Issue 5, Sept-Oct 2010

[12] Hossain Shahriar, Komminist Weldemariam, Thibaud Lutelliar, Mohammad Zulkernine, "*A Model Based Detection of Vulnerable and Malicious Browser Extension*", IEEE 2013