International Journal of Computer Applications (0975 – 8887)
Volume 70– No.2, May 2013

Implementation of White-Box Cryptography in Credit Card Processing Combined with Code Obfuscation

Nilima Yadav
M.Tech student
Mody Institute Technology (FET)
Lakshmangrah, Rajasthan

SarveshTanwar
Asst.Professor
Mody Institute of Technology (FET)
Lakshmangrah, Rajasthan

ABSTRACT

White box cryptographic algorithms aim to denying the key readout even if the source code embedding the key is disclosed. Full-privileged attack software shares a host with cryptographic software, having complete access to the implementation of algorithms, dynamic execution (with instantiated cryptographic keys) can be observed and internal algorithm details are completely visible and alterable. Chow proposed a new technique to secure cryptographic algorithms and key against white-box attacks, called white-box cryptography. Another technique such as obfuscation is mainly designed to facilitate securing of e-commerce and e-banking applications, which often embed cryptographic keys and critical information. In general, it can be used to protect all distributed client software where an owner loses control or where the user wants to protect against automated attacks. As we know main challenges in modern cryptography does how to encrypt or decrypt content without directly revealing any portion of the key and or the data and how to perform strong encryption mechanisms know that hackers can observe and or alter the code during execution. Considerding these problems we have suggested a novel approch in e–banking system (credit card processing) using white box cryptography to encrypt the key and obfuscation which gives a strong encryption. Combination of these two concepts gives a new level in modern cryptography as well as optimizes its performance and additionally we will make end points (Client and server) secure.

Keywords

White-Box cryptography, Credit card, Code obfuscation, White –box attacks.

1. INTRODUCTION

The initial objective of cryptography has been to design algorithms and protocols to protect a communication channel next to eavesdropping [10]. In black box cryptography the attacker only has access to the input/output of the algorithm. Today we even encounter an even worse attack model, where cryptography is deployed in applications that are executed on open devices (such as Personal Computers or on a tablet, Smartphone without exploiting secure elements [11]. White box cryptography is the new technique against attacks on white box attack environments. In white box attack model, the attacker is even stronger than in black box attack model, and the attacker can monitor all intermediate values [3]. Therefore, safety algorithms are needed against all operation steps being exposure. In white box attack, an attacker has full access to the software implementation of a cryptographic algorithm where the binary is completely visible and alterable by the attacker and the attacker has full control over the execution platform (CPU calls, memory registers, etc.) [6].The challenge is that white-box cryptography aims to address is to implement a cryptographic algorithm in software in such a way that cryptographic assets remain secure even when subject to white-box attacks.

2. WHITE BOX CRYPTOGRAPHY

The term “white-box cryptography” describes a secure implementation of cryptographic algorithms in an execution environment, such as on a desktop computer or a mobile device that is fully observable and modifiable by an attacker [1]. It is different from black-box cryptography, where an algorithm’s internal processing data is unavailable to an attacker. The white-box environment puts hard additional restrictions on implementations of the cryptographic algorithms [9]. In traditional cryptography, a black-box attack describes the situation where the attacker tries to obtain the cryptographic key by knowing the algorithm and monitoring the inputs and outputs, but without the execution being visible [8]. White box cryptography addresses the much more severe threat model where the attacker can observe everything, can access all aspects of the target system application, and may have the black-box knowledge of the crypto algorithm. White-box cryptography is aimed at protecting secret keys from being disclosed in a software implementation [2].

2.1 Attacks in White Box Cryptography

There are three types of attacks toward cryptographic module - Black-box attack, Gray-box attack and white-box attack.

2.1.1 Black-box attack

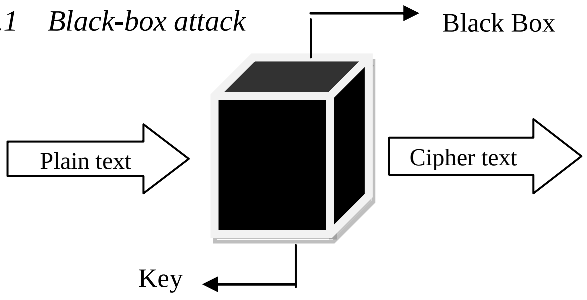

Figure 1: Black Box Attack

- Watches inputs and outputs
- Controls input text
- No visibility of execution

2.1.2 Gray Box Attack

- The Grey box scenario assumes that the attacker has partial physical access to the Key or that it is “leaking” so called side channel information.

- Side Channel Analysis attacks (SCA) exploit information leaked from the physical implementation of a cryptographic system.
- The leakage is passively observed via timing information, power consumption, and electromagnetic radiations.

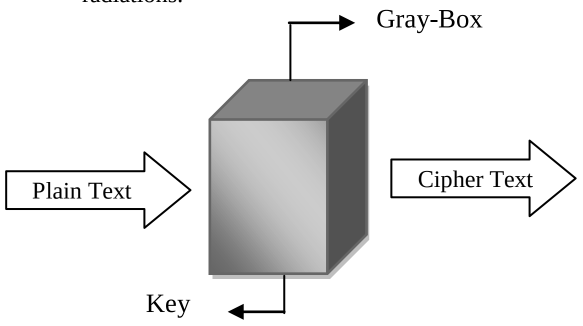

Figure 2: Gray Box Attack

2.1.1 White-box attack

- Attacker can observe everything.
- Attacker knows algorithm.
- Watches inputs, outputs, and intermediate calculations.
- Controls input text.
- Full visibility into Memory (debuggers and emulators).

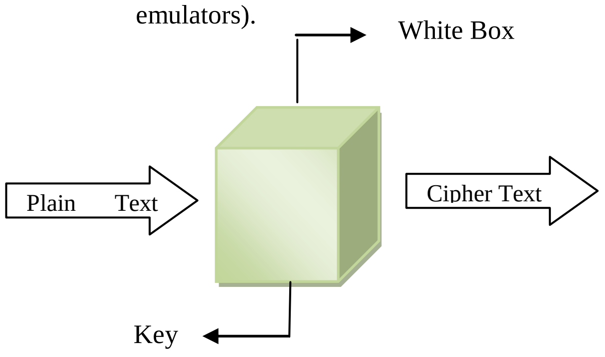

Figure 3: White Box Attack

2.2 Need of white box cryptography [14]

Standard cryptographic models suppose that endpoints, PC and hardware protection tokens are to be trusted. If those endpoints reside in a potentially hostile environment then the cryptographic keys may be directly visible to attackers monitoring the application execution while attempting to extract the keys either embedded or generated by the application from memory [14]. There we need white box cryptography to secure key as well as end points.

- Modern cryptographic algorithms are planned so that an encrypted message can be considered safe while it is travelling between the endpoints at which it is encrypted and decrypted.
- However, commonly-used cryptographic algorithms were not designed to operate in an environment in which their execution could be observed.
- The crucial points are the endpoints of cryptographic communication, where a message is encrypted, signed, or decrypted, because the secret cryptographic keys are required for each such operation.
- If an adversary has access to the device on which a message is cryptographically processed, and if the cryptographic keys are not sufficiently protected, he can extract these keys. This can invalidate the whole security system.
- Most hardware platforms, including personal computers, mobile phones, and embedded systems, provide an insecure execution environment for cryptographic operations.
- Adversaries can monitor the program Code and memory of such devices via special tools, and gain access to the secret cryptographic keys, which are usually internally revealed at some point during processing.
- White-box cryptography has emerged to specifically address the problem of entrusted endpoints in cryptographic communications.
- If the target device resides in a hostile environment then the cryptographic keys may be directly visible to an attacker.
- An attacker may be able to monitor the application and extract one or more cryptographic keys embedded or generated by the application.
- This is a common problem for PCs, set top boxes and other devices where DRM, conditional access or other security sensitive applications are involved

3. OBFUSCATION

Obfuscation is a typical code protection technique used to make your code hard to read. Obfuscation changes the name of your classes and methods to unreadable or meaningless characters, making it more difficult for others to understand your code [5]. Object oriented programming is useful everywhere because it offers several advantages to read, adapt or extent code. However, this way of programming in modules leaves many traces into an executable and reverse-engineers will exploit these traces as good as possible to reconstruct the original source code [7]. Therefore, programmers developed several techniques to maximally difficult to understand the internals of a program so that analysis becomes very hard. This technique applies one or more transformations to code that make a code more resistant to analysis and tampering, but preserve its functionality.

4. PROPOSED APPROACH

In todays world where e-banking is common to every second individual the concept of secuing the tranctions becomes very importatnt.so in order to facilalte the same we got motivated to come out with the concept of wbc (whitebox cryptography) coupled with code obfucation to develop this idea.

4.1 Statement of Problems

4.1.1 Problem-1

What security technique must be followed to make the data as well as the key which is used to encrypt the data?

4.1.2 Problem-2

How to make sure that, hacker is not able to alter/modify the environment (Code which is running) in which data is being encrypted and also secured at the end points?

1.1 4.2 Proposed Approach Steps

Credit card details are encrypted using AES,RSA and SHA with different combination of key size by taking reference from Marjanne Plasmans algorithms [4] and which is coupled with code obfuscation in real time scenario as part of my this paper.

STEP-1: Credit card details (Names, CVV No, Date of birth, Amount) are submitted from user and consider it as Plaintext for following algorithm.

STEP-2: Plaintext P is encrypted using AES and key KCEK.

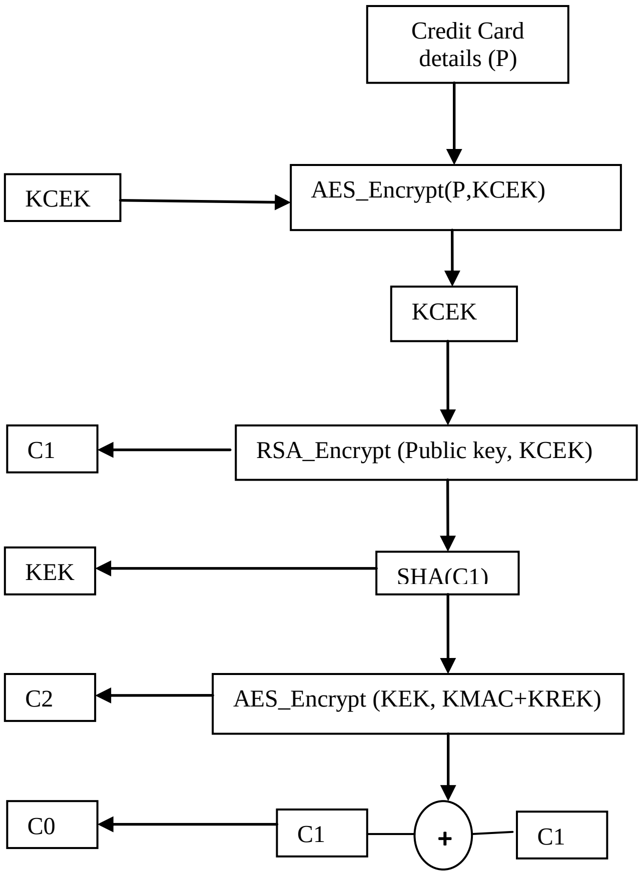

Figure 4: Content Encryption Flow Chart

STEP-3: KCEK is encrypted using RSA which is asymmetric key algorithm and used public key and get the cipher text 1 (C1).

STEP-4: Calculating the hash of C1 using SHA and generate MAC value of KEK.

STEP-5: Again encrypting the Hash, KEK using two randomly generated and concatenated key KMAC and KREK using AES algorithm.

STEP-6: Concatenate C1 and C2 and get new value of cipher C0.

$$\boxed{C0 = C1|C2}$$

- Content Encryption Key KCEK. KCEK is a randomly generated.
- The client sends C0 to the recipient device, after receiving C0; the receiver splits it into C1 and C2 and decrypts the cipher text using private key.

4.2 Module Description

The whole process is divided into four modules, module-1 describes encryption of plaintext and key (using AES-128 bits, RSA, SHA-128 bits), module-2 describes encryption of plaintext and key (using AES-256 bits, RSA, SHA-521) and module 3 and 4 describes security at end points (Client and server) using code obfuscation and doing comparison between module-1 and module-2 using different parameters respectively (computation time, memory and Output Bytes).

4.3.1 Module-1: Encrypt the plaintext and key using AES-128, RSA and SHA-1

In first module, encrypt the credit card values and key using AES-128, RSA and SHA-128.

- Plaintext for this process is come out in form of credit card details (Name, card No, CVV number and, amount).
- Card No, CVV No, amount to be detected from credit card duly distinguishes by separator.
- Above these details are concatenated with separator.
- We are using AES-128 bit algorithm to encrypt plaintext as know as credit card details.
- SHA-128 is used to calculate the MAC value and RSA is used for Key exchange between two parties.
- We will then calculate MAC value for the encryption done by RSA.
- Then we will encrypt the MAC using AES Algorithm.

4.3.2 Module-2: Encrypt the plaintext and key using AES-256, RSA, and SHA-2.

After completing first module come into second phase where using again e-banking process system known as credit card processing system with AES-256,RSA AND SHA- 2.

- Plaintext for this process is come out in form of credit card details (Name, card No, CVV number and, amount).
- Above these details are concatenated with separator.
- By using AES-256 bit algorithm to encrypt plaintext as know as credit card details.
- SHA-2 is used to calculate the MAC value and RSA is used for Key exchange between two parties.
- Calculate MAC value for the encryption done by RSA.Then encrypts the MAC using AES Algorithm.

4.3.3 Module-3: Security at end points (Client and server) using code obfuscation

Commonly-used cryptographic algorithms were not designed to operate in an environment in which their execution could be observed. The crucial points are the endpoints of cryptographic communication, where a message is encrypted, signed, or decrypted because the secret cryptographic keys are required for each such operation. So in module-3 securing both ends client side as well as server side.

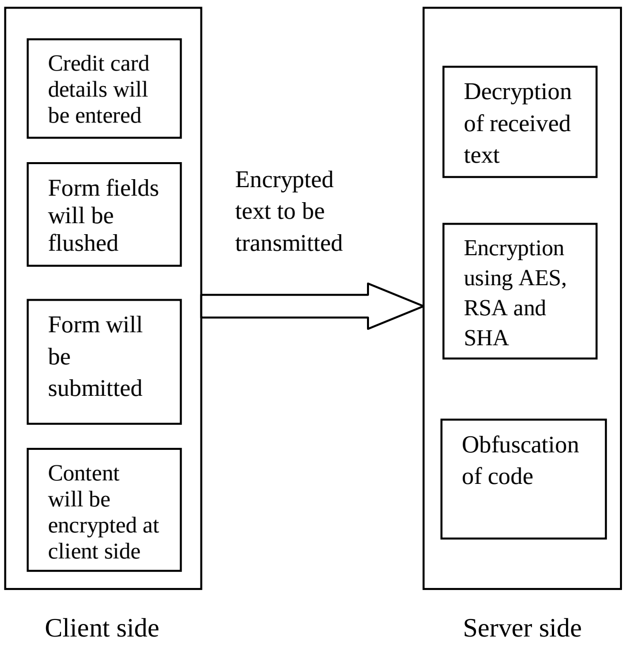
Credit card details will be entered
Form fields will be flushed
Form will be submitted
Content will be encrypted at client side
Encrypted text to be transmitted
Decryption of received text
Encryption using AES, RSA and SHA
Obfuscation of code
Client side
Server side

Figure 5 : Security at end points (Client and server) using code obfuscation.

5. EVALUATION PARAMETERS

Performance of encryption algorithm is evaluated considering the following parameters.

- Computation Time
- Memory usage
- Output Bytes

6. TECHNICAL ASPECTS

The following technologies will be use while devloping this project.

- Visual studio 2010 with .net framework 4.0
- Code obfuscation framework 4.0
- Ajax 4.0
- Cryptanalysis framework 4.0

7. CONCLUSION

Give the attention to prevent adversaries from tampering, reverse engineering, and illegally redistributing software. Here summed two protection techniques white box and code obfuscation which can be embedded in software to protect against analysis and tampering attacks. Another section shows the comparison between two different combinations of algorithms (AES-128/256, RSA and SHA-1/2) which solve the problem of key extraction.

8. FUTURE SCOPE

The Implementation of the concept has been done keeping in view the security at the Endpoints. In the future scope we will take initiatives to secure the communication medium as well as from malicious attack and unauthorised access over the network.

9. ACKNOWLEDGMENT

The authors wish to thank Dr. Prema K. V. head of department (HOD- CS) MITS, Lakshmangrah, Rajasthan for her constant advise and rejuvenating thoughts which helped us make this paper presentation in a better way.

10. REFERENCES

[1] S. Chow, P. Eisen, H. Johnson, P.C. van Oorschot , 16 August 2002,"White-Box Cryptography and an AES Implementation", (SAC'02), Ottawa, Canada.

[2] Marc Joye,2008, "On White-Box Cryptography" ,Thomson R&D France, Technology Group, Corporate Research, Security Laboratory 1 avenue de Belle Fontaine, France Security of Information and Networks,Tra®ord Publishing.

[3] S. Chow, P. Eisen, H. Johnso1, P.C. van Oorschot,2002 ," A White-Box DES Implementation for DRM Applications", Cloakware Corporation, Ottawa, Canad , Carleton University, Ottawa, Canada .

[4] Marjanne Plasmans ,2005.,"White-Box Cryptography for Digital Content protection", department of mathematics and computer science,

[5] James Muir, "White-Box Cryptography", Cryptography Developer Irdeto Canada MITACS Workshop on Network Security and Cryptography, 24 June 2010.

[6] James Muir,"Understanding the Advantages of Irdeto's White-Box Cryptography", December 2012.

[7] Hamilton E. Link, William D. Neumann,"Clarifying Obfuscation Improving the Security of White-Box DES", Sandia National Laboratories, 2004.

[8] Wil Michiels, "Mechanism for Software Tamper Resistance: An Application of White-Box Cryptography",Philips Research Laboratories Eindhoven, The Netherlands Department of Mathematics and Computer Science, DRM'07, Alexandria, Virginia, USA,Copyright 2007 ACM, October 29, 2007.

[9] Olivier Billet, Henri Gilbert,"Cryptanalysis of a White Box AES Implementation", Charaf Ech-Chatbi France T´el´ecom R&D, France, 2009.

[11] Brecht Wyseur Brecht, "White-Box Cryptography: Hiding Keys In Software", NAGRA Kudelski Group, Switzerland, 2009.