Chapter One

Introduction

An essential aspect in the world of software development is the security of Software, and it's the most critical issue nowadays. Software security contains and protects information and information systems from unauthorized access, use, disclosure, disruption, modification, or destruction in providing confidentiality, integrity, and availability.

People must hide their shared data through network communication to maintain their privacy [1].

But some problems affect software protection. Many Software gets distributed over the Internet. Once Software is distributed to a client machine, the software owner loses all control of the (client) Software. An illegal be obtained when Software is cracked, where illegal copying and distributing of cracked Software is a form of copyright infringement.

Attackers can violate the copyrights of Software by applying reverse engineering processes that give attackers a chance to understand the behaviour of Software and extract proprietary algorithms and critical data structures from it. Many reverse engineering tools can provide access control software's data and makes it easier for adversaries and reverse engineers to analyze Software and steal intellectual property [1] [2][3].

These tools include dis-assembler (translate binary code into readable assembly language) and de-compiler (translate byte code to source code). And it is difficult to enforce copyright laws and penalties upon users. "**Copyright is a form of protection grounded; it's the certain right for an author to original work they have written**". Therefore, copyright infringement is the illegal distribution and unauthorized copying of copyrighted Software. Protecting the copyright of Software is one of the most critical issues nowadays [3][4].

Software protection is required, and we need to solve all problems of software protection:

1. The Software contains secret, confidential or sensitive information. An obfuscation technique is used to protect the data.
2. It's necessary to preserve software implementation from reverse engineering.
3. There is the program's execution, where critical code is executed, and some confidential data is accessed.

During execution, one must protect the code and data from malicious intents, such as dynamic analysis and tampering. All these elements need to be sufficiently strengthened to guarantee data confidentiality and secure program execution to the user [23][4].

The obfuscation technique is essential to protect Software from the risk of reverse engineering. It transforms the program into another functionally equivalent program while hiding its internal logic to enhance the interpretation's difficulty, thus obstructing software analysis [1][2].

This research presents a novel solution aimed at protecting the copyright and distribution of Software; we developed ***Distributed obfuscation model technique*** that protects Software from the above problems. It operates at a high level of security against reverse engineering by integrating multiple protections.

1. **Research Objectives**
2. Help protect the intellectual property contained within Software by making reverse-engineering a program difficult and impossible.
3. Protect licensing mechanisms, unauthorized access, and shrinking the executable size.
4. Protect copyright and distribution of Software over client/server.
5. Improve the level of software protection by integrating many levels of obfuscations and combining different techniques to complicate the process of reverse engineering analysis.
6. To explore the disadvantage of the existing model.
7. Improve the protection of Software against reverse engineering analysis.
8. Develop robust protection techniques against reverse engineering analysis.
9. To ensure data confidentiality from any reverse engineering while distributing the Software over client devices and transmitting data through the network using DH and AES algorithms.
10. To read and analyze existing obfuscation techniques and cryptosystem models.
11. To define obfuscation techniques.
12. To develop a new distributed obfuscation technique that protects Software against reverse engineering analysis.
13. To simulate and test the proposed distributed obfuscation technique, including obfuscation/ de-obfuscation: key Generation and critical exchange.

Comparison between our model and existing models through time and critical length.

1. Model designed to prevent mass wiretapping and malicious corruption of Transmission Control Protocol -TCP traffic on the Internet.

1. Contributions

Even though the ***Distributed obfuscation model technique*** with multi-levels of obfuscation is robust and more secure than traditional techniques,

The main contribution of this thesis is presenting a suitable solution to protect Software, copyright and distribution of Software over client/server against reverse engineering analysis, and these contributions are summarized as follows:

1. Develop robust techniques that integrate multi-levels of obfuscations, traditional methods depend on only one level of obfuscation, and these techniques suffer from a reverse engineering analysis. So, depending on multi-levels of obfuscation make reverse engineering analysis impossible.
2. Protect Software against all attackers and reverse engineering analysis.

3. Develop new techniques using AES and DH encryption algorithms to improve the protection of Software and confidentiality of using this technique.
4. Another significant contribution, which improves the security of AES and DH algorithms against reverse engineering analysis, this enhancement in security is possible by using random key generation ideas based on crucial exchange and generation processes using the Diffie Hellman principle.
5. Protect data, Software and intellectual property from unauthorized access by another user during the distribution of Software over client/ server devices.
6. Motivation
7. The major problem of software protection is the distribution of Software (Client / Server) over the client devices, in which the owners lose control of their Software.
8. Over the last years, client devices have become more powerful. An attacker with malicious intent can violate the copyrights and tamper with the Software via applying much analysis and reverse engineering tools.
9. Illegal access could be obtained when the Software is cracked, where unlawful copying and distributing of cracked Software is a form of copyright infringement.
10. **Threat Model**

The main aim of threat modelling is to identify the essential functionalities of the software/ model and protect it.

We begin threat modelling by focusing on four key questions: ref [book]

"Threat Modeling: Designing for Security Published by John Wiley & Sons, Inc. 10475 Crosspoint BoulevardIndianapolis, IN 46256 www.wiley.com Copyright © 2014 by Adam Shostack Published by John Wiley & Sons, Inc., Indianapolis, Indiana Published simultaneously in Canada "

1. What are you building?
2. What can go wrong?
3. What should you do about those things that can go wrong?
4. Did you do a decent job of analysis?

Transmit data through the network that connects a server to several clients and maintains a shared secret Key that a different attacker can access. The Software can be attacked by reverse engineering analysis; software protection techniques suffer from a set of threats that our model will solve:

1. An unauthorized person, such as a contractor or visitor, might gain access to a company's computer system.
2. Confidential information might be intercepted as it is being sent to an authorized user.
3. Users may share documents between geographically separated offices over the Internet or Extranet, or telecommuters accessing the corporate Intranet from their home computer can expose sensitive data as it is sent over the wire.
4. Electronic mail can be intercepted in transit.

Threat to the client:

1. Virus: attached to an executable file, requires human action to spread, some damage software and hardware of the user.

Danial of the service (DOS): An attack occurs on network structure during data transmission, and it disables a server from servicing its client. (ref: Denial of Service Attack Techniques: Analysis, Implementation and Comparison

Khaled M. Elleithy, Computer Science Department, University of Bridgeport, Bridgeport, CT 06604, USA, Drazen Blagovic, Wang Cheng, and Paul Sideleau, Computer Science Department, Sacred Heart University

Fairfield, CT 06825, USA)

the attacker sends a lot of requests to the server in a tempt to make it down, flooding the server with large packets of invalid data

1. Problem Statement

The need for robust software protection techniques against tampering and reverse

engineering analysis is highly needed and recommended nowadays, and these techniques should address the need for more confident Software in an untrusted environment.

Many studies have investigated one-to-one protection, but there need to be more studies constructing many-to-one security. Most of these approaches protect intellectual property and are seen as trade secrets.

Research Aims

- The research develops new techniques depending on obfuscation techniques to prevent

Reverse engineering analysis and modifications during the distribution of the Software over client/server over client device.

Improve the level of software protection by integrating obfuscation. The AES encryption technique makes reverse engineering analysis hard, thus compiling the programs and making them infeasible.

1. Research Questions

This study is meant to address the following questions and seek answers to them:

1. Can we use the Advanced Encryption Standard (AES) algorithm combined with the Diffie-Hellman (DH) exchange key algorithm in the obfuscation structure to improve the

security of obthe fuscation quality of obfuscation and to protect the Software against reverse engineering analysis
2. Research Hypotheses
3. Research Structure

This section briefly summarizes what each chapter explains.

Chapter One: Background.

This chapter describes the thesis idea, justification for choosing the research, problem statement, research aims, objectives, questions, hypotheses and research structure.

Two: Theory and Literature Review.

Chapter Three: Research Methodology and procedure (Proposed model.)

Chapter Four: Results and discussions

This chapter talks about my obfuscation model thesis use.

Chapter Five: Conclusions and recommendations

This chapter presents the testing and result.

Chapter Six: References

This chapter presents the conclusion and future work.

Chapter seven: Appendixes

Chapter Two

Background

- ### Introduction

This chapter will focus on symmetric encryption, the simplest type of encryption. In symmetric encryption, the KeyKey used to decrypt is the same as the KeyKey used to encrypt. You'll start by learning about the weaker forms of symmetric encryption, the classic cyphers that are only secure against the most illiterate attacker, and then move on to the more critical conditions that are secure forever.

- Encryption
- Definition

Cryptography is derived from the Greek word 'crypto', meaning secret ', and graph means writing used to conceal a content of a message from all except the sender and the receiver used to authenticate the correctness of the statement to the recipient. Cryptography ensures integrity, availability and identification, confidentiality, and authentication of the user and the user's security and privacy of data provided to the user [1].

The word cryptography may use alternatively with the words cryptology or cryptanalysis. Still, each of them has its meaning that differs slightly using the term "Crypto", which comes

from the Greek "Krypto" to mean hidden and ends with "graphy", which means writing. Hence, the

whole word cryptography means hidden writing done as an output of the encryption

process in the remote system [19]. Cryptanalysis is what the layperson calls breaking the code.

The areas of cryptography and cryptanalysis together are called cryptology [20].

Cryptographic systems provide privacy and authentication in computer and

communication systems. As shown in figure [3.1], encryption algorithms encipher the

plaintext, or clear messages, into unintelligible cypher text or cryptograms using a key. A

deciphering algorithm is used to decrypt or restore the original

information. Cyphers are cryptographic algorithms [18].

Figure [3.1] Cryptography

Cryptography is used for many goals achieved simultaneously in one application or only one [17]. These goals are:

1. Confidentiality: it ensures that nobody can understand the received message except the

One who has the decipher KeyKey.

1. Authentication: it is the process of proving the identity, meaning that the user or the

the system can prove their own identities to other parties who don't have personal

knowledge of their identities.

1. Data Integrity: it ensures that the received message has not been changed in any way from its original form.
2. Non-Repudiation: it is a mechanism used to prove that the sender sent this message,

And the message was received by the specified party, so the recipient cannot claim that they did not send the message.

1. Access Control: it is the process of preventing unauthorized use of resources. This

goal controls who can have access to the resources, if one can access, under which

restrictions and conditions the KeyKey can occur, and what is the permission level

of given access [17].

Cryptosystems have two types based on the number of keys used; they are either

symmetric (private key encryption), in which case both the enciphering and deciphering keys

must be kept secret or asymmetric (public-key encryption), in which case one of the keys can

be made public without compromising the other [18] [12] [16].

3.2 Asymmetric encryption (public-key encryption)

Each person has a pair of keys (a public key and a private key). A person's public KeyKey is

published, but the private KeyKey is kept secret. Messages are encrypted using the recipient's

general KeyKey and can only be decrypted using his private KeyKey. In this method, the sender and

the receiver eliminate the need to share keys (secret information). All communications

use only public keys; no private key is transmitted [10]; see figure [3.2].

Figure [3.2]: A simplified model for asymmetric encryption.

3.2.1 Rivest–Shamir–Adleman Algorithm (RSA)

RSA implements a public-key cryptosystem, as well as digital signatures.

1. Public-key encryption: In RSA, encryption keys are public; during the decryption

Keys are not; only the person with the correct decryption key can decipher an encrypted message. Everyone has their encryption and decryption keys.

1. Digital signatures: The receiver may need to verify that a transmitted message

Originated from the sender (signature) and didn't just come from there

(authentication), done using the sender's decryption key, and anyone can verify the signature using the corresponding public encryption key [23].

The security of the RSA algorithm has been validated since no known attempts to

break, it has yet to be successful, primarily due to the difficulty of factoring in large

product of two prime numbers [23].

RSA operation involves four steps key Generation, key distribution, encryption and

decryption. The main idea is to find large positive integers ?, ? and ? such that with

modular exponentiation for all ? as equation 3.1:

(?17?)? ??? ? = ?

Where encryption Key ?, decryption key ? and ? is a composite number which is a product of

two large prime numbers. Although knowing of **e** and ? or even ?**,** it is challenging to find ?**.** [24].

Key Generation: key Generation can be done using the following steps [24]:

First, we select two large prime numbers of about the same size, ? ??? ?, then compute n and

j(n) as equations 3.2, 3.3:

? = ? ∗ ?

(3.2)

? (?) = (? − ?) (? − ?)

(3.3)

After that select encryption key ?, ? < ? < ?(?), based on relationship specified on

equation 3.4:

??? (?, ?)) = ?

(3.4) Then calculate the Private KeyKey using equation 3.5:

? = ?18−?(??? ?(?))

(3.5)

Finally, the public KeyKey can be computed using n and e. Also, private KeyKey is equal to d; RSA is

used for the encryption/decryption process as follows:

Encryption: message ? at the senders' side is converted into an integer ? where ? < ? and

??? (?, ?) = ?**.**

The cypher text ? is calculated by using the receiver public KeyKey ?, which is sent to the receiver. as

equation 3.6:

? = ?? ??? ?

(3.6) Decryption: At the receiver's end ? is deciphered from ? by using the receiver's private KeyKey ? by

computing equation 3.7 [24]:

??=(??)?= ? ??? ?

(3.7)

3.2.2 Diffie –Helman Algorithm (DH)

The Diffie–Hellman is a specific method of exchanging cryptographic keys. The critical

exchange method allows two parties without prior knowledge of each other to jointly

establish a shared secret key over an insecure communications channel. The KeyKey can then encrypt subsequent communications using a symmetric cypher [25]. the

The following steps show how the Diffie-Hellman Algorithm Key exchanges work [26]:

1. Compute global public elements: in this step, choose ? as the prime number, and ? is

The primitive root of ?, such that ? < ?.

1. User **A** KeyKey generation: this can be done by selecting private KeyKey ?

<

?, then calculate public KeyKey ?

, as equation 3.8:

19 ???= ? ∗ ???, such that ?

??? ?

(3.8)

Key Generation for user B: this can be done by selecting private KeyKey ?

, such that

??< ?, then calculate public KeyKey ?

, as equation 3.2:

???= ? ∗ ??? ??? ?

(3.9)

Calculation of secret Key by user A: Private Key of user **A** can be computed as an equation

3.10:

? = (??)?? ??? ?

(3.10) Calculation of secret Key by user B: Private Key of user **B** can be computed as an equation

3.11:

? = (??)? ??? ?

(3.11) Notice that the result is that the two sides have exchanged a secret key value**.**

? 3.2.3 Elliptic Curve Cryptography algorithm (ECC)

ECC is an alternative mechanism for implementing public-key cryptography. ECC is based on discrete logarithms that are much more difficult to challenge at equivalent key lengths. The security of a public key system using elliptic curves is based on the difficulty of

computing discrete algorithms in the group of points on an elliptic curve defined over a finite field. Elliptic curve equation over a finite field ?

can be described by equation

[3.12] [21]:

20? ??= ??+ ?? + ?

(3.12)

Here, y, x, a and b are all within ?

, and p is an integer modulo p. a and b is the

coefficients that determine the points on the curve. Curve coefficients have to

fulfil one condition, that is:

????+ ????≠ ?

This condition guarantees that the curve will not contain any singularities [21].

Each value of a and b gives a different elliptic curve. The public KeyKey is a point on the

turn, and the private KeyKey is a random number. The general KeyKey is obtained by multiplying

the private KeyKey with a generator point G in the curve [22].

3.3 Symmetric encryption (private key encryption)

Using the same secret KeyKey encrypts and decrypts messages. Its problem is transmitting the secret KeyKey to a legitimate person that needs it [10].

Figure [3.3]: A simplified model for symmetric encryption.

Advanced Encryption Standard (AES)

AES is based on a substitution permutation network. It is fast in both hardware and Software. AES has affixed block sizes of 128, 192, and 256 bits, and it can specify block and critical dimensions in any multiple 32 bits. The block size has a maximum of 256 bits.

AES algorithms have many characteristics: resistance against all known attacks, speed and code compactness on a wide range of platforms and design simplicity [26].

AES operate on a 4 × 4 matrix of bytes, termed a state. The AES cypher is specified as several repetitions of transformation rounds that convert the plaintext into ciphertext.

Each game consists of several processing steps, including one that depends on the encryption key. A set of reverse bands are applied to transform ciphertext back into original plaintext using the same encryption key [27]. The encryption algorithm is organized into three games. Round 0 is simply an add critical round; round 1 is a full round of four functions, and round 2 contains only three parts. Each game includes the additional essential function, which uses 16 bits of KeyKey. The initial 16-bit KeyKey is expanded to 48 bits, so each round uses a distinct 16-bit round key [26]. Figure 3.4 represent the AES encryption process and the relationship between several round and cypher key sizes, where ten cycles need a key length of 128 bits, 128 cycles supports a key length of 192 bit, and 14 processes need a key length of 256 bit.

Figure 3.4: AES general encryption process [27]

The encryption algorithm involves the use of four different functions or transformations:

Add KeyKey, nibble substitution, shift row, and mix column, shown in Figure 3.5.

Figure 3.5: AES structure encryption process [26].

- Standard Ciphertext algorithms
- Key management algorithms

Rivest Shamir Adleman algorithms(RSA) / etc /DH

- Attack Analysis
- Summary

Chapter Three: Models of obfuscation

- Introduction
- Obfuscation techniques

- Reverse engineering
- Summary

Chapter 4: Proposed Model: DOSP

Chapter 5: Results and Analysis

Chapter 6: Conclusion and future work

Reference

We provide online educational paper writing services all over the world at the best price.Buy assignment today and get high-quality content. This is a sample of a paper related to computer science. Please consider me if you want to read more about computer science.Visit this articles on computer science assignment.