



جامعة عمّان العربية للدراسات العليا

Web Application Security of E - Money Transfer Systems

أمنية تطبيق الويب لأنظمة التحويل المالي الألكترونية

BY

Mohammed Saad Abdul Wahhab (200910213)

SUPERVISOR

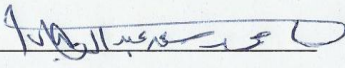
Prof. Dr. Alaa Al-Hamami

A Thesis Submitted In Partial Fulfillment of the
Requirements for The
Degree of Master of Science In Computer Science.

Department of Computer Science
College of Computer Science and Informatics
Amman Arab University
September, 2011
Amman Jordan

AUTHORIZATION OF THESIS

I, the Undersigned "Mohammed Saad Abdul Wahhab" authorize hereby "Amman Arab University" to provide copies of this thesis to libraries, institutions, agencies, and other parties upon their request.

Signature 

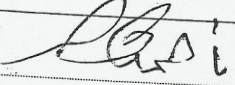

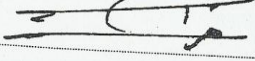
© Copyright By: Amman Arab University of Graduate Studies (AAU).

II

APPROVAL

Name: Mohammed Saad Abdul Wahhab.
Degree: Master of Computer Science.
Title of Thesis: Web Application Security of E-Money Transfer Systems.

Examining Committee:

اعضاء لجنة المناقشة	
التوقيع والتاريخ	الإسم الثلاثي
	رئيساً أ. د. أحمد عودة الجابر
	عضواً ومشرفاً أ. د. علاء الدين الحامد
	عضواً د. فراس المشاقبة
	عضواً
	عضواً ومشرفاً مشاركاً

DEDICATION

"I dedicate this thesis to all those who assisted me in my application for my Masters Degree. Amongst those are my late father, the martyr, Dr. Saad Abdul Wahhab Al Shaaban and all my family members including my mother and sisters as well as my relatives. I also dedicate it to Dr. Alaa H Al Hamami and to all my friends"

Acknowledgment

First of all, my prayers and thanks go to Allah, who has helped me complete this thesis. I would like to thank my supervisor Prof. Alaa Hussein Al-Hamami for his support, patience and good direction that guided me towards a high quality methodology.

I would like to thank the examining committee, all my colleagues and relatives for supporting me from the beginning. Many thanks go to all the lecturers and the administrative staff of Amman Arab University.

I would thank my mother for her prayers and encouragement, thanks I would also extend my thanks to my sisters and all my friends.

Table of Content

acknowledgment	V
table of content.....	VI
list of figures	X
list of abbreviations	XII
abstract.....	XIV
arabic summary	XVI
chapter one introduction.....	1
1.1. background.....	1
1.1.1 the need for web application securities	1
1.1.2 cyber security causes	2
1.1.3 web application security (was)	4
1.1.4 web application security roadmap objectives are building a foundation by doing the following [13]:.....	7
1.1.5 web application security threats on money-transfer systems	8
1.1.6 online-banking two-factor authentication schemes.....	8
1.1.7 on-line money transfers' attacks types	9
1.1.8 silentbanker: introduction	10
1.1.8.1 silentbanker attack methodology of the two-factor authentication schemes	11
1.1.8.2 silentbanker trojan analysis.....	12
1.1.8.3 silentbanker characteristics.....	13
1.1.8.4 silentbanker infection and installation method.....	13
1.1.8.5 silentbanker tools selection.....	14
1.1.9 securing the money-transferring process	15
1.1.9.1 security problem analysis.....	15
1.1.9.2 the society for worldwide interbank financial telecommunications (swift)	16
1.1.9.3 encryption and decryption of communication	17
1.1.9.4 secure sockets layer (ssl)	18
1.1.9.8 security control.....	18
1.1.9.9 public key infrastructure (pki) and digital signatures...21	
1.1.9.10 the secure socket layer and secure channels (ssl)...21	
1.1.10 the need for improving web applications' security and for a performance assessment	22
1.2 statement of the problem	23
1.3 aims and objectives	27

1.4 significance of the study and expected learning outcomes.....	28
1.5 php web development programming language.....	28
1.5.1 what is php?	29
1.5.2 why php are gaining more market share?.....	29
1.5.3 php advantages.....	30
1.6 system engineering.....	31
1.7 thesis organization.....	31
Chapter two Literature reviews	32
2.1 public fears about strategic cyber breakthroughs	32
2.2 on-line banking threats supporting the public fears and appealing worldwide attention	33
2.3 silentbanker malware threat as one of most recent dangerous e-banking threat.....	33
2.3.1 SilentBanker Motivation	35
2.3.2 SilentBanker Infection Symptoms	36
2.3.3 SilentBanker Infection Method and Tools.....	36
2.3.4 Typical Client-Side web SilentBanker Attack Process...37	
2.3.5 Anti-Virus Companies are detecting SilentBanker Trojan Threat.....	38
2.3.5.1 Client-Side Solutions	38
2.3.5.2 SpyBlock Technique.....	39
2.3.5.3 Cryptographic Technique.....	40
2.3.5.4 Entrust Solutions.....	41
2.3.5.5 The Manual Removal for the SilentBanker.....	42
2.3.6 Web Application Security (WAS).....	42
2.3.6.1 Web Applications Development Methodologies	44
2.3.6.2 Creating a Security Framework for Web Applications (WA)	45
2.3.6.3 General WAS Developer Mistakes.....	46
2.3.6.4 Measuring Decision-Makers' Awareness about the Importance of WAS in Companies.....	47
2.3.6.5 Recommendations for Improving the Existing Web Applications Securities Assessment	49
2.3.7 Inter-Bank Electronic Fund Transfer Systems (EFTs) for SWIFT Coding.....	50
2.3.8 E-Commerce Security Mechanisms Transactions.....	53
2.3.8.1 User ID, Passwords, and Tokens.....	53
2.3.8.2 Public Key Infrastructure (PKI), Digital Signature and Digital Certificates.....	54
2.3.8.3 Secure Socket layer (SSL) and Securing Channels...55	
2.3.8.4 Securing Payments.....	56
2.4 software security design and testing	56

2.5 making strong encryption easy to use.....	58
chapter three methodology of work.....	59
3.1 Introduction.....	59
3.2 System Security Engineering Process Model	59
3.2.1 Phase (0): Developing the Suitable Security System Model for Money Transferring System Web Application	61
3.2.1.1 Security gaps and weaknesses Analysis	61
3.2.1.2 Phase (0) Results	62
3.2.2 Phase (1): SilentBanker Threat Analysis.....	63
3.2.2.1 SilentBanker Infection and Installation Mechanism Analysis.....	64
3.2.2.2 Phase (1) Results:	64
3.2.2.3 Phase (1) Result shows	65
3.2.3 Phase (2): Developing the Suitable Security System Model for Money Transfer Web Applications	66
3.2.3.1 Phase (2) Results	66
3.2.4 Evaluating the Designated Security Software	66
3.2.5 The last step before real implementation is described in figure	67
3.2.6 Other phases	67
3.3 Security Permissions Access Coding.....	68
3.3.1 Create a Custom Encryption Permission	69
3.3.2 Joining Public and Private Keys using Symmetric Keys.....	69
3.3.3 Making Strong Encryption/Decryption.....	69
3.4 Using the RSA for Joining Up Public/Private Keys.....	70
3.5 Conclusion.....	72
chapter four the proposed solution.....	73
4.1 Introduction.....	73
4.2 Results	74
Chapter five Conclusion	88
5.1 Conclusion:.....	88
5.2 Future Work.....	88
References.....	90
Appendixes	98

CHAPTER THREE: METHODOLOGY OF WORK

3.1 Introduction	57
3.2 System Security Engineering Process Model	57
3.2.1 Phase (0): Developing the Suitable Security System Model for Money Transferring System Web Application	
3.2.2 Phase (1): SilentBanker Threat Analysis	
3.2.3 Phase (2): Developing the Suitable Security System Model for Money Transfer Web Applications	
3.2.4 Evaluating the Designated Security Software	
3.2.5 The Theoretical Evaluation or “security assessment”, and Testing Implementation phase	
3.2.6 Other (3, 4, and 5) phases	
3.3 Security Permissions Access Coding	65
3.3.1 Create a Custom Encryption Permission	
3.3.2 Joining Public and Private Keys using Symmetric Keys	
3.3.3 Making Strong Encryption/Decryption	67
3.4 Using the RSA for Joining Up Public/Private Keys	68
3.5 Conclusion	

CHAPTER FOUR: THE PROPOSED SOLUTION

4.1 Introduction	70
4.2 Results	71
4.2.1 On the Client-Side	
4.2.2 In Case of No Penetration	
4.2.3 In Case of Any Penetration	
4.2.4 The proof for the Lock Status	

CHAPTER FIVE: CONCLUSIONS AND FUTURE WORK

5.1 Conclusions	86
5.2 Future Work	86
<u>REFERENCES</u>	89

LIST OF FIGURES

Figure 1: A Typical Web Based Application Life cycle	5
Figure 2: SSL Encrypted end-to-end	6
Figure 3: Two-Factor Authentications Schemes that are used in Online Banking	9
Figure 4: Threats Types and its emerged year	10
Figure 5: An Example on Banking Attack Process (OWASP)	12
Figure 6: WAS location	14
Figure 7: A Sample SWIFT Payment Message	16
Figure 8: Logging before and after HTML Injection.	22
Figure 9: Real HTML Injection code of SilentBanker Trojan	22
Figure 10: Man in the Browser (MitB)	23
Figure 11: Man in the Browser Bank Transaction	24
Figure 12: ICDM Phases.	43
Figure 13: A Method of Funds Transfer to Conduct Payment	49
Figure 14: Modified SWIFT Network	49
Figure 15: System Security Engineering Process Model	58
Figure 16: "Security System Analysis"	60
Figure 17: "Attack Tree with Probability and Impact Estimations"	61
Figure 18: "Bell-LaPadula"	63
Figure 19: The Software Evaluation and Testing V-model	64
Figure 20: "High-Level Model of Security Testing and Evaluation"	65
Figure 21: "System Security Code Access Permission"	66
Figure 22: File Encryption.	67
Figure 23: File Decryption	67
Figure 24: The Three Proposed Solution Phases.	70
Figure 25: The Entry Banking Screenshot.	71
Figure 26: Bank Authentication Scheme.	72
Figure 27: The Client Enters The Credential Full Information In Order To Complete The Transferring Process.	73
Figure 28: Bank Decryption and Verification processes.	74
Figure 29: Bank screenshot presents a Successful Transferring Process	75 76
Figure 30: Bank Web Application Security System showed a penetration for its web security presented as a failure login.	77
Figure 31: The Encryption and Decryption process held by the Bank WAS proved the penetration process.	78
Figure 32: The Banking System adds a new client account for its clients' Data base	79 80
Figure 33: The successful adding for the new client	
Figure 34: Bank Screen Shot Shows that the Addition Process is Successfully Completed	80 81
Figure 35: The Addition Process execution through the Bank Server	82 83
Figure 36: The Bank Money-Tranfaring Process	84
Figure 37: The Lock Status Proof	

Figure 38: "Try Again Later"

Figure 39: The Penetration Status was detected from the Bank
WAS and a message was sent showing it.

LIST OF ABBREVIATIONS

AES	Advanced Encryption Standard.
ATM	Automated Teller Machine.
BHO	Browser Helper Object.
CA	Certificate Authorities.
CAPTCHA	Completely Automated Public Turing test to tell Computers and Humans Apart.
CSMC	Cyber Security Management Center.
DLLs	Dynamic Link Libraries.
EFTs	Electronic Fund Transfer Systems.
EV SSL	Extended Validation Secure Sockets Layer.
FAA	Federal Aviation Administration.
FBI	Federal Bureau of Investigation.
GCI	Common Gateway Interface.
HTML	Hyper Text Markup Language.
HTTP	Hyper Text Transfer Protocol.
ICDM	Internet Commerce Development Methodology.
ID	Identification.
IDS	Intrusion-Detection-System.
IE	Internet Explorer.
IP	Internet Protocol.
ISP	Internet Service Provider.
IT	Information Technology.
KPI	Key Performance Indicator.
LAN	Local Area Network.
L2TP	Layer Two Tunneling Protocol.
MAC	Media Access Control.
MitB	Man-in-the-Browser.
MitM	Man-in-the-Middle.
OCR	Optical Character Recognition.
ODBC	Open Database Connection standard.
OIG	Office of the Inspector General.
OS	Operating System.
OTP	One-Time Password.
OWASP	Open Web Application Project.
PC	Personal computers.
PIN	Personal Identification Number.
PKI	Public Key Infrastructure.
PP	Protection Profile.
PPTP	Point-to-Point Tunneling Protocol.
PWSec	Process for Web Services Security.
QFD	Quality Function Deployment.
RSA	Rivest, Shamir and Adleman.
SA	Structured Analysis.
SD	System Design.
SM	Security Models.
SMS	Short Message Service.

SOA	Service-Oriented Architecture.
SPA	Secure Payment Application.
SSL	Securing Sockets Layers.
ST	Security Target.
TA	Threat Analysis.
TEA	Tiny Encryption Algorithm.
UDP	Universal Dialing Plan.
URL	Uniform Resource Locator.
ST	Security Target.
SWIFT	Society for Worldwide Interbank Financial Telecommunications.
TCB	Trusted Computing Base.
TCP/IP	Transmission Control Protocol /Internet Protocol.
TLS	Transport Layer Security
VPN(s)	Virtual Private Network(s).
WA(s)	Web Application(s).
WAS(s)	Web Application Security(s).
WS	Web Service.
WS-BTS	Web Service-based of e-Bank Transfer System.
WSecArch	Web Service Security Architecture.
WSecReq	Web Services Security Requirements.
XML	Extensible Markup Language.
3D	Three Dimensional.

Web Application Security of e – money transfer system

By

Mohammed Saad Abdul Wahhab

SUPERVISOR

Prof. Dr. Alaa Al-Hamami

Abstract

Information security science started to play a vital role in our life and became an important issue that used for judging on any system about either its success or failure. E-banking applications for transferring money is considered as one of the most important applications that banks nowadays are taking care of maintaining its process validity and accuracy as a necessity for the health of the transferring process to transfer the correct amount to the right receiver.

E-money transferring process can be attacked by hackers through using different malwares and viruses for changing the transferee information and the transferring amount, one of them is called "Silent Banker"; it is considered one of the most important threat that appealed global senior banks' concern around the world because of its high capability into penetrate the most powerful security banking systems and the ability to use different tools to do so, which cost banks large and painful losses.

This thesis was interested in proposing a solution to the "SilentBanker" problem through blocking possible security vulnerabilities that SilentBanker can penetrate the security system

through. Two suggestions were recommended for working together side by side, namely:

1. Use the "Lock" of the browser in order to prevent any interference during the conversion process.
2. The use of the "electronic Signature" by banks for verifying the sender identity before implementing the money transferring process. Using the "RSA" encryption method through the "PHP Web Applications Programming Language" for websites programming was chose among all other different languages because of its dealing with servers not with user and thereby ensure that more security than other programming languages can provides. The necessary tests were done performed this thesis to prove the validity of the proposed solutions; where prevented from entry unauthorized during process of transferring funds by the program during the test As well as the program is designed for real-time so can be deployed, or trading in this security program to global markets .

أمنية تطبيق الويب لأنظمة التحويل المالي الألكترونية

Arabic Summary

الملخص

بدأت الامنية تلعب دوراً مهماً في حياتنا واصبحت من المواضيع الهامة والتي تحدد نجاح الانظمة وفشلها. ومن ضمن التطبيقات المهمة هي التطبيقات المصرفية وخاصة موضوع النقل المالي حيث تصبح المحافظة على نقل المعلومات مهمة جداً ويجب ان تكون صحيحة لكي يتم نقل المبلغ المطلوب من المصرف الى الغاية المطلوبة.

تتعرض عملية نقل البيانات الى تدخل الفايروسات لكي تغير المعلومات الخاصة بجهة المحول لها والمبلغ المحول حيث يمكن التلاعب بقيمته . ومن اهم الفايروسات التي تؤثر على عملية نقل المبلغ هو سايلنت بانكير اتروجنز والذي يكون هدفه في تغير موضوعين هما المبلغ المحول والجهة المحول لها.

تهتم هذه الرسالة بوضع معالجات لهذه المشكلة من خلال غلق الوهن الامني المحتمل الذي يستطيع السالينت بانكر من اختراق النظام الامني والتي تمت عن طريقين هما:

1. حجز الوسط البيني (اللوك) لمنع اي تدخل اثناء عملية التحويل.
2. باستخدام التوقيع الرقمي للتحقق من شخصية المرسل من قبل البنك قبل تنفيذ عملية التحويل.

تم استخدام طريقة التشفير (RSA) في عملية التشفير وكذلك استخدام (PHP) كلغة برمجة الموقع من بين لغات متعددة وذلك بسبب تعاملها مع المزود وليس مع المستخدم وبالتالي توفر امان اكثر من لغات البرمجة الاخرى وتم اجراء تجارب لأ ثبات صحة الحلول المقترحة; حيث منعت من الدخول الغير مخول اثناء عملية تحويل الاموال من قبل البرنامج اثناء الاختبار وكذلك تم تصميم البرنامج على الوقت الحقيقي حتى يمكن نشر او تداول هذا البرنامج الامني في الاسواق العالمية.

Chapter One

Introduction

1.1. Background

1.1.1 The Need for Web Application Securities

The need for web application securities information can be considered as an organization's most valuable asset guard where attacks on it could threaten the organization's integrity, availability and confidentiality. Web applications security is considered nowadays as the key business issue and the biggest threat for organizations that concerns intellectual property ,the critical client data and trade secrets, and such interest resulted from loss of confidential data in the first place and the loss of company's reputation in the second one place, in addition to the financial penalties obligations, and Lack of awareness of any Web application security breaches which can cause huge damages to the company's reputation and brand image and as a result on the whole business itself [1].

Considering any Internet link as including a risk in while understanding the reality of spreading about 87% of threat through the internet connection [2] realizes the need for securing any web application for the following reasons:

- The Customer behaviour itself neither considers as a potential threat that it can't be predicted nor relies upon [2].
- The Operating Systems for Personal Computers (PC) are open for any threat and is not secured in a proper way because of an

- unlimited abroad network connection that can't guarantee its reliability and authentication integrity that can't be guaranteed by any enterprise [3].

1.1.2 Cyber Security Causes

Considering cyber security threat as a major reason for explaining the need for Web Application Securities; knowing that Cyber security nowadays is causing in tremendous damages among both public and private businesses' sectors and still evolving. Hackers attacked the public-facing of US Federal Aviation Administration (FAA) Web applications by gaining unauthorized access on 48,000 of both its employees and Air Traffic Control System [4]. More within e-commerce of financial systems; large companies have Faced Bankruptcy from E-Banking Loss Fraud [5]. One important is Trojan SilentBanker which our research will cover. Its importance came from –not being the first e-Banking Trojan threat – but as a very serious one with an emerged worrying wide scale by targeting of over 400 banks domain names of large American banks and other banks worldwide and results in a huge Bankruptcy and financial loses [5]. It still dangerous because of a special features to avoid two-factor authentication, the capability to perform both Man-in-the-Middle (MitM) attacks and Man-in-the-Browser (MitB) attack [6], its different attributes that give it the ability to work without users detecting [6] as capturing related information account before encrypting and then resending it to the attacker database, and the back-door potential capability that provides the attackers the ability to install malicious injecting coding and screenshots on computers.

As Trojans do not proliferate, they permit attackers to do attacks without drawing thought to them and as much as being undiscovered the more opportunity it has to concession computers to protect against it the better chance for causing a complete computer crash [3], addition to, the ability of updating itself, as it relays Uniform Resource Locators (URLs) and the banking websites Hyper Text Markup Language (HTML) to the attackers as a daily task [7]. Finally; its keyloggers capability of cookie stealing for sensitive login data like credit card, passwords, and bank account information through changing client's registry settings and other important windows system files as much time it spends undiscovered ,so it is very important to remove Trojan [3]. As banks try to develop their detection and fighting fraud, Trojan horses also are developing their attacking techniques continuously in more sophisticated way. Out from Security Engineering Perspectives; web threats are taking place because of the following:

- (1) Insufficient configuration for Web applications for preventing any unauthorized access.
- (2) The well-known weaknesses of ready-available Web applications' software's didn't influenced by any correction but only installed a "readily available security software patches" that was released by software vendors into public use.
- (3) An incomplete knowledge for organizations about understanding firewalls and related security controls of applications' layers and vulnerability scanning prevented an accurate risk assessment to hold on.

Those reasons show Decision-makers responsibility for Web applications Security of having a proactive consideration for such security issues and potential cyber attacks awareness; detecting, reporting and responding to security incidents procedures in a timely-manner when it happened, and to set related-full-time Jobs for these responsibilities. They also need to re-think again in Cost Trade-offs planning to try to provide a less-expensive alternative. A recent FBI study estimated losses resulted from both business and consumer accounts have exceeded \$100 million resulted in a huge financial losses [8].

1.1.3 Web Application Security (WAS)

A Web Application or Web Service is a “software application that is accessible using a web browser or HTTP(s) user agent” [9]. **Web services** (WS) recently was emerged in business and technology for the purpose of implementing Service-Oriented Architecture (SOA) for information base-systems [1] due to the huge instability in technology and the need for a “mission-critical software-intensive systems” where the traditional bottom-up approach for is lacking the development process that should construct a general framework for security as an integration process among all the presented “WS-based software development life-cycle” stages that shown in Figure 1 [6].

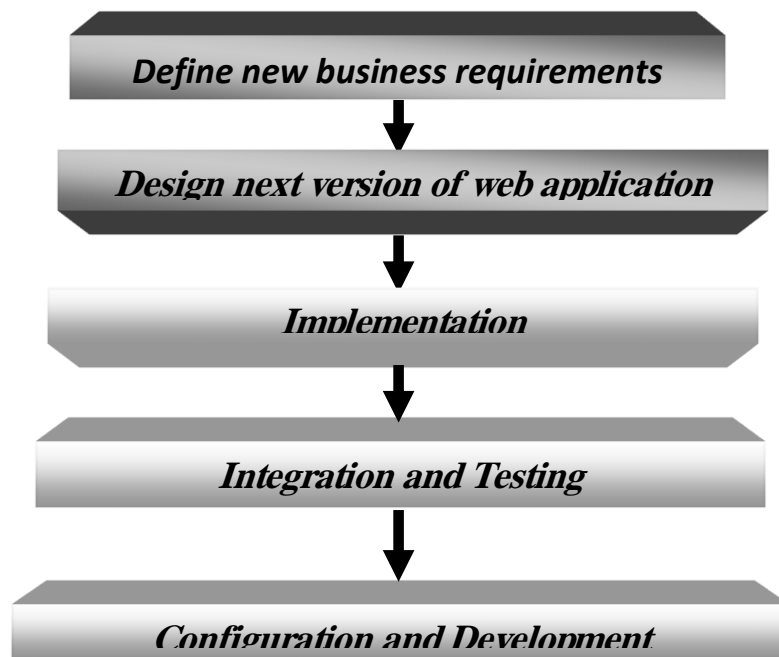


Figure 1: A Typical Web Based Application Life Cycle [6].

PWSSec (Process for Web Services Security) science emerged for guiding the WS-based security systems development for facilitating the integration process mechanism once the system of WS functional architecture completed or the architecture has been defined completely the pattern-oriented security architecture under a certain technical standards that should overcome both vulnerabilities and threats [10].

Web Application Security (WAS) is the automatic securing process for the HTML web applications pages when to browse by any server by automatically “extracting all of the acceptable responses defined in the HTML page, and enforcing HTTP requests to conform to the automatically generated security; locating between firewalls and load where it functions like a proxy for bi-directional information flow of requests and responses” [11], SSL can provide security for the encrypted end-to-end as shown in Figure 2.

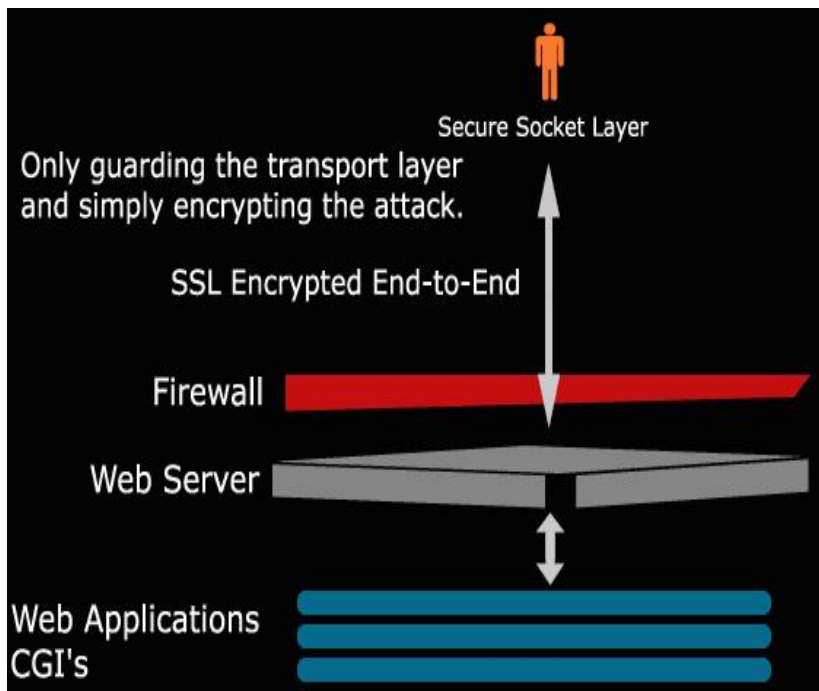


Figure 2: SSL Encrypted end-to-end [9].

Web Services-based systems are to finalizing the whole of secure system development sub processes stages of the following [12]:

(1) Risk-based security engineering called WSSecReq (Web Services Security Requirements) sub process which applies an application-centered approach that identifies the security threats sets inside any system for the purpose of defining its security requirements for preventing those threats and refining and arranging them into a tree-like structure that will shape the pattern-oriented security architecture for identifying the attack scenarios in security profiles and its impact by applying an “ISO-compliant 15408 risk analysis methodology”; standard-centered security design will be identify after then where security reusable-based approach requirements are then identified.

(2) WSSecArch (Web Service Security Architecture) sub processes that allocate particular security requirements inside the WSSecReq subprocess in order to shape the mechanisms of the “WS-based security” architecture.

(3) WSSec- Tech (or called as the Web Services Security Technologies) subprocess can identify the standards sets for WS-based security that can employ the WSSecArch subprocess.

1.1.4 Web Application Security Roadmap Objectives are Building a foundation by doing the following [13]:

a. Identifying Web Application vulnerabilities can lessen the internet risk and chunking publicity from web application vulnerabilities and so can save time through.

b. Concentrate on Web Application vulnerabilities “Web Application Firewall CapEx and deployment times; 30 days to evaluate each vendor if conducting a bake-off; 4-8 weeks to deploy chosen tool after the evaluation phase; Ongoing management and fine-tuning can be expected after deployment; CapEx for Web Application Firewalls will vary between vendors. Expect approx. 25K-40K per appliance and you will need at least two for redundancy; and Budget for 75K-100K”.

c. Monitor the assessment tools for Web Application security for diagnosing any need for code level changes; providing a static analysis for source coding; and so make the necessary mapping for web application vulnerabilities for tracking its Authentication and Authorization; Client-side Attacks; Logical Attacks; Command Execution and Information Disclosure and choose the most suitable WAS assessment tool for the desired threat classification framework and scoring model.

d. Building up the necessary Response plan for Web Application Incident for preventing having the undesired WAS blind-sided that can provides effective pre and reactions that can introduce trust and confidence on threat prevention ability.

1.1.5 Web Application Security Threats on Money-Transfer Systems

A clear definition for Malware was introduced as “A malicious piece of code, which is made intentionally to damage our system or interrupt the normal computing environment by trying to inject or re-install itself for binary executable considerations on the basis of certain features [1]”. For getting a remote access control to information system, record and forward data from the controlled system to a third party without the user’s authorization or knowledge. Information systems which have been compromised are able to perform illegal security actions which can harm the information system, or otherwise affect the data and system integrity [8]. Malwares nowadays are targeting money transfer systems (like e-transfer); both formal (like e-banking) and informal ones silently through web applications through developing their distinctive capabilities to beat the most complicated security safeguard for any bank security in either a static or dynamic behaviour and shaped into different types of worms, viruses, spyware and Trojan horses that differ in both functionalities and activities [14].

1.1.6 Online-Banking Two-Factor Authentication Schemes

Online-Banking Authentication Schemes currently are classifying as

Two-Factor Authentication that require proof of two to allow a business deal to take place and most of these schemes can be beaten most recent Trojans [6]. Figure 3 shows the Two-Factor Authentication Schemes that are used in Online-Banking.

Scheme	Technique	Advantages	Disadvantages
Something you are	Keystroke dynamics	<ul style="list-style-type: none"> No extra hardware needed High user's acceptability 	<ul style="list-style-type: none"> Not reliable. High enrollment time
Something you know	Transfer password	Simple to use	Can be easily captured as login credentials.
Something you have	Matrix card	Cheap and relatively simple to use	<ul style="list-style-type: none"> Can be lost or forgotten. Difficult to share
	Hardware tokens	Difficult to be tampered	<ul style="list-style-type: none"> Expensive Can be lost or forgotten Difficult to share
	Mobile phone	No extra hardware needed.	<ul style="list-style-type: none"> No coverage or run out of battery Cost for bank

Figure 3: "The Used Two-Factor Authentications Schemes in Online Banking" [6].

1.1.7 On-line Money Transfers' Attacks Types

On-line Money Transfers' Attacks Types have different types are developed nowadays into a more authoritative and competent one that makes defeating them so hard and needs for a complex authentication protocols like the two-factor schemes ones.

- **Social Engineering** that attacks communications' means [6].
- **Phishing** emerged in 1996 as a most technically complex well thought-out of e-commerce fraud crimes [3].
- **Pharming** or sometimes called as DNS poisoning who attack users (or customers) by placing copies inside either bank web pages or user browser and recognizing a digital authority certification [3].

- **Trojans** that used in cyber attack, there're different types of Trojans; Keystroke logging, HTML Injection, and Screenshots and Mouse Event capture. This thesis interest is about one of Trojan horses' type of **SilentBanker** as a part of a cyber attack system for money-making that uses the HTML Injection coding as a tool for executing attacks using Man in the Browser (MitB) techniques where Fraud detection systems has a weak position that infection happened on user' own computers across their own Internet connection. Banks can't detect those real time attacks were mutual authentication and tokens can be avoided and fraudster has full control over the customer's computer [6]. Figure 4 shows Threats Types.

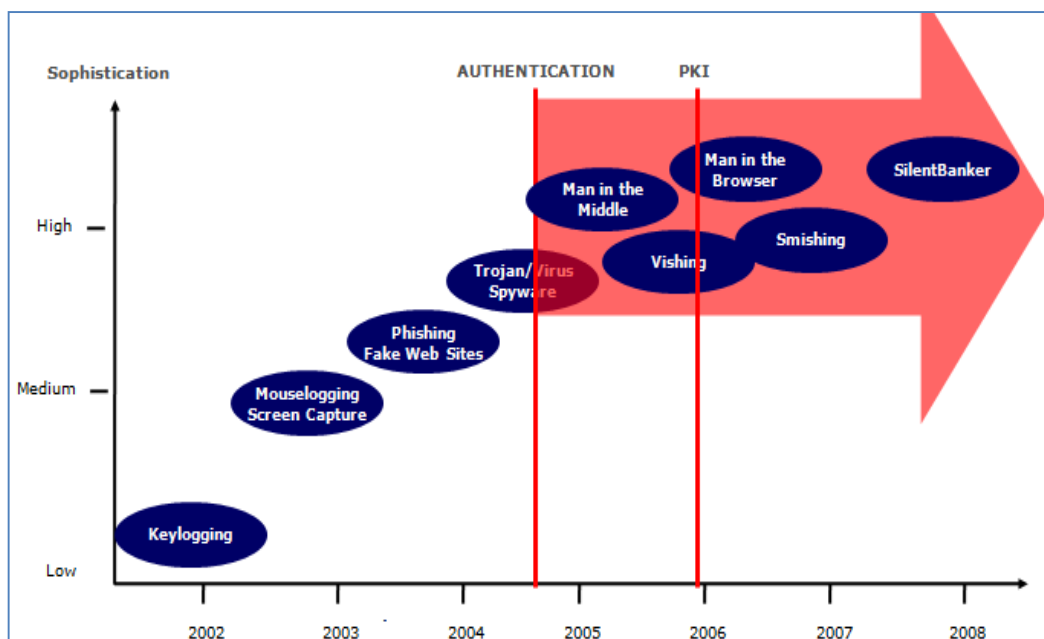


Figure 4: Threats Types and its Emerged Year [34].

1.1.8 SilentBanker: Introduction

SilentBanker as known as one of Trojan Malware threats (Also Known As: Spy-Agent.cm as McAfee definition) has recently appealed attention since it's discovered (Semantic, 2007). Many

articles, web journal and anti-viruses websites have recently written about the reasons of why such threats for the web applications security are considered as a key business issue nowadays. Approved by public thread fears due to huge losses both financially and non-financial ones represented by data loss and un-authorized access were reported by the Federal Aviation Administration FAA in 2009 as a "Review of Web Applications Security and Intrusion Detection in Air Traffic Control Systems" review [4]. More focuses on the e-commerce sector of e-banking threats losses was held by krebs in his articles first through the Washington Post journal such as his article "FBI: Cyber Crooks Stole \$40MM from US Small, mid-sized Firms" [8], next through "N.Y. Firm Faces Bankruptcy from \$164,000 E-Banking Loss" article in KrepsOnSecurity Journal in 2010 [5].

1.1.8.1 SilentBanker Attack Methodology of the Two-Factor Authentication Schemes

By targeting over 400 of biggest American banks; SilentBanker clearly stated a big threat by having the ability to get around the two-factor authentication by interrupting transactions using man-in-the-middle attacks; silently alter the user destination bank account details to the instead attacker's account details while user cant notice anything incorrect within this transaction where all of those traffic are intercepts by Trojan before the encryption process in transaction are taking place over SSL [3].

SilentBanker defeating the two-factor authentication scheme by attacking the two-factor schemes as the following process steps:

- 1) SilentBanker infection happen when on-line bank user are using a two-factor authentication scheme at login-time or when to request a transfer, the user should give the bank password, SilentBanker One-Time Password (OTP) will catch the most sensitive data of login password, account number and of course the user's name.
- 2) Now it's the Trojan turn in changing silently the account number destination then allows the transfer process to keep on while changing the transferred amount but show the client the correct account number in order not to feel about something wrong are happening. Figure 5 shows the SilentBanker Attack Process.

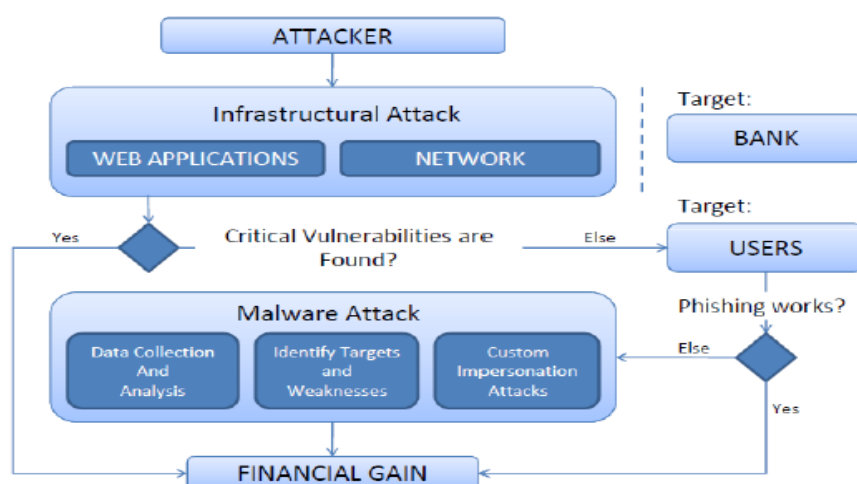


Figure 5: An Example on Banking Attack Process (OWASP) [6].

1.1.8.2 SilentBanker Trojan Analysis

More specifications for the unique characteristics of SilentBanker as to represent as a very dangerous threat was determined specifically by anti-threats and viruses software's solution companies first in 2007 through the Symantec corporation report titled as "Symantec Internet Security Threat Report Trends for July–December 06", then in 2008 by Reed when he wrote his article "New Trojan Intercepts

Online Banking Information" at NetworkWorld journal [7], after then by Theerthagiri in his thesis "Reversing Malware: A detection Intelligence with In-depth Security Analysis" in 2009 with a full description for the MitB attach process [6], and Virus Spyware Removal Guide "Remove Trojan SilentBanker B" in 2010 [9] and didn't stopped there; as attackers are continuously updating their attacks' processes and techniques, new characteristics will be developed and will talk about since then, and an-endless continuous tries will be suggested and to recommend -most of times as an innovative solutions[13, 7].

1.1.8.3 SilentBanker Characteristics

SilentBanker emerged in 2006 as a first malware for employing phishing. That "When victims visited the crooks' fake banking site, SilentBanker installed malware on their PCs without triggering any alarm, SilentBanker also took screenshots of bank accounts, redirected users from legitimate sites, and altered HTML pages" [7, 9, 15].

1.1.8.4 SilentBanker Infection and Installation Method

SilentBanker used different methods in attacking banks: Gaining a certain access to users' accounts can be made through clicking links in spam emails -malware is installed, or so that a try to visit a malicious web sites can provides the password stealer can get free installation on to the users system without user knowledge. Appendix A shows that after this Trojan execution, some files will be created for the purpose of collecting information and saving the Trojan encrypted configuration (A-1, A-2). Next, this Trojan makes entries in the registry, so that it can execute at every time when

Windows starts (A-3, A-4, A-5, and A-6). SilentBanker is doing this by “Redirecting legitimate requests to attacker controlled computers; altering the HTML of pages shown to the user; Altering requests sent by the user to the bank after then; capturing Web sites screen shots of where the user must click instead of type the password; then, Sending full pages received by the victim to the attacker; Downloading new versions of itself and new configuration files; recording user names and passwords; recording clipboard content; stealing cookies, digital certificates, and Adobe .sol files; and sending of all software installed lists on the compromised computer to the attackers” [16].

1.1.8.5 SilentBanker Tools Selection

This SilentBanker malware are using the following tools for attacks [6]:

- a) Using the VMware Workstation 6.5.1 for creating a controlled environment that can provides the suitable environment for the SilentBanker to execute and react.
- b) Using the Autoruns v9.53 for providing the necessary startup information of auto-starting locations and configuration during the system bootup. The location of WAS can be seen in Figure 6.

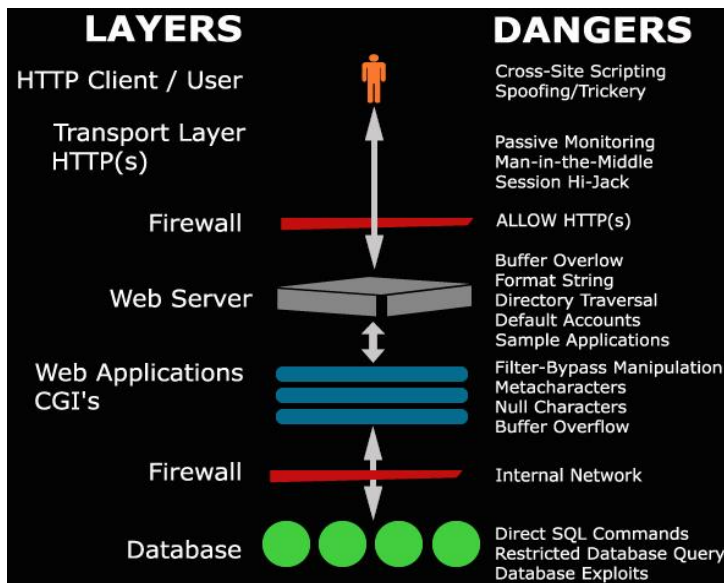


Figure 6: WAS Location [9].

- c) Using the TCPview v2.54 for displaying all of TCP and UDP connection on lists.
- d) Using the WinHex v13.6 as a hexadecimal editor for processing and recovery of low-level data.
- e) Using the Root kit revealer for root kit detection utilization.
- f) Using the Wireshark for providing similar functionality to tcpdump, and supports plugins filtering, sorting, and protocols supported.

1.1.9 Securing the Money-Transferring Process

1.1.9.1 Security Problem Analysis

In real world, an insecure environment fact is neglecting the inability to isolate the cryptographic protocols of their environment around; and presenting operating systems design missing the security consideration as a necessary is a critical vulnerabilities that enforced the problem in a private key inside memory without any

other protection consideration. Presenting deeper analysis for the SilentBanker characteristics', Infection and Installation methods can provide more understanding for the security problem [6].

1.1.9.2 The Society for Worldwide Interbank Financial Telecommunications (SWIFT)

Banks Consortium (1970s) introduced The Society for Worldwide Interbank Financial Telecommunications (SWIFT) for more efficient and secure payments sending means in providing authentication as shown in Figure 7 using an international standards of (ISO 8731) algorithm that present encryption and non-repudiation services for the digital signatures usage. Enhancing SWIFT capabilities by adding MAC keys of shared public key mechanisms between banks using public key cryptography and digital signature protection, but this system has major threat that the main practical attacks do not involved the payment system mechanisms. Other possible technical attacks of Trojan software injection were not used in large-scale bank but exploited procedural vulnerabilities but alert must always be there were any defeating point-of-failure even if it solved and threat couldn't pass over there for the purpose of making a periodic critical management assessment [17]. Figure 7 shows A Sample SWIFT Payment Message.

Account	Transaction Ref#	ISIN#	Security Description	Opening Balance	T/D	S/D	Transaction Type	Amount	Status
Transaction Types									
MESSAGE TYPE (MT)		DESCRIPTION							
MT540		Receive Free							
MT541		Receive Against Payment							
MT542		Deliver Free							
MT543		Deliver vs. Payment							
MT544		Confirmation of Receive Free							
MT545		Confirmation of Receive Against Payment							
MT546		Confirmation of Deliver Free							
MT547		Confirmation of Deliver Against Payment							
MT548		Statement Status and Processing Advice							
MT535		Statement of Holdings							
MT536		Statement of Transactions							
MT537		Statement of Pending Transactions							

Figure 7: A Sample SWIFT Payment Message [17].

1.1.9.3 Encryption and Decryption of Communication

Communication Encryption and decryption can provide the following [17]:

- a. Data communication secrecy,
- b. Encryption systems applications of both electronic signatures and authentication can verify message integrity, authentication and protection of data. Before sending message, receiver performs the calculation of the existing and another authentication code could be generated if message didn't altered using multiple algorithms of Encryption/decryption or the Rivest, Shamir, and Adelman (RSA) one as called as an "electronic signature" because it is from a proprietary private key and no one else can falsify his/her signature while some hashing function can be used if no authentication keys are available. Major problem in using this method is in falsified message generates the same authentication code with the original message. Securing transport tunnel should also be presented using a network protocol encapsulating within carried packets of the secondary network. Layer Two Tunneling Protocol (L2TP) is "an extension of the Point-to-Point Tunneling Protocol (PPTP) used by

an Internet Service Provider (ISP) to enable the operation of a Virtual Private Network (VPN) over the Internet”. Using Vaults of all of data, server and hardened OS vaults can protect both security services design where vital applications of PKI systems and firewalls are taking place –as well protecting e-commerce applications and other business data.

1.1.9.4 Secure Sockets Layer (SSL)

Is a popular protocol that are used for running messages transmission security through webby using a program layer that combines both Netscape browsers and Microsoft under a de facto standard until Transport Layer Security evolvement and located between Transport Control Protocol (TCP) and the Internet's Hypertext Transfer Protocol (HTTP) layers. Sockets part is “the method that is used for passing data back and forth between a client and a server program in a network or between program layers in the same computer, and uses the public-and-private key encryption system from RSA, which also includes the use of a digital certificate.” [19]

1.1.9.8 Security Control

SWIFT system has specific security features including user access control, sequence number control, encryption between operations and end-to-end authentication through the following [19, 20]:

- **End-to-end authentication** provided by SWIFT of applying secret-key and end-to-end authentication payments between banks using symmetric crypto-system by using end-to-end layer for providing a secured transactional level capabilities through digital certificates

- introduced by the SWIFTNet Public Key Infrastructure (PKI): (1) trust Provision through non-repudiation, access control and end-user authentication; (2) sustaining the access-control for end-user, and confidentiality and integrity to end-to-end which all can prevent fraud; and (3) facilitating encryption and individual signing communication.
- **Sequence number control** preventing messages' replication, or deletion or even retransmitting for another date by using a sequence numbering for payments messages that should be check by the SWIFT operating center. This will afford the control under a specific authentication for end-to-end sequences of both banks sides.
- **Users' access control** should identify their identity using an issued password by SWIFT before logging into the system. When password is sent for each parties' separately, each password table includes a sequence of passwords planned under a sequence number rules while each login utilize the next password in sequence so passwords' interception through line tapping cant remark the next password response number for each password client that should be received and checked through SWIFT before transactions beginning.
- **Encryption between operating centers** should be held for ensuring the messages' privacy through the international lines between countries that are protected by encryption. SIPN level core security solutions presents in providing filter zing for an IP Packet for the purpose of protecting network nodes from any denial-of-access hit or unauthorized access –through using the “POP” access control lists for limiting SWIFT authorization; IP Encryption

- Authentication and Integrity as a Hardware encryptors for providing an indirect authentication of IP traffic; Router Authorization for providing all Secure IP Network routers and also work in hiding all of the internal network topology; Firewalls for preventing any unauthorized servers and routers access and service attacks' denials [12].

A proposed improvement model should cover the gap of lacking any encryption between regional processors and banks while knowing the truth of the ability of the attackers to use public key cryptography for getting the right for a privacy stealing vulnerability and authentication using The RSA cipher invention in the cryptography field for deploying the public key scheme in modifying the key escrow mechanism to be used in SWIFT system without leaking private key information realizing that for each bank have its unique private and public keys where various keys are used for transactions' authentication and are unknown to SWIFT as previously were and can provide "bank-to-center authentication and link-by-link encryption from end-to-end"; a SWIFT terminal can produce a random number for each transmitted message as the end-to-end authentication session key, this key will be encrypts after then with the receiver's public key. Using the same procedure to perform bank-to-center input sequence number authentication, the operating center can verify the input sequence number authenticity of the client' banks and so the receiving bank can identify the output sequence number out from the operating center using the same method as a "link-by-link" authentication that can through encryption performs using a distinctively session key for encrypting messages

and session key with the public key of the code in other link side. Authentication code is encrypted then contained in the text of cipher for getting it ready to be performed using both authentication code and decrypted plain text that can be store inside its private key each bank can keep on any required authentication session keys and encryption information in an automatic mechanism while managing its public key for fast cryptography development for simplifying both authentication and encryption mechanism [18].

1.1.9.9 Public Key Infrastructure (PKI) and Digital Signatures

PKI secure infrastructure are used commonly within e-commerce and help in decreasing many of securities threats according to passwords but have many obstacles in usage related to high cost and complication in management and implementation [19].

1.1.9.10 The Secure Socket layer and Secure Channels (SSL)

Securing channels are for ensuring the authentication, data integrity and confidentiality in e-commerce dealings. The Secure Socket Layer (SSL) is defined as a “method of providing a secure channel between clients and merchants using de facto protocol for securing communication channels in e-commerce although it does not provide mechanism for handling payment that positioned above the transport layer and below the application layer in the protocol stack for providing the necessary secure services to many different applications through the web”. SSL establish a secure connection on public key encryption for the merchant server authentication and a shared private session key setup between merchant web server

and customer using symmetric encryption for securing communications after then for providing more efficiency [19] and confidentiality using end-to-end encryption; data integrity and authentication through digital signatures and hashing. Algorithm involved in initializing and communicating in and ending a SSL session between a client and a server based on Gosh (1998) are listed in Appendix B.

1.1.10 The Need for Improving Web Applications' Security and for a Performance Assessment

The continuous Tries to produce an innovative solution have raised the importance of improving web Applications' Security and performance Assessment to ensure both the prevention duty of preventing unauthorized access to systems, and the detection and monitoring duty of cyber-security incidents [4]. The need for a Performance assessment was first recommended By the IEEE Computer Society in 2008 in Web Application Security Assessment Tools; auditing standardizations and regulatory framework were also recommended by Maguire and Miller in their article "Web-Application Security: From Reactive to Proactive" in 2010 for the same IEEE Computer Society in 2008 [8]. Other innovative solutions were represented by the VoIP recommended in 2008 by Adhikari in his article at the InternetNews.com journal titled "Beating Online Fraud with a Phone" as a recommended solution for the Authentication schemes overpass that deeply analyzed by Delgado1, F'uster-Sabater1 and Sierra2 in 2008 in their article "Analysis of New Threats to Online Banking Authentication Schemes" [1]. More focus in defeating the latest one was in

"Defeating Man-in-the-Browser: How to Prevent the Latest Malware Attacks against Consumer & Corporate Banking" article wrote by Enrust in late 2010 [21].

For all; was the call for the need for standardizations' implementation and best practices guidelines, generally accepted government auditing standards and regulatory framework [8], and seems that it is a good idea to have a neutral expert third party for assessment, Re- prioritizing organizations' security risk [1] When all needs will meet; setting the *'right' order for the administrative orders' priorities* and pro-active mindsets will achieved [22] Sarbanes-Oxley Act [23] and the Federal Information Security Management Act [24] are showing the roadmap to an effective and manageable guide for.

1.2 STATEMENT OF THE PROBLEM

Silent banker Trojans e-Banking Threat

A well-known e-Banking Threats of Silent banker Trojans to be represented as a "state-of-the-art in online banking fraud" (which is designed to steal banking credentials and account information of bank customer by taking one of as an HTML injection coding) [6]. The case is when sale of certain products through a Web application occurred; SilentBanker will instill particular malicious HTML inside the client browser for defying any security protections inside any browser and theft passwords and account number and amount [3], assuming for both sides that their transaction has done as shown in Figures 8 and 9 which are examples of Real HTML injection code.

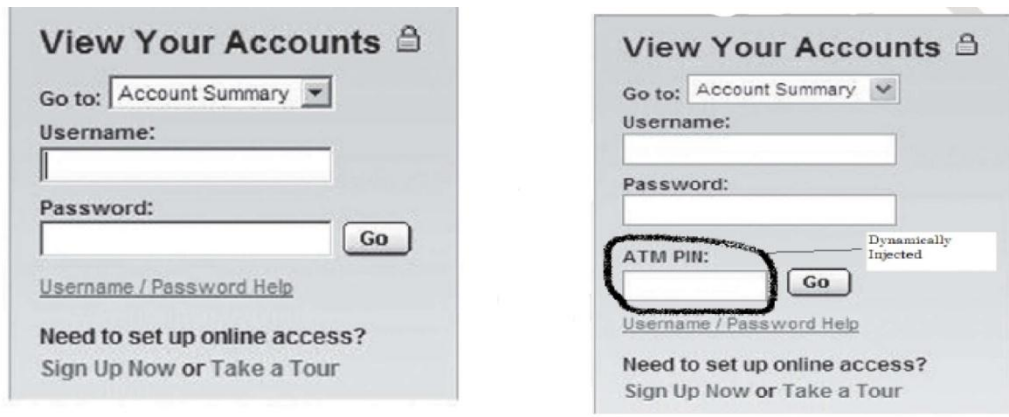


Figure 8: Logging before and after HTML Injection. The ATM PIN field has been dynamically injected into the page via the code inside the .bin file [6].

```

</div><br><div><strong><br><label for="pin">ATM
PIN</label>:</strong><br />
<span class="mozcloak1"><input type="password" accesskey="N"
id="pin" name="pin" size="7" maxlength="4" style="left:75px;"
tabindex="3" /></span>

```

Figure 9: Real HTML Injection code of SilentBanker Trojan [6].

When customer asks e-banking to transfer the necessary payment to a certain payee; the customer writes the necessary information on the e-banking screen that is consists of payee name and the amount that should be transferred to. Meanwhile; the SilentBanker inside Customers' web browser waits customer to press the Next button to initiate payment as shown in Figure 10. As initiating payment as shown in Figure 11; SilentBanker change the e-banking website that is seen by Bank by changing the information inside –as change payee name into the SilentBanker account name, and the amount to transfer- so e-banking transfer the amount to the SilentBanker instead of the real payee, E-bank in its turn sent payment SWIFT [25] backward to the customer but SilentBanker change that information that first changed into the original real

information that customer wanted the payment to conduct to so that customer will not feel that something wrong had been happened as shown in Figure 11.

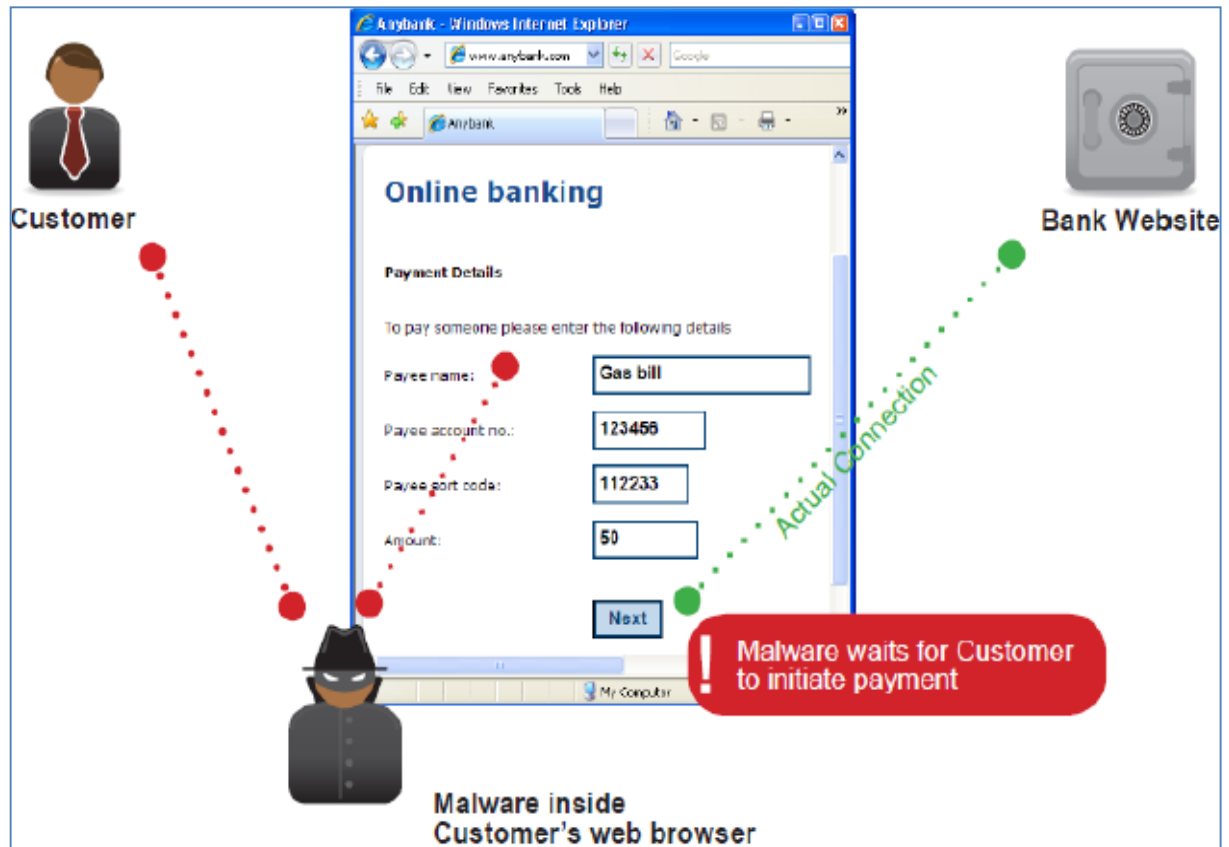


Figure 10: Man in the Browser (MitB) [6].

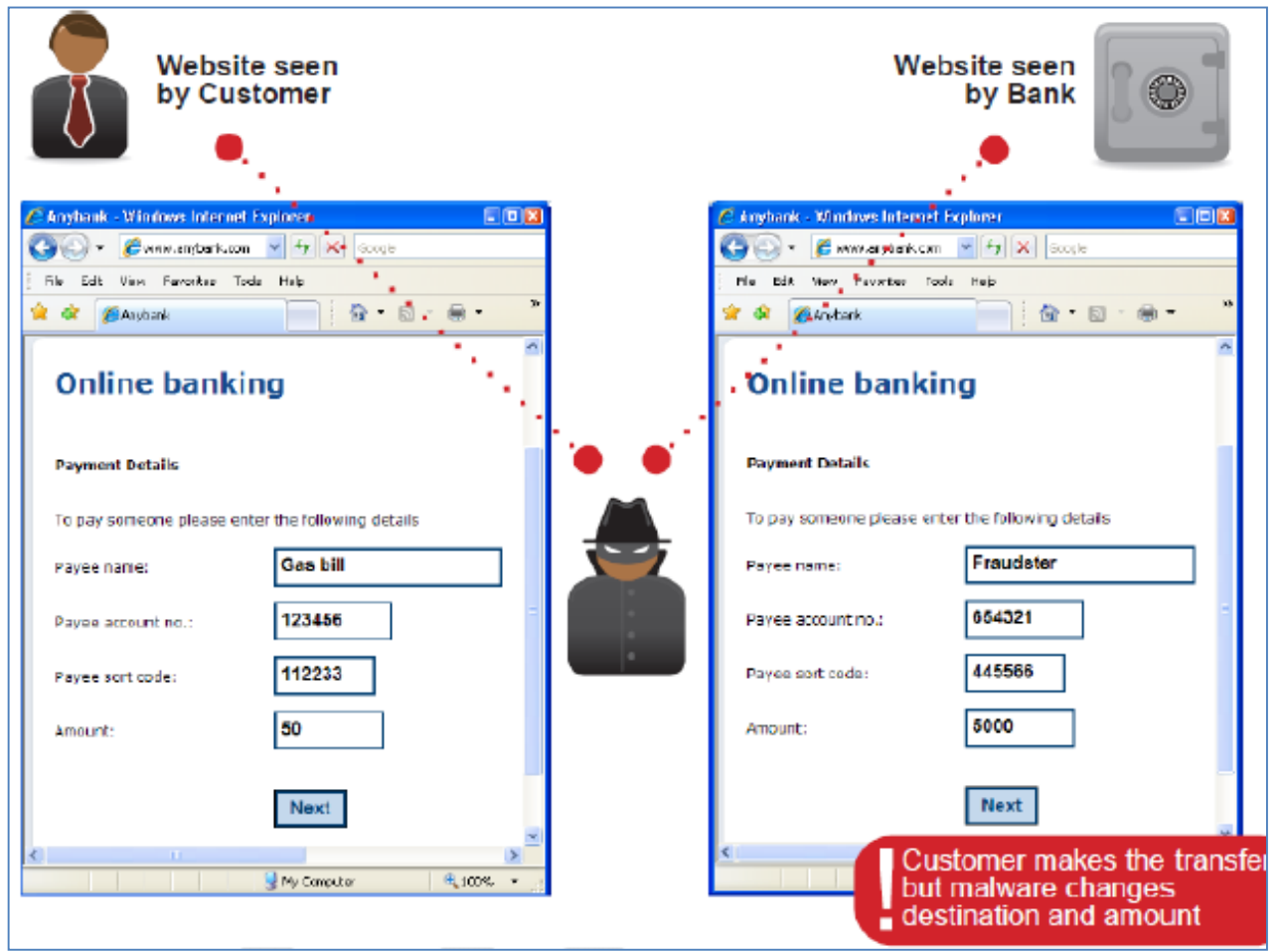


Figure 11: Man in the Browser Bank Transaction [6].

1.3 AIMS AND OBJECTIVES

This thesis aims to analyze a complicated malware of SilentBanker Trojan attack which attacks e-banking system and to understand the security protection that was defeated by the malware. It trying to develop e-banking web application security as a reactive method might complicate communication with clients not to bring the solution for. For that, trying to think in a proactive way might provide the Information Security Industry with the necessary contribution. In order to achieve this aim, and while bearing in mind that such web services security challenges are highly complicated and technologically advanced, to overcome those challenges and complexity and to achieve our aims the following objectives should be tried to meet:

1. Study of the related literature reviews.
2. Study and analyze the Web Application Security Model for e-banking systems.
3. Study and analyze the SilentBanker Trojan Threat; its characteristics and behavior, its way of attach and the attack process analysis, infection tools and detection, and related injection and encryption coding.
4. Finally; determining the conclusion and suggest for recommendations for further future development for both clients (represents as either individuals or companies) and e-banking web application security development for banks.

1.4 SIGNIFICANCE OF THE STUDY AND EXPECTED LEARNING OUTCOMES

This research outcomes should support IT Managers as well as decision-makers in choosing an appropriate control measure(s) that can reduce security incidents' damage or losses of the SilentBanker Trojans attacks, Developing an implementation management Plan for the designated web applications securities applications, increase their awareness on the importance of a frequent running for web applications' security risks assessments and tests-applying throughout the entire development lifecycle [26], increase the amount number of the web applications running in organizations [1], and finally help in setting a mitigation strategy for a risk management system of security incidents that sets the 'right' administrative orders' priorities, change their mindsets into a pro-active instead of re-active one in dealing with vulnerabilities & security risk priorities [8]. All will increase managers' awareness for the need for a strong and solid back up from the top management in order to imply such security process Otherwise; its application will have a very high risk of failing.

1.5 PHP WEB DEVELOPMENT PROGRAMMING LANGUAGE

The PHP is a well-known web development language that is commonly used by Web developers where nowadays is known as PHP5 and used in its fifth major rewrite.

1.5.1 What Is PHP?

The PHP can be defined as a "server-side scripting language that can be embedded in HTML or used as a standalone binary, and produced by Microsoft's Active Server Pages, Macromedia's ColdFusion, and Sun's Java Server Pages"[10]. Where many tech-journals called PHP as "an open source ASP" –in spite of misleading formula- but the reason was because of its similar functionality to that of the Microsoft product.

1.5.2 Why PHP are gaining more market share?

Incrementally, PHP and server-side Java became more usable and gained more momentum over the ASP in market and PHP users are increased to think of it as a "collection of super-HTML tags or small programs that run inside the Web pages except on the server side, before they get sent to the browser which mainly focused on scripting the server-side as well as Command line scripting (in which a PHP script have the capability to run without using any browser or even server for scripting a regular executed using 'cron' (on UNIX or Linux) or Task Scheduler (on Windows) that can be used in its turn in simple text processing tasks) while doing the usual data collection and dynamic Web page production, or even think in sending and receiving cookies and manipulating texts, added for its ability of accessing stored databases inside any disk or network With PHP. PHP's capabilities also include PDF files, flash movies and outputting images, and can also easily output any text, such as XML file. PHP also can communicate other services by using LDAP, IMAP, SNMP, NNTP, POP3, HTTP, COM protocols and countless others [27].

1.5.3 PHP Advantages

Many advantages users can get from choosing the PHP to use [27]:

1. Users can use it for free! As considering the PHP as an Open Source solution available to used in different various platforms under a full- featured and widespread web-based environment. Furthermore, PHP supports the Open Database Connection standard (ODBC) for users to have the ability to connect to any other supporting database of this world standard.
2. Experienced programmers are in a continuously trying to for improving PHP; as a result it is now used in building dynamic data-driven Web sites under a tight budget.
3. PHP is considered as an easy web programming language as it mixes best and most flexible features of both C and Perl languages and included it inside its specifications, while also presenting as one of the strongest web programming language as combining best of its native database features included (MySQL, Oracle, Direct MS-SQL, Informix, Sybase, PostgreSQL, Unix dbm, dBase, IBM DB2, FrontBase, InterBase, Ingres, FilePro, Hyperwave, Adabas D, Solid, Ovrimos, Empress, Velocis) through giving the ability for a direct access to databases using SQL statements.
4. The PHP have the flexibility to be used on all operating systems' types including Microsoft Windows, RISC OS and Mac OS X, Linux and other Unix variants (including Solaris, HP-UX, and OpenBSD) and others. PHP is also supporting the Web servers' usage while being supporting the CGI standard as a part of the web server itself. Web servers are include (Microsoft Internet Information Server, Personal Web Server, Apache, Netscape and iPlanet servers,

Oreilly Website Pro server, OmniHTTPd, Caudium, Xitami, and several others. Anyway, with PHP, user has the 'freedom of choice' regarding a Web server and the operating system.

1.6 SYSTEM ENGINEERING

System engineering is going forward into an emerging stage where quality attributes of security software promotes a critical role for related banking and money transferring applications for the E and M-commerce usage. Reliability, Quality, performance assessment, and the required level for system security provide an influence factors in customers making and buying decisions. Product liability laws are also taking into consideration such security systems aspects. All gathered reasons introduced the necessity for a particular deliberation of system security when scheming high-secure software applications. Explaining all related steps for secured information systems configurations and related necessary information for formalizing such process model of software application can enhance security engineering capabilities supported by providing standards for Information system security.

1.7 THESIS ORGANIZATION

The thesis will be organized by involving the following chapters:

Chapter 1: Introduction.

Chapter 2: Literature Reviews and Similar Studies.

Chapter 3: Methodology of Work.

Chapter 4: Discussion, Analysis and Results.

Chapter 5: Conclusion and Recommendations.

Chapter two Literature reviews

2.1 public fears about strategic cyber breakthroughs

Public fears were due to huge losses both financially and non-financial ones represented by data loss and un-authorized access reported by the Federal Aviation Administration (FAA) in 2009 as a "Review of Web Applications Security and Intrusion Detection in Air Traffic Control Systems" [4]. A public report from the Office of the Inspector General (OIG) found the security gap that are most vulnerable to threats is the Admin/MS system assets and recommend the suitable assessment security tools that can –added for its assessment –identify all of system vulnerabilities in a logical systematic organized approach. Giving high and medium risk vulnerabilities the priority to handle in WAS that providing “Mission Support services” in order not to fall towards a strategic drift, recommendations for the public sector were to **(1)** guarantee the compliance usage for all Web applications with Government security standards; **(2)** supporting the managerial processing through (2.a) identifying specific and famous vulnerabilities that are linked with a specific Web applications, (2.b) on-time installation for related security patches, **(3)** take an immediate reactions for correcting high-risk vulnerabilities under a periodic audit, **(4)** defining resolve solution under the “Cyber Security Management Center (CSMC)” standards and “local Intrusion Detection System

(IDS) monitoring deployment”; and finally, **(5)** identifying the necessary information and procedures remediation in a correct timely-manner for assessing cyber incidents based on CSMC standards [4].

2.2 on-line banking threats supporting the public fears and appealing worldwide attention

More focuses on the e-commerce sector of e-banking threats loses was held by krebs in his articles first though the Washington Post journal as his article "FBI: Cyber Crooks Stole \$40MM from US Small, mid-sized Firms" [22], and "N.Y. Firm Faces Bankruptcy from \$164,000 E-Banking Loss" KrepsOnSecurity Journal article in (2010) [5]. Around the world, organizations and individual users have been successfully targeted and defrauded of millions of dollars. Users not only reacted by switching their banks after experiencing identity fraud, but also there was fighting from the businesses side for the purpose of recovering the millions of dollars lost, and they take several legal procedures against those banks [21].

2.3 silentbanker malware threat as one of most recent dangerous e-banking threat

Knowing that the available e-banking payments methods nowadays are highly insecure and cyber frauds through produced high financial sufferers [5, 8] and critical information losses [4] and business reputation and brand image shaking [14] due to latest innovative online banking threats like phishing, Pharming, and new Trojans malwares generations that are now using various infection vectors tools (like P2P networks, Spam, web, and HTML injection

coding) [6] that can overcome the available protection methods (like cryptography). One of latest Trojans was called the SilentBanker threat that consider as a very dangerous type caused in huge financial losses in more than 400 of large US banks and others worldwide [7, 28] where many articles, web journal and anti-viruses websites had recently wrote about. More specifications for the unique characteristics of SilentBanker as to represents as a very dangerous threat was determined specifically by anti-threats and viruses software's solution companies first in 2007 through the Symantec corporation report titled "Symantec Internet Security Threat Report Trends for July–December 06" [29], it's dangerous also was presented through its infection vector of using HTML injection coding to defeat the two-factors authentication scheme and strong SSL sessions [4]. Trojans are using certain tools were used and presented in "Keystroke logging, HTML Injection, and Screenshots and Mouse Event capture" [6], bearing in mind that malware family is wide enough to include different worms, viruses, and Trojans. its infection and attacking methodologies presented as its capability to "add themselves within existing files and making copies, worms carry themselves in their own containers, and Trojans appear sometime previously encrypted within files that intent to offer other functions [6].

As knowing its attack methodology and WAS vulnerabilities the protection methods can be advice, where previously was the extremely technical protection methods can completely protect systems nowadays it is not; previously, banks used to have the traditional "back-end" control form, while banks nowadays are

pushing towards having an easier business logic of depending on user behavior profiling for including front end authentication of the user's (clients) from their own computers caused emerging many problems of credential theft and user's identity or that needs more advanced security measures and methods [6]. Recommendations for using Phone One-Time Password (OTP) Token for avoiding avoid a real-time MitM attack through sending an SMS message – but this method approved that it is not totally secured, more recommendation are calling for retuning back into the traditional back-end controls for getting more secured systems as a best solution. Online-Banking threats can also be classified into MitB threats that target the client-side operating system [6].

2.3.1 SilentBanker Motivation

Presenting the SilentBanker motivation in stealing the client credential information knowing that clients usually store their ATM PIN numbers, bank account number, and login password as a hostname hunting attack “or called Pharming” can predict host incorrect IP address and bypass information from client to the genuine “authentic” website and “vice-versa” for affording delay finding and authenticity; A web proxy attack collect client credentials information through a malicious web proxy and forward it to a “authentic” website. It was noticed that such MitM attacks can't be discovered easily because unavailability for any warning signs for any mistake as faking website still working appropriately [6].

2.3.2 SilentBanker Infection Symptoms

CleanGuide.com defined the SilentBanker Infection Symptoms as the followings [9]:

“* Google, Yahoo Searches are redirected. Desktop background image and Browser homepage settings are changed. This is a common symptom of a very serious SilentBanker infection.

* SilentBanker slows down the infected computer considerably and you will feel like your computer is stuck. This includes opening programs, shutting down your computer, and slow Internet.

* An unwanted pop ups might appear and corrupts your windows registry and uses it to deploy annoying pop up ads out of nowhere.”

MacAfee added on the previous symptoms the following [15]:

- In the process list Trojan is running.
- Mentioning the presence of entries' registry and files.
- The activities of network with servers.

2.3.3 SilentBanker Infection Method and Tools

MacAfee presents various Infection Method for SilentBanker, including clicking links, spam downloads for Password Stealers, and Injection Coding attack that considers as vulnerable software injected custom malicious code into user server computer (like XPath injection code) [12, 15, 31].

2.3.4 Typical Client-Side web SilentBanker Attack Process

A full description for the SilentBanker attach Process is included in the introduction where a) Data Collection and Analysis, b) Identify Targets and Weaknesses, and c) Customer Impersonation attack were described [6]. Typical Client-Side web Attack presents attacker target is about controlling the client web for maneuver his communication across web application through three steps; first is install injecting malware on the client' browser, after then monitoring the client interaction, and third starting of critical security operations for manipulating the sensitive information flow to the web application for achieving the attacker's objectives [12].

As The Trojan horse prepared by attackers; The Browser Helper Object (BHO) is loaded automatically each starting time of the Internet Explorer (IE), promoting an access to all attackers and within the HTML browser' components so easily can determine the surfing web sites as well as modifying the web pages' contents. Sometimes, "instead of using a BHO, the attackers can inject Dynamic Link Libraries (DLLs) into running applications or choose to intercept and manipulate Operating System (OS) calls. The key observation is that the online banking web application has no way to determine whether the client it is interacting with has been compromised. Furthermore, when the client has indeed been compromised, all security defenses that can secure client communication channel (Transport Layer Security TLS encryption) are concluded to fail, and web application cannot determine whether it is directly interacting with a user, or with a malicious application performing illegitimate actions on behalf of a user" [46].

2.3.5 Anti-Virus Companies are detecting SilentBanker Trojan Threat

Anti-Virus Companies is providing nowadays better understanding for Banking Trojans Malware problems through producing anti-viruses detection and removals solutions. Symantec presents as one of an international leading company that produce an innovative anti-viruses' solutions and services for ensure security for users, but the major problem is presenting in the endless game between both cyber fraudsters and e-banking systems developers; as hackers improve their cyber tools as the WAS developers need to improve the security solutions – as the need for all the time upgrading. SilentBanker known as one of Trojan Malware threats (Also Known As: Spy-Agent.cm as McAfee definition) had recently appealed attention since it's discovered [29]. Virus Spyware Removal Guide "Remove Trojan SilentBanker B" in 2010 [9] and didn't stopped there; as attackers are continuously updating their attacks' processes and techniques, new characteristics will be developed and will talk about since then, and an-endless continuous tries will be suggested and to recommend -most of times as an innovative solutions [6].

2.3.5.1 Client-Side Solutions

Several solutions in the client-side that were presented for the purpose of moderating spoofed web-site-based phishing attacks. The Internet Explorer plug- in is known as pwd-Hash that converts the password of the user in transparent way into domain-specific password. Some phishing attacks' protection is a tool's side-effect. As a reason of considering the password as domain-specific, any

password which has been fished is not useful. For the purpose of mitigating phishing attacks SpoofGuard was developed as solution for plug-in. it looks for “phishing symptoms” such as masked links and names with similar sounding domain [46].

Through highlighting both solutions the conclusion was that they focused on mitigating the spoofed the phishing attacks of web-site-base. And they are considered as vulnerable resource against the attacks of the client side as they depend on the environment’s integrity which they are running in. in similar manner; the introduced solution “Internet Explorer anti-phishing” features are ineffective in order when the attacker has authority over the environment of the user [16].

Securing Online-Banking Services require a “trusted terminal” on the client-side for sensitive Input/output information (of PINs/passwords, account numbers, money amounts) transactions using the TLS cryptographic security protocol that can establish the encryption keys between the browser of clients’ computer and bank server, and can use a separate device for a client “lock-in” form for a specific service provider are seems to be a good solution for cryptography layers’ security but not enough” [32].

2.3.5.2 SpyBlock Technique

This technique intends to protect the user passwords against dictionary attacks and network sniffing. In was presented in order to combine the SSL and password-authenticated key exchange. In addition, and for the purpose of defending from session hijacking, Pharming, and cookie sniffing proposes, this tool which is described

as system of client-size, offers an outline of confirmation's transaction over an authenticated channel. The tool consists of an authentication agent and a browser extension which can run in the environment of virtual machine that is "protected" from spy. The drawback of this technique is that it is requires to be installed and configured by the user, and that opposes the solution of server-side that has to be pure. On the other hand smart cards and chip cards are popular choices for the hardware-based solutions that were proposed for the purpose of making secure input capable on the untrusted platforms. Unhappily, for the attackers it is possible to circumvent these solutions if the implementations depends on untrusted components for example operating system calls and drivers [12, 13, 25]. As another choice for the smart-card based solutions, various solutions have been proposed by using handhelds as a medium of secure inputs [2, 23]. Albeit that the hardware solutions are effective and useful, they are expensive, and have the drawback that the user have to install them [7].

2.3.5.3 Cryptographic Technique

The community of cryptography has also investigated various protocols for the purpose of identifying people over insecure channels [22, 29, 26] a scheme is introduced in one of the earliest papers [14] so they user has to respond to a challenge, through memorizing a secret contains humble amount of five digits and ten characters that was presented but no usability study was provided, the importance of such study is shown at the end of the paper that is presented by Hopper and Blum [22]. For human identification they develop secure scheme, but after using 54 persons and performing

user studies, the conclusion was that the approach of their study “is impractical for use by humans”. 160 seconds average is the transactions period which can be performed only with 10% of the population. On the other hand this scheme takes less than the half of 160 seconds and the successfully completed transactions’ percentage is 95% [21].

2.3.5.4 Entrust Solutions

Entrust Solutions are presented as an “identity-based approach” for fraud detection, strong authentication, PKI, digital certificates, and SSL. Phishing and spear-phishing attacks are also known as a “man-in-the-browser” attack. Interestingly, a recently filed suit specifically highlights that the deployed strong authentication (OTP) tokens –as one of Traditional two-factor authentication solutions - were inadequate for protecting the online business user and identified as an important emerging issue. Where the MitB attacks are “the state of the art” in online banking fraud, common security techniques are presented as an unsuccessful methods for deactivating any threat. Understanding man-in-the-browser (MitB) attack mechanism, and reviewing different counter-measure possibilities assessments can provides an effective detection solution [46].

Favorably, effective strong techniques that can stand against the attacks of man-in-the-browser, either by depending on “Fraud Detection Engines” which operate on a targeted Web page or site in preference to the infected computer. The effective and proven techniques that have the fewest disadvantages are [46]:

- a. **“Out-of-band transaction detail confirmation, followed by one-time-pass code generation”**: this effective technique enhances devices that are carried by end-users such as mobile phones; it also authorizes transaction details’ review outside the malware’s influence on the PC of user.
- b. **“Fraud detection that monitors user behavior”**: this technique detects and monitor the movement of the user though a web site for banking system, and that includes execution transaction steps in addition to the move that leads there, and for financial institutions it adds flexibility in order to adapt to the continuously developing features of malware, it also distinguish dubious activity patterns for the intervention that is immediate.

2.3.5.5 The Manual Removal for the SilentBanker

Anti-Virus companies are providing nowadays an easy-installation toolkit for a manual anti-SilenBanking; CleanPCGuide company provided solutions for detecting and removing SilentBanker (like Spyware Doctor) [9]. McAfee introduced a “hot-link” at its website engine and DAT files for detection and removal, the operated modifications for INI files and/or the system registry in order to hook the startup of the system, will be removed in successful way if the cleaning up process has been done with DAT combination and the recommended engine (or higher) [15].

2.3.6 Web Application Security (WAS)

Given that the truth about developing a certain Security for any Web Application (WAS) of e-Business system is a difficult and complicated process to address but it can be justified by looking at

the generated benefits of securing both applications at all levels and connected network issues of “data transmission security, authentication, and counting access control” for all commercial products security chain included. WAS policies are including “new security technology, Web Application Shielding, a runtime application-level security proxy that used for an automatic recognition for the specific application security policy page by analyzing transaction process with client- The proxy then automatically enforces this policy on returning requests, and preventing hackers from exploiting application vulnerabilities and removing the need for tracking and patching every hole in the application” [26].

Benefits will be generated from providing the necessary security for WAs presents in (a) providing a higher security level –but trading-off in reducing the security resource requirements within the organization”; (b) enhancing the development process among all application cycle stages; (c) making a certain change in security environment that creates the culture of making the security development process to concentrate on its planned process and not to deviate apart, “Errors created throughout the development, testing and deployment stages will not cause security breaches within the web site Nor would any security holes in 3rd party or public domain applications” , this will in total –reduce the required time for building such security development process, and improve the customer experience [26].

2.3.6.1 Web Applications Development Methodologies

The Internet had a huge impact on IS development process methodologies - but little were talked about. Developing such e-commerce applications has specific requirements in drawing its methodology which called the Internet Commerce Development Methodology (ICDM) for providing the necessary a stimulating culture for innovative thinking and the necessary strategic focus in structuring a sufficient managerial framework for managing the engineering development process for e-commerce applications- in total will give the relevancy sensing for the whole process.

Figure 12 shows the ICDM Phases that includes many subsystems inside to consider – including web pages- and consider the necessary customization of factors according to specific industry type [28].

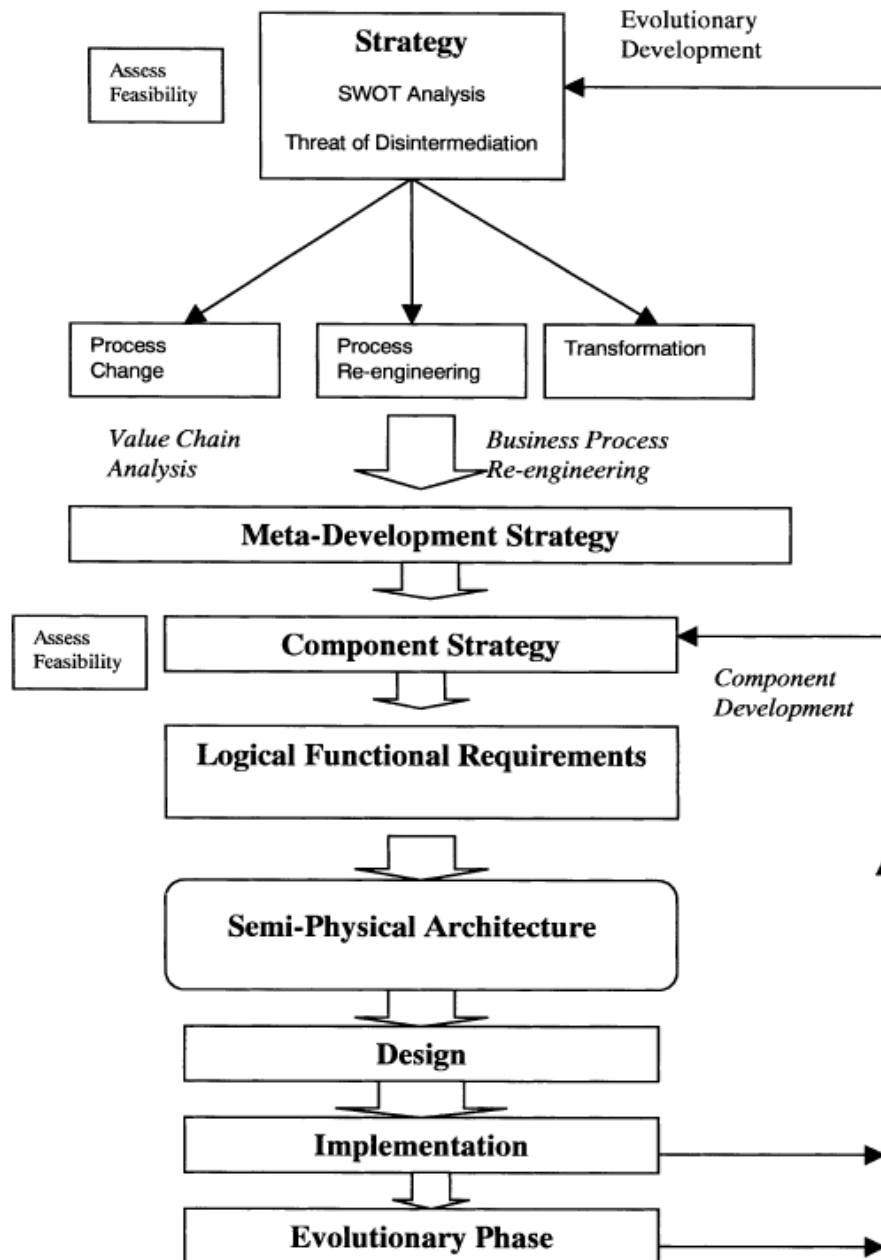


Figure 12: ICDM Phases [28].

2.3.6.2 Creating a Security Framework for Web Applications (WA)

The importance of providing the security framework for Web Applications presents in consistency and order in the analysis and reporting processes for vulnerability classes and applications

analysis process and expand its scales away from a company Web applications into client-server and desktop applications' type –as shown in the following steps [13]:

- *“Authentication—user and entity authentication and process issues.*
- *Authorization—issues related to accurately and effectively making access control.*
- *Configuration management—infrastructure issues, including the configuration of the Web server, application server, and runtime technology (.NET Common Language Runtime or Java Virtual Machine) decisions.*
- *Data protection—issues related to how the system protects sensitive data when it's stored or transmitted between system components or across the Internet from the user to the system.*
- *User and session management—issues related to how the system manages users through such things as password reset mechanisms and transactions.*
- *Data validation—problems arising when the user or system components accept or result in malicious or incorrect data.*
- *Error handling and exception management—issues related to how the system handles security-related errors and exceptions.*
- *Auditing and event logging—issues resulting from auditing and event-logging mechanisms (or the lack thereof).”*

2.3.6.3 General WAS Developer Mistakes

1. It was introduced that common mistakes for WAS are (1) Trusting all data coming from Client-Side; recommendation for analyzing all input parameters data from the client-side [9].

The Information industry nowadays focusing on enhancing WAS securities and services and overcoming its gaps. Analyzing related security vulnerabilities and each type's positives and negatives can make a difference. The advice for companies is not to accept what tool (of code-source analyzer) that vendors' offer but conduct its own test –not on their Web site nor code. Testing tools should be held using a “well-know” piece of code or even a site that you already have been analyzed before for detecting the “right” most critical threats to specific business, and also for defining this tool limitation for further assessment. Recommendation for considering “page count (the number of dynamic pages); • navigation (client-side or dynamically generated, for example); • client-side technology (such as Applets, flash, JavaScript, and AJAX); and • your environment's complexity (such as using forms wizards or Captcha programs—completely automated public Turing tests to tell computers and humans apart)” [13]. Figure 16 shows different WAS tools are used.

To continue the above common mistakes; (1) Check any un-escaped special input strings, (2) Authentication mechanisms that use ActiveX or JavaScript “Lack of re-authenticating the user before issuing new passwords or performing critical tasks, and (3) Hosting of uncontrolled data on a protected domain” [9].

2.3.6.4 Measuring Decision-Makers' Awareness about the Importance of WAS in Companies

Few studies were held for measuring the companies awareness about the importance of WAS for their companies [14]. Considering the Web applications security nowadays as a the biggest concern for many organizations that consider its information and intellectual

property as a critical and most valuable assets to take care about and to protect from fraud [14] . In a survey for CENZIC targeted different companies that tries to make their WA “hacker proof”, Results showed that there’re no deep awareness about hacking the WAS nor its security breaches in their companies. Theft detection was aware for preventing breaches and unauthorized access in the first place while little considerations were held for downloads with no interest in identifying threats where their biggest interest was of losing confidential data that might affect the company reputation. Knowing that Large numbers of companies do not offer a time scheduling for holding periodic assessments for WAS risks assure the truth about the little awareness about the importance of defeating WAS threats and detecting it under a timely-manner mind, the truth of having few companies running for a sufficient numbers of WASs (21% of companies are using 10 up to 100 WAS) that are not totally supported with latest applications (of Web 2.0 or rich media technologies such as Flash, Silverlight and AJAX) can provides a clear picture about the existing gap of updated activities in this field that mostly depends on research and white papers’ results that can be spread through Webinars, podcasts and emails [30]. Recommendations for increase decision-makers about the importance of having up to date WASs, security experts that have a full time jobs in their companies for making more frequent risk assessments for WAs and giving more efforts in defining various threats, and also contribute in enhancing decision-makers’ awareness in WAS important value to their companies [14].

2.3.6.5 Recommendations for Improving the Existing Web Applications Securities Assessment

In "Web Application Security Assessment Tools" article (2008), the need for a Performance assessment was recommended [22]; the security requirements and architectures of such systems Best practices should held a periodic security risk assessments for implementing security standards using specific tools among the whole life-cycle WA development, web service-based information systems to implement Service-Oriented Architectures.

A global approach for software and security engineering should define all of the necessary tasks, activities, security artifacts, tools and organizational structures for securing the WS-based solution and to include within Web Service Security (PWSec) process and applications according to a real web service-based case study. Analyzing, designing and implementing the internal organizational information systems security is by applying PWSec that unite both management and risk analysis with security architecture under a standard-based approach and tool-built support solutions for promoting inter and intra diverse integrated systems [10]. “(i) feeding the repositories specified in the subprocesses as PWSec is applied in more case studies; and (ii) using the UMLPWSec tool” will promotes sustainability for applying subprocesses related activities and creates various required security artifacts for PWSec and pre-defined IT governance regulations [10].

Authoritarian framework and Auditing standards were recommended to use by Maguire and Miller in their article "Web-Application Security: From Reactive to Proactive" (2010) for increasing all decision-makers' awareness about having a proactive role and thought for application's security, that "attackers can bypass nearly all lower-layer security controls simply by using the application in a way its developers didn't envision. Learn how to address vulnerabilities proactively and early on to avoid the devastating consequences of a successful attack". Another call was introduced for standardizing the implementation according to best-practices guiding principle, governmentally approved its regulatory framework and auditing standards [22], advised for having a "neutral expert third party for assessment and Re- prioritizing organizations' security risk" [1] that can set the 'right' administrative orders' priorities under a pro-active mindsets the suitable risk management system for mitigating the risk that are included in security incidents [22].

2.3.7 Inter-Bank Electronic Fund Transfer Systems (EFTs) for SWIFT Coding

E-Business produced a huge business opportunity for introducing fast and convenient payment systems according to specific requirements for every service. Both operational and functional flow of the electronic funds transfer process as its security control method should be examined and evaluated using a standard-leading inter-bank payment system called the "Society for Worldwide Inter-bank Financial Telecommunications System (SWIFT) coding" that has specific security features to stick with. As

Banks usually play dual role as both payee and payer as between bank-bank and bank-client relationships; Inter-bank EFT are usually performed using on-line transactions across private networks for transferring funds. Check payment is a contrary situation where consequence actual cryptographic processes are required daily presents by sorting check, verifying signatures, and capturing information, EFTs presents the same-day at-the-moment payments [17].

Figure 14 shows the transferring method used by the SWIFT inside a transferring network that is described in Figure 13. Figure 14 shows a case when clients' uses commercial bank A for submit money to client B of commercial bank, bank A will send an electronic credit transfer message called "SWIFT" to bank B which credits the payment amount to customer B's account in its turn. After a determined accounting period, the clearinghouse computer system will calculate the defrayal contributed banks and send them through the telecommunication channels to the central bank which use the accounts in the commercial banks for carrying out debit/and or/ credit operations for clearing the transfer amount differences among banks and carrying out the payment fund flow process. The daily funding transfer transactions might include transferring a large amount of money presenting the importance of having a smooth security system that have five essential network security requirements of Confidentiality, Integrity, Access control, Message information originality and authentication for carrying the details of the designated transaction, and legally Non-repudiation [17].

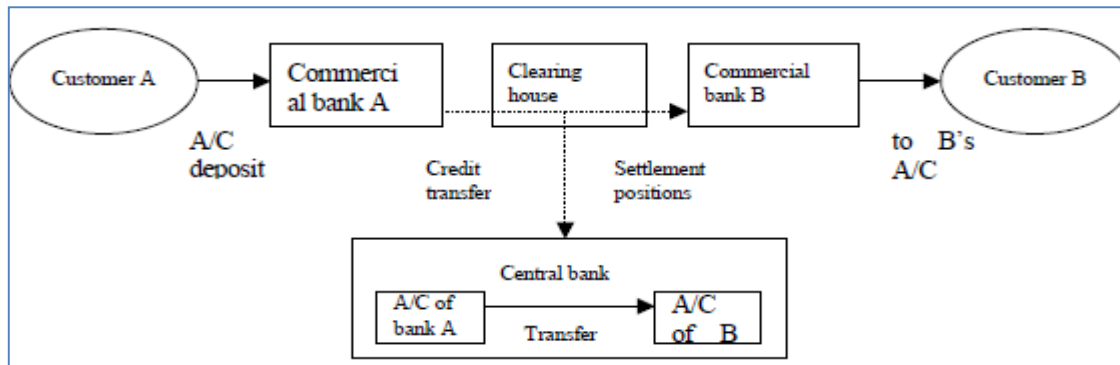


Figure 13: A Method of Funds Transfer to Conduct Payment [17].

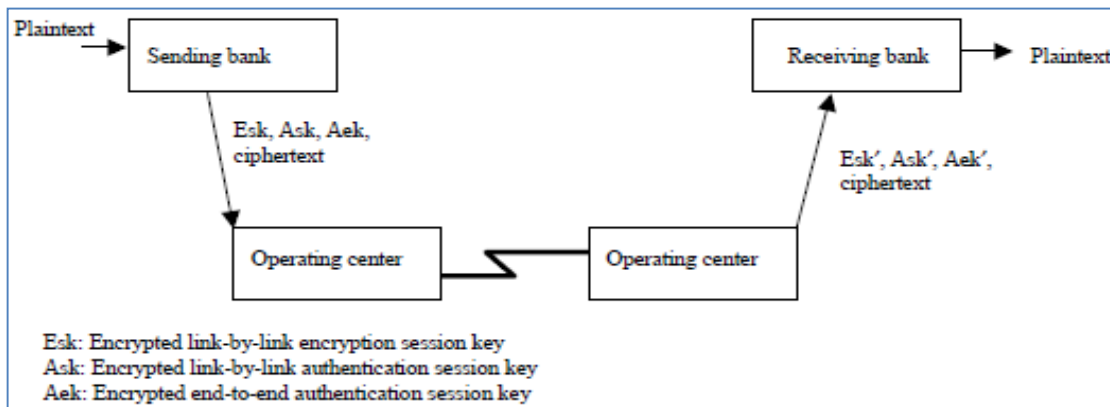


Figure 14: Modified SWIFT Network [17].

Security Risks and Planning can come from hackers as a result of using Open Technologies and internet connections with clients and others by (1) hacking tools; (2) spoofing Packet/Address; (3) Sniffers; (4) diagnostics Stealth; and (5) Sweepers. Current planning for SWIFT security includes: preparing the policy of security; allocating everyday jobs including duties and responsibilities; possessions classification by conveying business risk analysis and assessment for guarantee availability, integrity, and confidentiality and through measuring physical parameters of proper maintenance and locks as promoting a quality standards; promoting security awareness between employees; secure access control and transmissions of message encryption, authentication, signature and origin and delivery proof and system integrity of Operating System security and database integrity and delivery level [17].

2.3.8 E-Commerce Security Mechanisms Transactions

Transactions mechanisms include: “(a) user identifications (IDs) and passwords, (b) Public key infrastructure and Digital certificates, (c) Digital Signatures [25], (d) Secure Socket layer (SSL), and various (e) Payment Protocols” [16].

2.3.8.1 User ID, Passwords, and Tokens

The authentication process of any client includes (a) User IDs and passwords, (b) tokens and (c) biometrics” as fundamentals. The Federal Financial Institutions Examination Council [2001] consisted on three major factors to include at any authentication methodologies “(a) something a user knows (a password or pass phrase), (b) something the user possesses (a token) or (c) something the user is (a biometric characteristic, such as a fingerprint)” [16].

User IDs and passwords technique can provide the necessary authorization through “verifying the user ID against the password provided”. Both Ford (1998) and Verisign (2004) agreed on classifying passwords and IDs identification as the least secure methods are used for authentication, and named five main threats for using password in e-commerce; “Guessing, External disclosure, Replay attacks (by capturing the password and resubmitting it at a later stage, without knowing what it is), Eavesdropping, and Host compromise (attacker are gaining access to the system that stores the password)” [33, 34].

Tokens physical tokens are most time joining passwords to overcome the password vulnerabilities; this combination will create a “multi-factor authentication” and store the private keys for stopping any unauthorized access- which in total will improve the authentication reliability [33] - as the Federal Financial Institutions Examination Council (2001) defined and focused on its “ability to prevent transmission of shared secrets over an open network through local authentication makes it a better alternative than the username –password technique” [16].

2.3.8.2 Public Key Infrastructure (PKI), Digital Signature and Digital Certificates

The public key infrastructure (PKI) technique is providing the necessary “confidentiality, non-repudiation, authentication, and data integrity” for electronic transactions encryption [16]. Verisign (2004) described PKI as “one of the most popular methods of protecting information is through the use of a virtual key system, in which information is encoded according to an encryption algorithm so that it can only be decrypted by a user who holds the correct key” [18]. Using asymmetrical keys are used for performing the inverse operation (one key will encrypt the message and the other decrypt) of one another in order to prevent network-private information-sharing. PKIs have some drawbacks like the slow while involving the trusted third party under a Certification Authority (CA) agreement [35].

Digital Signatures are formed by combining PKI cryptography with hashing techniques for ensuring “authentication, data integrity and non-repudiation” through certain hashing algorithms that performs

the original message contents and compare it with the received message hash for making sure whether matched or not. PKI role comes next in verifying the unique private key for the sender for authentication and non-repudiation ensures [16].

Digital Certificates techniques are for solving the problem about assuring receivers that “an owner of a public key is not claiming to be someone he is not” through showing that the data structures are associating subjects with KPIs [35] under a certificate authorities (CA) control [16].

2.3.8.3 Secure Socket layer (SSL) and Securing Channels

For the purpose of supplying secure channel between merchants and clients the Secure Socket Layer (SSL) method is developed. SSL was defined by Ghosh (1998) as “the “de facto” protocol for securing communication channels in e-commerce although it does not provide mechanism for handling payment”. From the public key infrastructure some methods are used by SSL for the purpose of providing security such as: end-to-end encryption provides confidentiality, while hashing, digital signatures, and digital certificates are providing data integrity and authentication [20]. Through digital certificates server is enforced only with authentication by SSL, and the authentication of the user is performed over SSL by username and password. Radha (2004) cleared that there is no providing for the non-repudiation by the protocol, and as soon as there is a transformation to the server for communicated information there is no security assurance that the SSL is only protecting the communication channel. So may the

server will deny the idea that it received a client order or it has operated unauthorised transactions by using his credit card details. The SSL's inability with the mentioned weaknesses is making it less secure, in order to handle the payments' transferring and for e-commerce, than the protocols of value payment [35].

2.3.8.4 Securing Payments

In order to provide security for the payment services through using payment implementations and protocols through existing payment schemes. There was an indication by Ghosh (1998) that the difference between secure web sessions like SSL and secure payment protocols, is that secure web sessions leave up the details of payment to the merchant and that allows a direct access for merchant to the payment details, however, secure payment protocols and for the of guaranteeing that payments are received by merchants with keeping the details of payment confidential like credit card numbers through providing method [20]. The advantage of secure payment protocols is that the merchant does not have the details of credit card and they are safe-guarded from expected unsafe merchant, on the other hand this method would be useful in order to prevent unauthorized access to the payment details of client if the system of merchant is not secure. The protocols of payment include Secure Payment Application (SPA), 3D Secure, and SET [16].

2.4 software security design and testing

Masalin (2001) discussed in his thesis how electronic banking are nowadays raising its concern about developing information system security under the e-commerce and M-commerce wide applications'

utilization. Developing such systems were introduced in research under Software Security Engineering field where software systems developers have to find information systems vulnerabilities using critical analysis capabilities for analyzing the system security vulnerabilities according to specific type of attack. Global Market release for such security systems applications introduced a systematic thinking about such analysis for security aspects and information system implications under specific phase's determination and quality attributes for system security engineering development. This thesis outlines a personal research conclusion in System Security Engineering field start preparing inside Nokia Networks (1999) as theoretical starting point for security development process for such product (for commercial release preparation) through expressing all related procedures for such secured information systems configuration and product process model formalization for security engineering improvements. "Quality Function Deployment (QFD) method", "Security assurance model (ISO/IEC 15408, 1999)" and the necessary regulated Laws for security requirements can determine the required specifications for such systems security in order to achieve the determined goal of System Security Engineering. Providing the necessary information about "Customer Requirements Analysis and System Security Engineering, Software Design and Development, Security Testing and code review, Certification process, Security in support functions, and Conclusions and Recommendations from the pilot projects" covers most interested areas any security system developer concerns about. This article also provided more related useful websites like "Common Criteria for IT security evaluation,

National Security Agency, SFS ry, Information Technology Security Evaluation Criteria, and SSE-CMM” as try for providing a comprehensive reference for professional security systems analysts [41].

2.5 making strong encryption easy to use

BACKBLAZE (2008) is an on-line company for Business Backup was searching for solutions that protect its clients’ privacy within the encryption area under a trade-off of the ease of use. Recommendations in building military grade encryption system was the proposed solution have the necessary capabilities for providing protection using military grade encryption transparently to provide the required ease of use for users, provides the option for users in changing their password, and allocate utilizing IT for providing data access without asking for any password. The solution was promoted the mix algorithms of AES symmetric and RSA public/private keys that provides the necessary encryption and decryption for both authentication sides. The slowness of RSA public/private keys made BACKBLAZE to think in mixing this solution with another one (symmetric) for improving speed. The US government adopted this solution for military usage, that because of the usual benefits provided for such security systems, this mix gave systems the ability to make many copies from the user’ private key where various passwords can encrypt each copy according to for achieving the purpose of providing an IT access without sharing passwords [45].

chapter three

methodology of work

3.1 Introduction

Under web-security based researches and throughout an audited articles and well-know and reliable Journals and some related surveys' results as a theoretical part, the practical part for this thesis methodology was developed by using the PHP Web development Programming language for defining the SilentBanker threat problem and promote the necessary "security solution" for employing theories into practice.

3.2 System Security Engineering Process Model

Providing the necessary information for designing the "System Security Engineering Process Model" as described in Figure 15 that shows the necessary phases for designing such model; starting by defining the analyzes methodology of security problem features for the desired web application security; designing the process phases, formalizing and customizing it according to the web application security specifications, and finally implement security testing and code review on a small scale as prototyping.

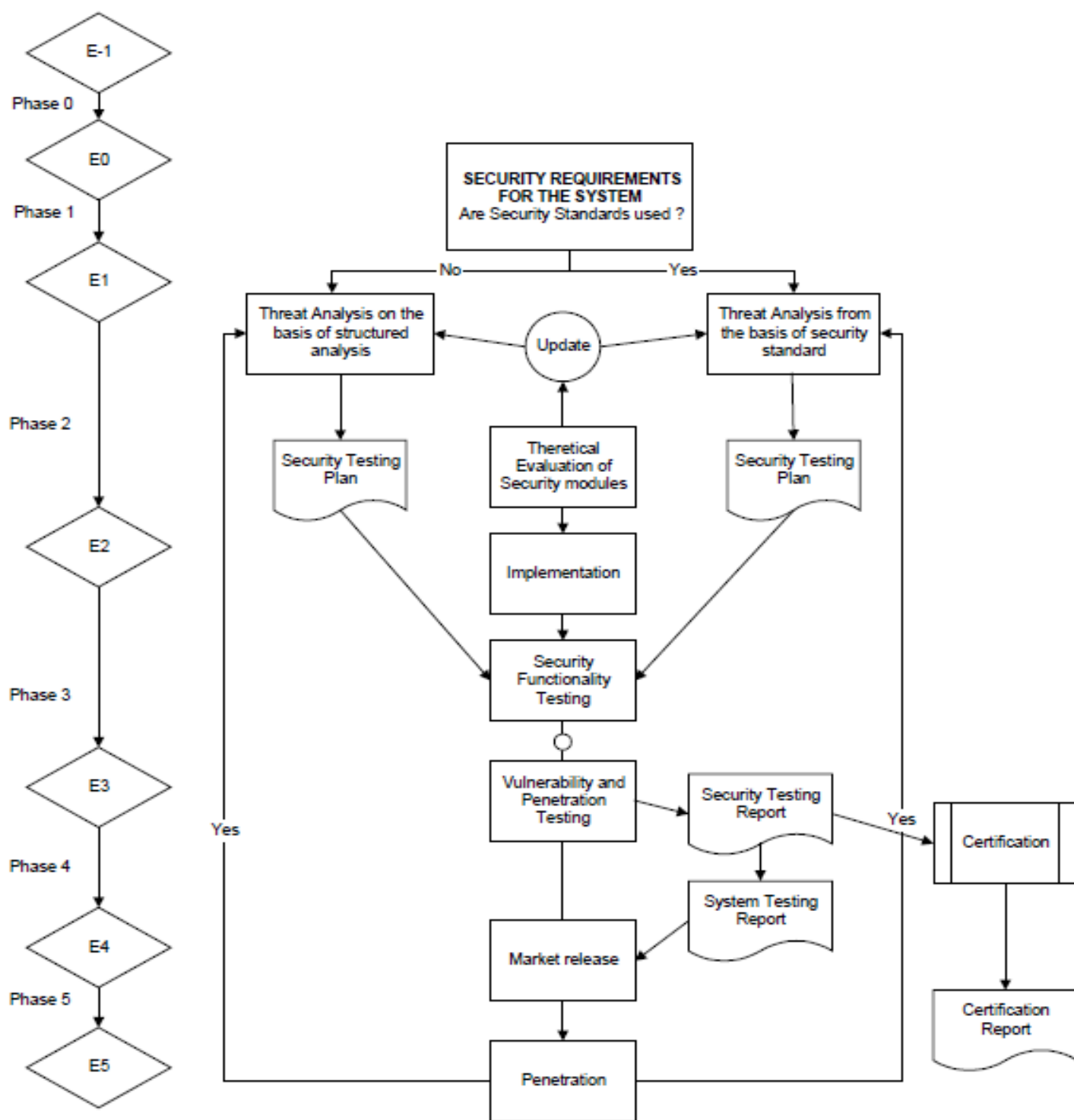


Figure 15: System Security Engineering Process Model [41].

3.2.1 Phase (0): Developing the Suitable Security System Model for Money Transferring System Web Application

The thesis first in **phase (0)** develops the suitable security system Model for the web application of money transferring system through making the necessary analysis for the security problem for developing the appropriate software design –as a necessary stage for drawing the necessary structural analysis for setting the system security engineering process that draws the attack tree and determine the necessary checklist and classify the security requirements' according the desired risk levels; the security system model can be drawn now as we can now for setting the required Security Target (ST) for the problem and Threat analysis implementation in Structured Analysis (SA).

3.2.1.1 Security gaps and weaknesses Analysis

Security gaps and weaknesses in product design are introduced from the unsystematic analysis for specific security specifications' considerations. TA should introduce the required criteria for evaluating system security requirements, vulnerabilities and threat identifications is by using “attack trees and systematic reviews” methods through a systematic threat identification mechanism - after the specification of *security requirements, features and functions*. Security System specifications and the environment architecture (Network configuration including all of the security tools of Virtual Private Networks (VPNs), and the used Firewalls in securing the system) as an inputs for threat analysis process preparation- as shown in Figure 16. Analyzing security threats using Attack trees are determined after defining the operational environment requirements.

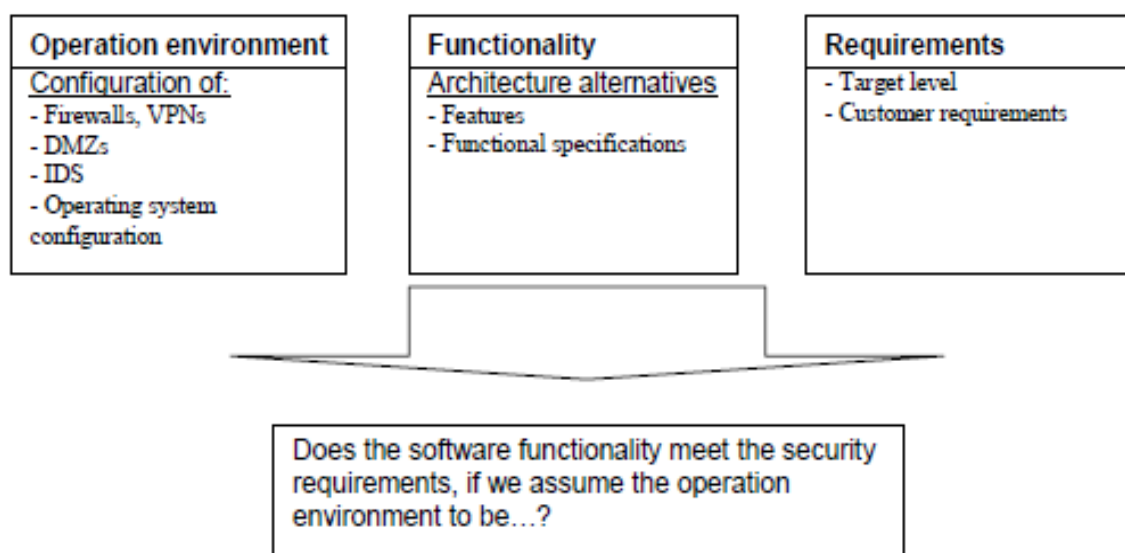


Figure 16: "Security System Analysis" [41].

3.2.1.2 Phase (0) Results

Phase (0) Results determined the Security gaps, weaknesses, and vulnerabilities of Banking Web Application Security that SilentBanker Can Pass through:

- (1) SQL injection hacking.
- (2) XSS: Cross-site scripting.
- (3) Cookie Poisoning.
- (4) Password Cracking.
- (5) HTML injection Code as our SilentBanker case study is.

According to, The System Security Engineering process used both benchmark methods of *checklist* and *attack tree* to make the necessary comparison within the required standards of security in a systematic way [36]. Security Requirements can now be determined according to software requirements and the used standards specifications for defining the desired system security *risk level* , implemented through risk rating usage for "most sensitive

information that is handled by the computer system and the minimum user clearance of logical and physical access to the system” as shown in figure 17. The **'Attack Tree'** helps in determining the necessary requirements for designing the Security Model, and so makes the necessary improvements according to results.

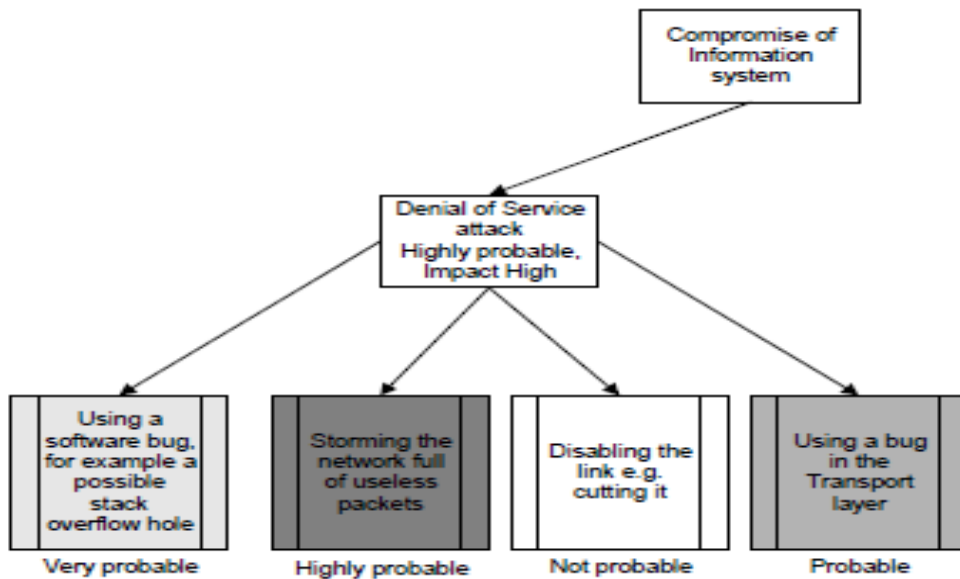


Figure 17: “Attack Tree with Probability and Impact Estimations” [41].

3.2.2 Phase (1): SilentBanker Threat Analysis

Threat analysis will affords for any system analyst an unbiased representation for the system security design and also confirm the specific security features requirement according to the system usage, The Threat analysis consider as the necessary admission document for security testing phase and that is why it consider as the “most vital activity in system security engineering”[36]

3.2.2.1 SilentBanker Infection and Installation Mechanism Analysis

After then, the thesis in phase (1) tried to introduce the SilentBanker threat analysis through analyzing the SilentBanker Trojans threat; its Infection and Installation mechanism of the “Usermode hooks attack” methodology thorough analyzing its security problem and the HTML Injection attack tool, its technical details as a thorough analysis for the SilentBanker problem.

3.2.2.2 Phase (1) Results:

Defining Threat Analysis (TA) as a “systematic security analysis of information system held by software’s engineers - where all relevant security weaknesses are identified, assessed and, possible countermeasures evaluated- through design the user interface and other security related features and in the same time consider security implications” [36], SilentBanker Attack Tool and installation methods can then be determined, knowing that the SilentBanker own specific extraordinary features; including hijacking session, the capabilities of HTML injection coding, HTTP(s) sniffing, and network tracing, SilentBanker Trojan will attack only Internet Explorer “as a Browser-Helper-Object (BHO)” not any Browsers else as it, as nowadays improved its origin rootkit attack technique without using any encryption for the stolen data upload shows how the SilentBanker is consider as a entirely silent and operationally effective and can’t easily be detected. ***SilentBanker Installation Method and Web Injections*** is fully described in appendix C [37, 38, 42].

3.2.2.3 Phase (1) Result shows

Threat analysis (TA) with the help of Structured Analysis (SA) that begins in defining the main architecture for the system, and the scenarios for possible attack under a general level [36] as shown in figure 17 using one of most formal and well-known Security Models “Bell-LaPadula” that “(a) defines the concept of secure state and fundamental modes of access, and (b) gives rules for giving subjects access to objects with specific security policy under presenting three access modes of read only, read & write, write only” [41]. Access rights are illustrated in Figure 18 as a “process between security layers, where subject cannot read object of higher sensitivity and where a star property means that subjects cannot write to object of lower sensitivity, the Strong star property enforces that objects cannot read or write to any other security layer under certain security functions for encrypting communications where trust relationships are analysed using a recognized formal methods while cannot write or read for any further security layer [36, 39].

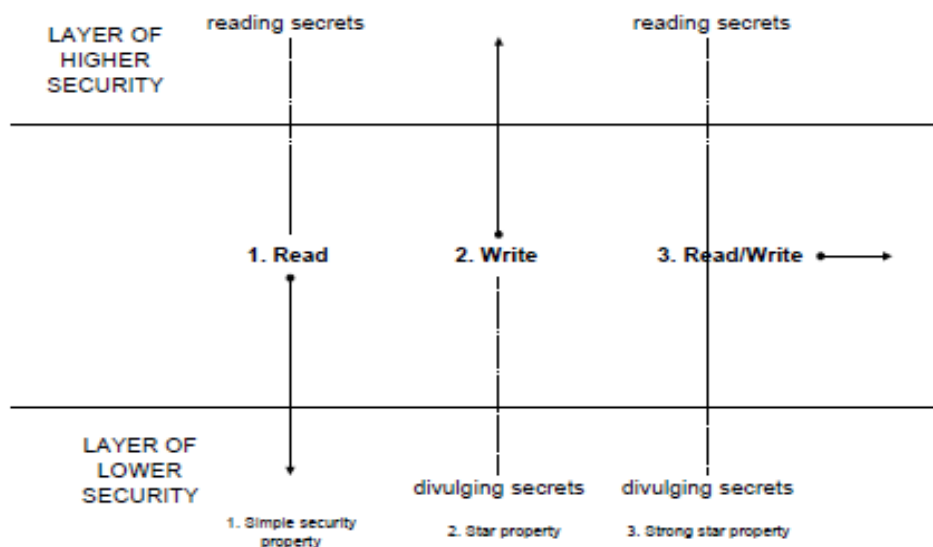


Figure 18: “Bell-LaPadula” [44].

3.2.3 Phase (2): Developing the Suitable Security System Model for Money Transfer Web Applications

As the problem is now clear enough - it's the right time now for **phase (2)** theoretical evaluation of the designated security module through designing the suitable security plan and implementing it for testing the security functionality and penetrating SilentBanker vulnerability [36].

3.2.3.1 Phase (2) Results

After completing Phase (2), the required Security Software System for e-banking that prevents the SilentBanker threat from having the necessary access is completely designed. Evaluating such system effectiveness is explained next.

3.2.4 Evaluating the Designated Security Software

Security Software System design should have a systematic evaluation in order to make sure weather this system has achieved the required security level or not. The illustrated V-model in Figure 19 implemented in each phase for making the required security development [36, 41]:

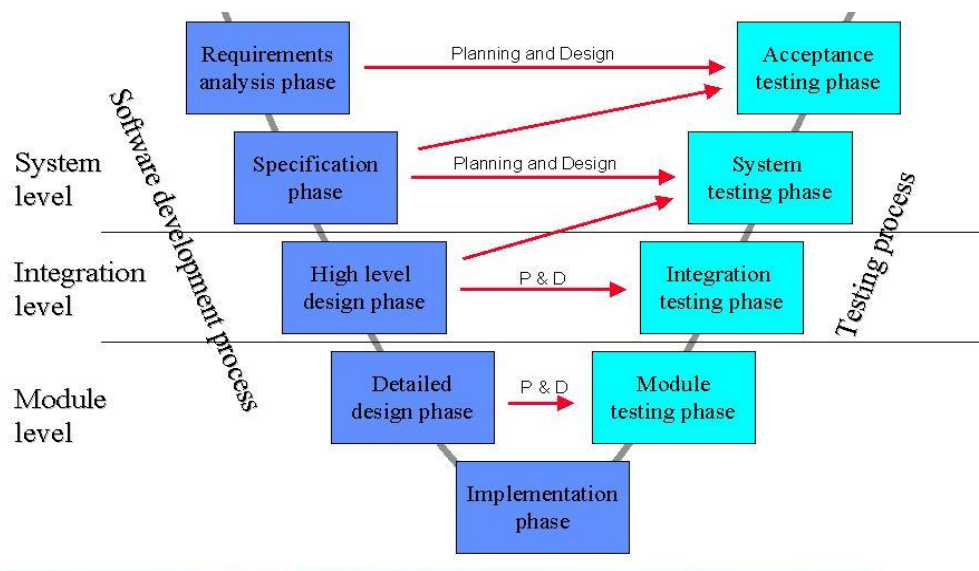


Figure 19: The Software Evaluation and Testing V-model [41].

3.2.5 The last step before real implementation is described in figure

20 and called as the **Theoretical Evaluation** or “security assessment” held for each security functions for a structured standards analysis that held in performing the threat analysis. **Testing Implementation** phase held through security check coding review if someone try to access to the account via trying enter username or password wrong, and show it to the user (there is someone try to access to this user (n) times), after understood the logically of the Web Application how it works found many ways to access to the page like (SQL Injection , HTML Injection , etc) then take care of these ways and block it via code and via the HTTPS mode to make it more secure , and by check if someone try to change the data it gives the user alert message tell him there is someone try to change the data (encryption data the send it to next page , in the next page try to decrypted , if the decrypted dose not work this there is someone trying to change the data , then it’s not work [36].

3.2.6 Other phases

(3, 4, and 5) present the preparations steps for releasing this software product into market use as a wide commerce utilization- which is not as this thesis interest to discuss.

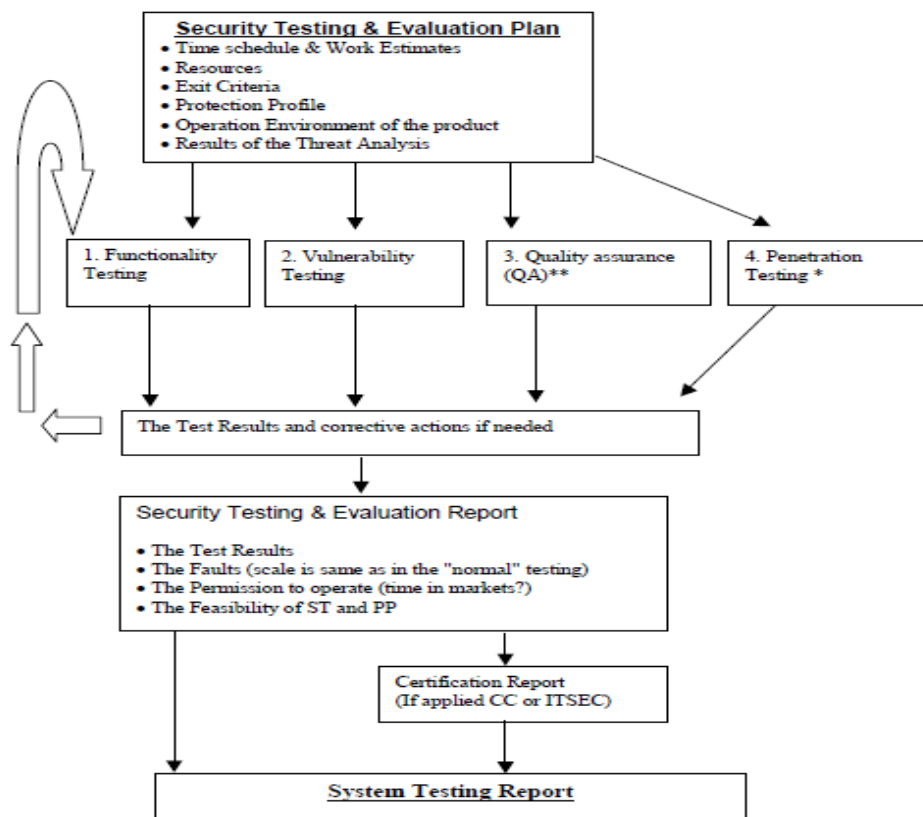


Figure 20: “High-Level Model of Security Testing and Evaluation” [41].

3.3 Security Permissions Access Coding

Security permissions access Code should be obtained from “System.Security.CodeAccessPermission” through introducing the necessary **Demand** method implementation that can clearly defined using the “**IPermission** interface” and “**IStackWalk** interface” combined together. Putting into practice the “Code access permissions (not identity permissions)” is by introducing “**IUnrestrictedPermission** interface” for guarantee an automatic unrestricting permission for the involved parties’ and hierarchy ranking them for through the “**EncryptionPermission**” as shown in Figure 21 [28].

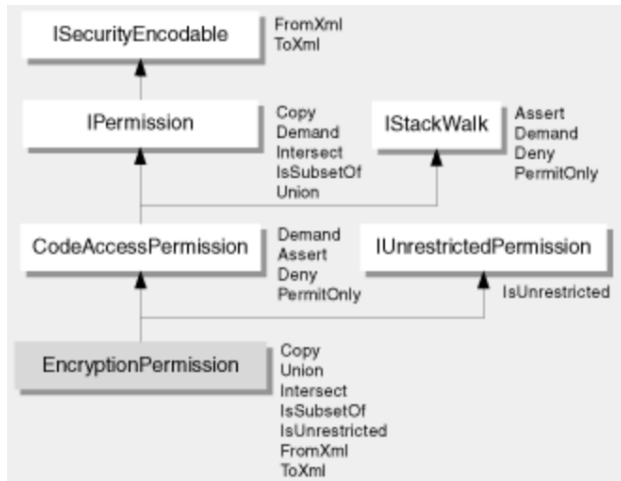


Figure 21: “System.Security.CodeAccessPermission” [28].

3.3.1 Create a Custom Encryption Permission

The Required “Custom Encryption Permission” codes are scheduled in appendix D [42].

3.3.2 Joining Public and Private Keys using Symmetric Keys

The required joints are fully-detailed listed in appendix D [43].

3.3.3 Making Strong Encryption/Decryption

Encryption means to put protecting users’ privacy as a most important priority under an easy-to-use service tradeoff as shown in figures 22 and 23 through “(a) Protect data with military grade encryption, (b) Implement encryption transparently so users don’t have to deal with it, (c) Allow users to change their password without re-encrypting their data, and (d) allow IT access to data without the user’s password through business environments” [45].

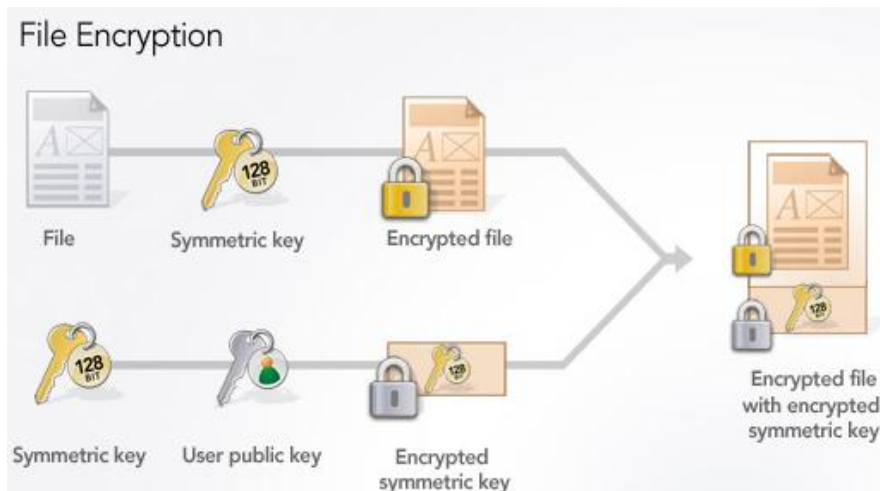


Figure 22: File Encryption [45].

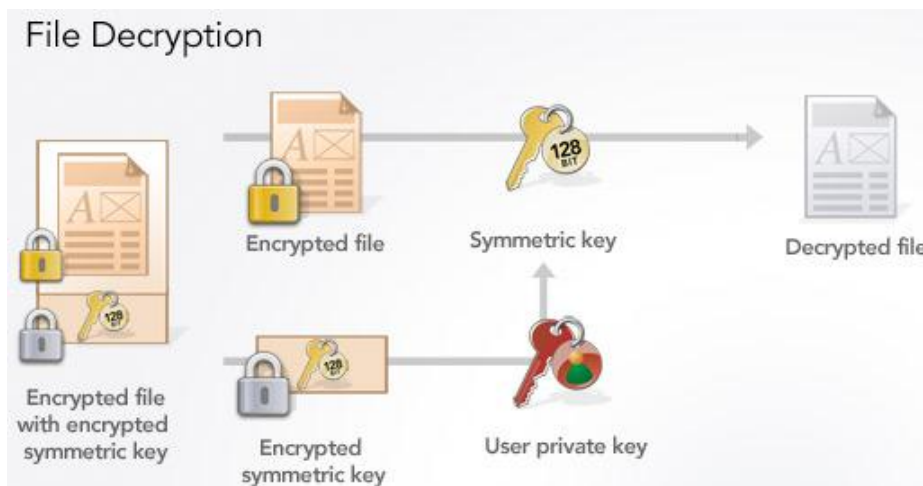


Figure 23: File Decryption [45].

3.4 Using the **RSA for Joining Up Public/Private Keys**

Mixing both public/private and symmetric key algorithms together have a certain purpose; Public/private keys can encrypt data using one key and decrypt it with the other so encrypt data do not condition password type and can speed up the public/private key algorithms slow as a trade-off through using the symmetric key algorithms bearing in mind that if the same key is used in decrypting data, securing data is conditioned in using the symmetric key. Combining algorithms together will promote an RSA public/private key pair

when installing a client, where public key is store on local disk and private key is to be transmitted. Securing the AES key held by encrypting user's public key and sending the encrypted file side by side with the encrypted key over http and will damage the unencrypted AES key. In order to decrypt a file, the user's private key is used to decrypt the AES that used later on in decrypting the file. The user's private key is stored and protected by a highly-secured password. In spite of the unsuitability of such scheme for changeable password, this scheme provides major important benefits; as the AES defines the adopted encryption standard of the US government that protects confidential data, user's public key can run transparently security, password can be used for securing the private key instead of direct data encryption and this password can be changed through re-encrypting only the private keys as well as copying the user's private key and encrypting every copy through using various password for the purpose of promoting an IT access to data without sharing passwords [45].

3.5 Conclusion

This Security System Analysis Methodology adopted in designing the proposed Security System for this thesis according to the required specifications usually are used in software engineering in order to standardizing the required security protection design that promotes in reducing designing errors for fully implementations for the required tasks.

chapter four

the proposed solution

4.1 Introduction

This thesis proved that the traditional security techniques are ineffective, and should be improved continuously. SilentBanker Trojan Threat is presented as a difficult dilemma to solve since the facts about its characteristics exist. The proposed architecture and Prevention Mechanism for the e-banking web-application security are to recommend as a presented proposed solution through three phases listed in Figure 24; the first phase presents using Securing Sockets Layers (SSL) to protect any e-communication session transactions with distant clients through "locking" Browser for each web application but was not enough stand-alone solution and presents the need for both the second and third phases to choose the right places for both encryption and decryption coding using "Digital Signature" method for defining the digital identity for users through using both private and public keys in both the encryption and decryption coding; this combined solution will provide an efficient one and will hoard time and money.

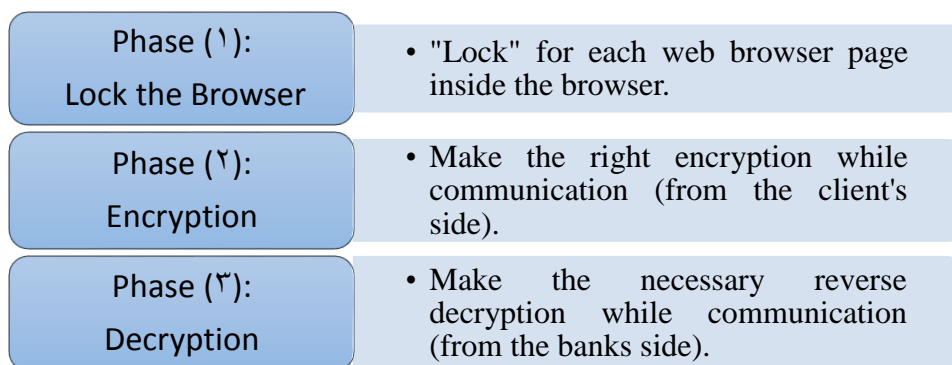


Figure 24: The Three Proposed Solution Phases.

4.2 Results

Results were showed as the following:

On the Client-Side

1. The client will first try to enter the e-banking screen shot –out from his/her computer.
2. While considering the “lock” icon included at the bank screen inside the down-bar (for preventing the SilentBanker entry), e-bank will ask for both the client’ username and password that can pass the security specifications inside the e-banking web application as shown in Figure 25, the necessary entrance -coding (E.1) at the e-banking WAS is shown in Appendix E.

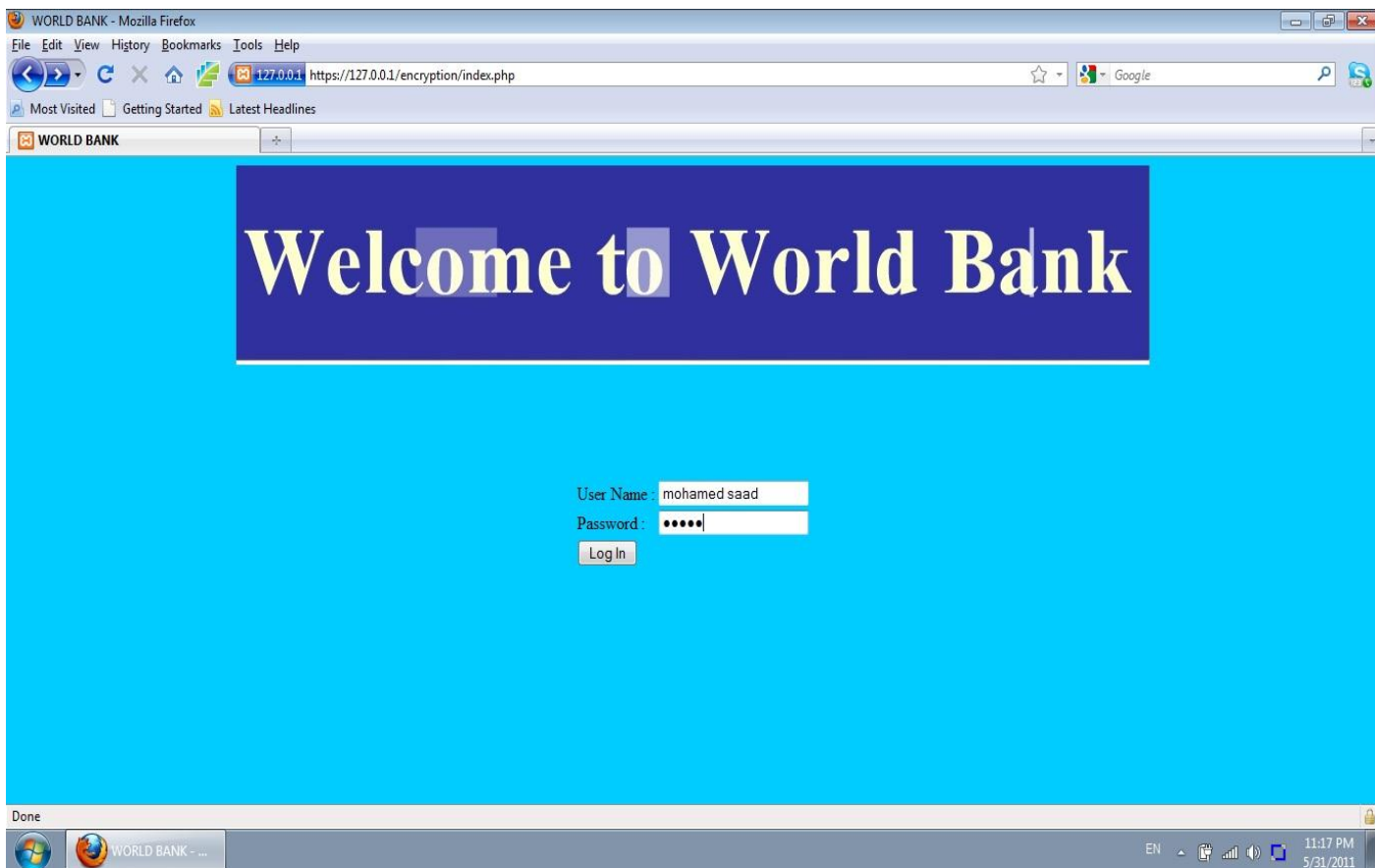


Figure 25: The Entry Banking Screenshot.

In Case of No Penetration:

Bank Authentication Scheme

3. Bank Authentication Scheme: The Bank - in its turn- will check about the client private data (of user name and password) if it is already available inside its client data list or not as shown in Figure 26, the necessary Check -coding (E.2) at the e-banking WAS is shown in Appendix E.

4. After the check and ensure that everything is correct, The Bank then will ask the client to enter the payee information in the right place.

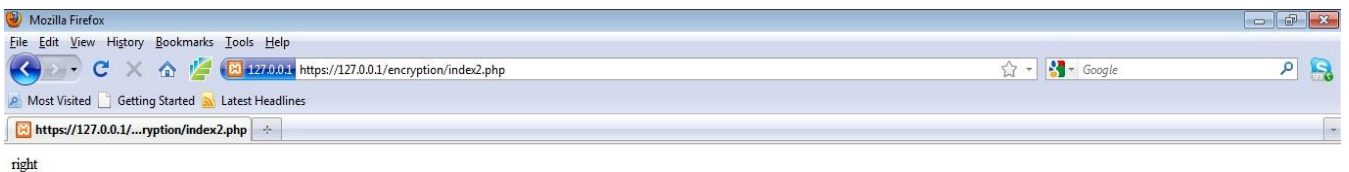


Figure 26: Bank Authentication Scheme.

Client-Entry

5. The client in his turn will enter the necessary full information, the transferee amount and currency type - as shown in Figure 27, the necessary coding (E.3) at the e-banking WAS is shown in Appendix E.

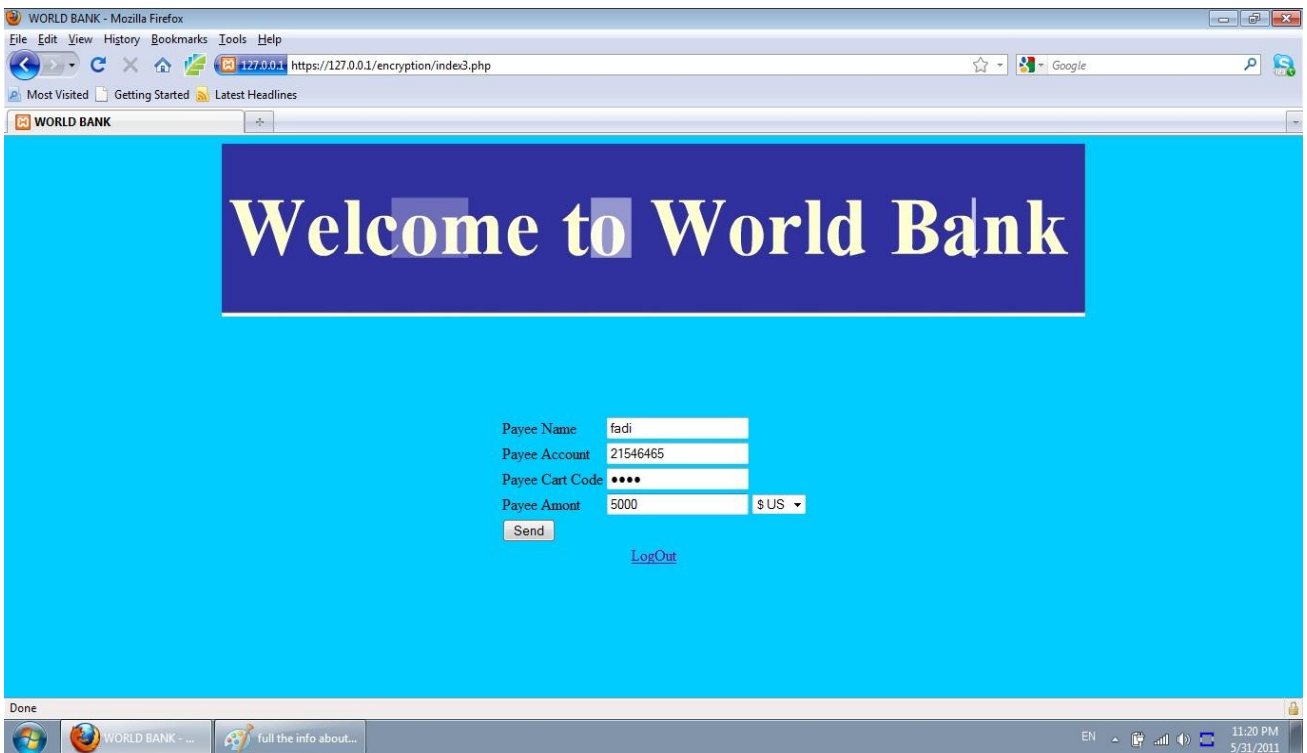


Figure 27: The Client Enters The Credential Full Information In Order To Complete The Transferring Process.

Bank Decryption and Verification processes

6. Bank Decryption and Verification processes are described in Figure 28, the necessary coding (E.4) shown in Appendix E.

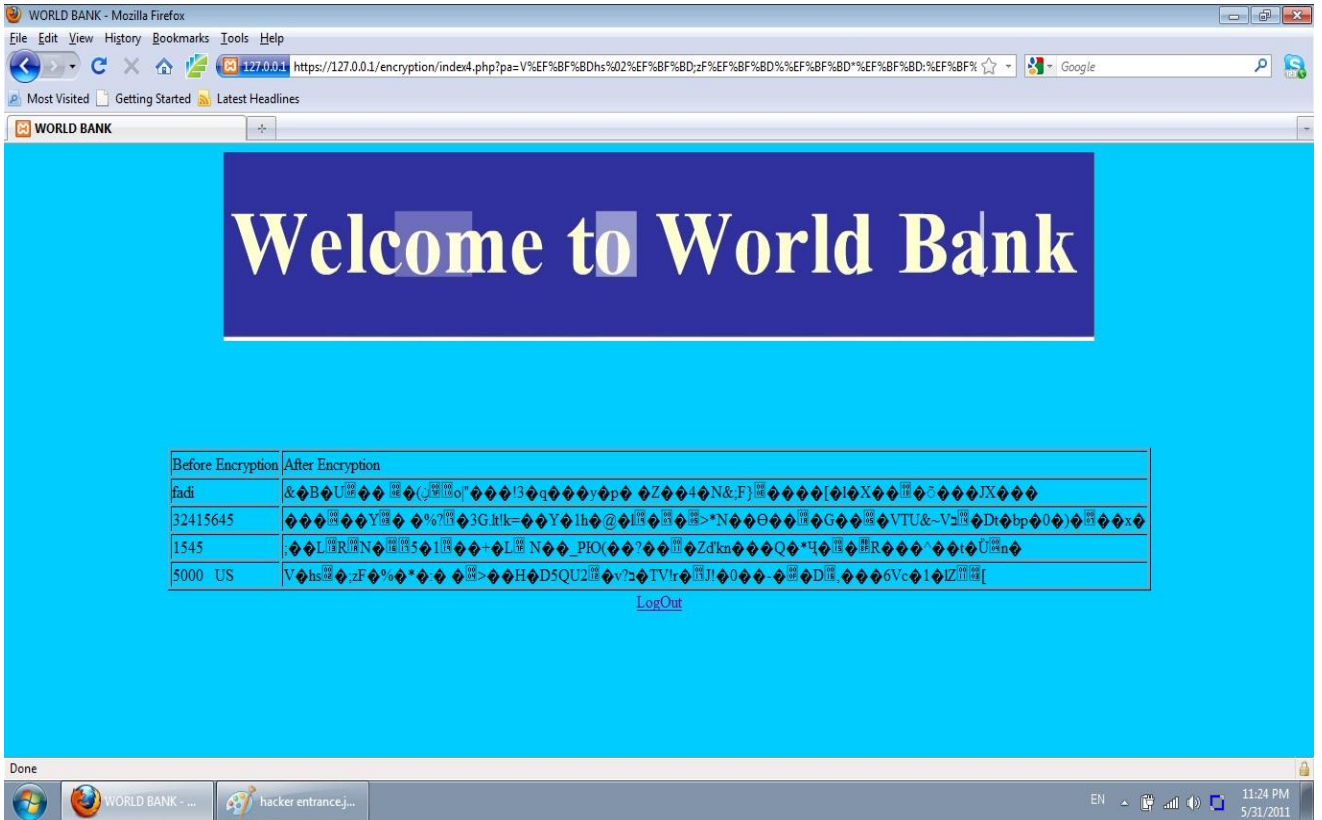


Figure 28: Bank Decryption and Verification Processes.

7. The Bank screen shows that the previous operations' process is successfully done and nothing wrong is happening—as described in Figure 29 and 30.

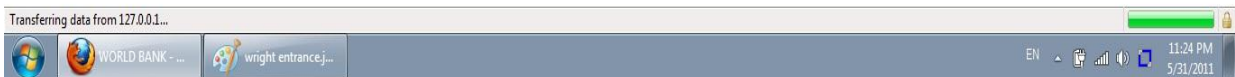
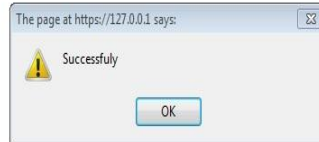


Figure 29: The Bank screenshot presents that Transferring Process is Successfully Completed.

In Case of Any Penetration:

The following scenario presents on both sides:

Encryption and Decryption process on Bank side:

On the Other case; were the encryption and decryption process that held by the Bank Web Application Security System showed a penetration for its web security, the below Banking screen shot (Figure 30) showed text message till the client that “there’s something wrong” and asked him to try to login again.



Figure 30: Bank Web Application Security System showed a penetration for its web security presented as a failure login.

8. If something is wrong (if someone try to access to the account via trying enter username or password wrong, and show it to the user (there is someone try to access to this user (n) times), Bank screen will deliver a failure notice that shows that the transferee process didn't successfully complete. As can be notice; WAS of e-bank couldn't encrypt and decrypt the hacked entry data as can be noticed; WAS of e-bank couldn't encrypt the entry "hacked" data- so bank decided to stop the transferee process and then alarm the banking system about the hacking process as described in Figure 31.

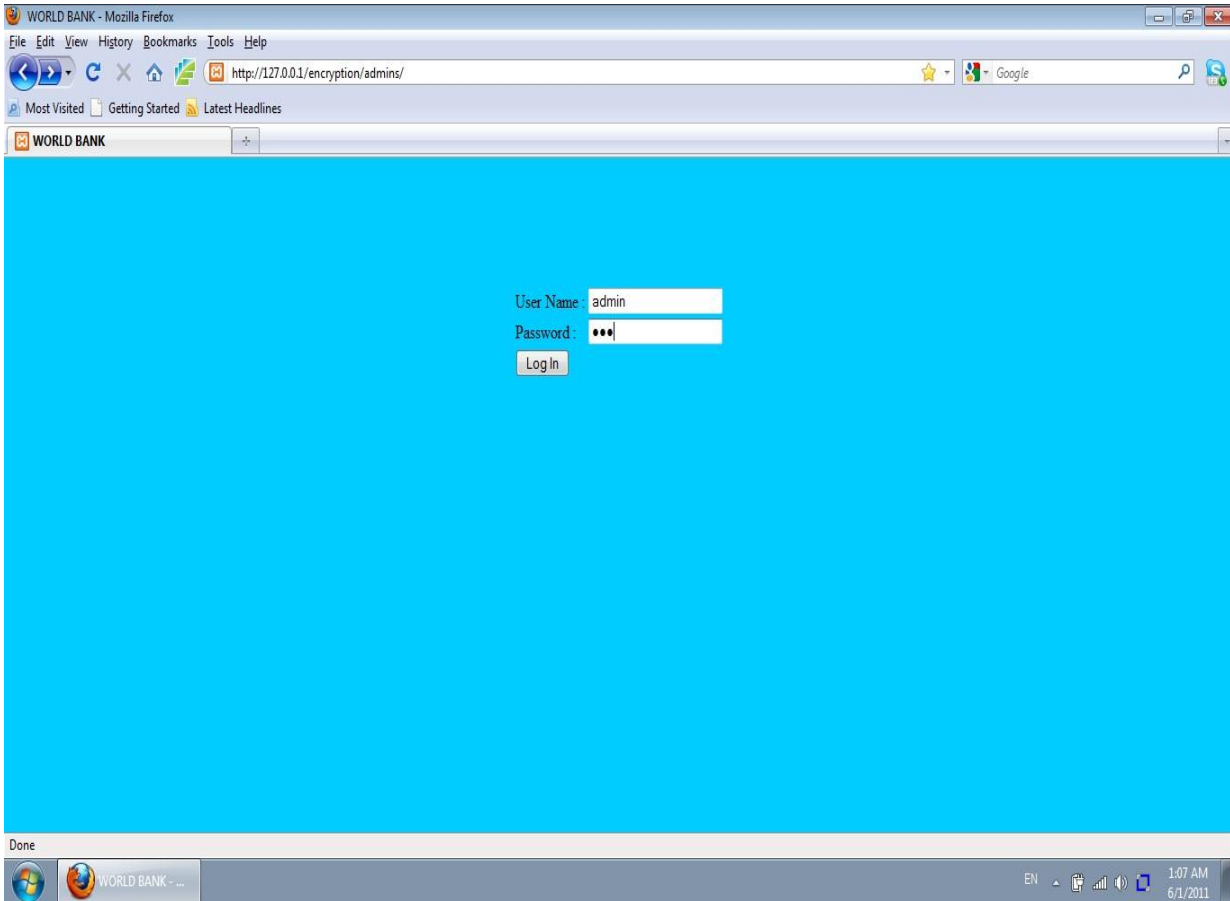


Figure 32: The Banking System adds a new client account for its clients' Data Base.

The successful adding for the new client is shown in Figure 33. The below Banking Screen shot also shows the possibility of Deleting clients as adding them to the bank database.

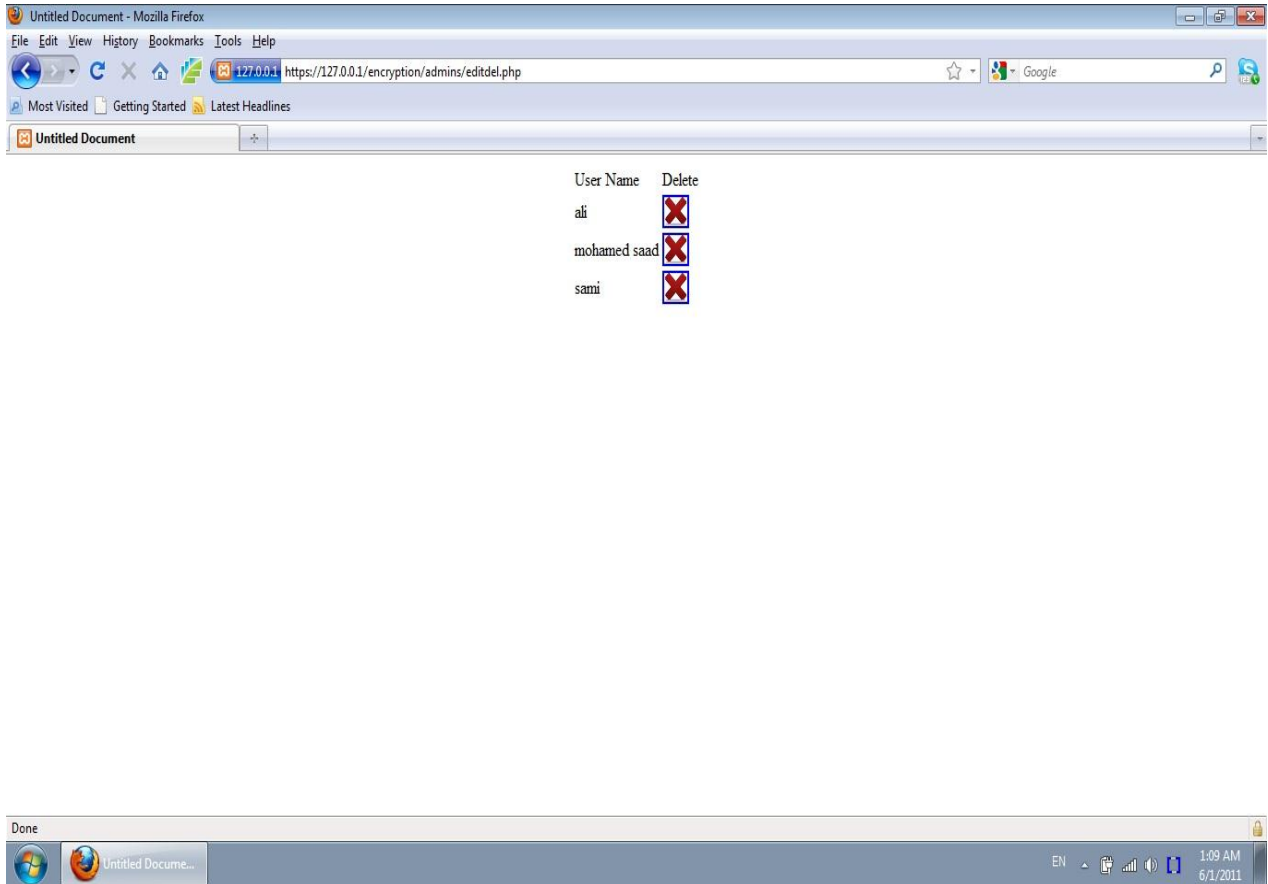


Figure 33: The successful adding for the new client.

Figure 34 shows that the addition process is successfully completed.

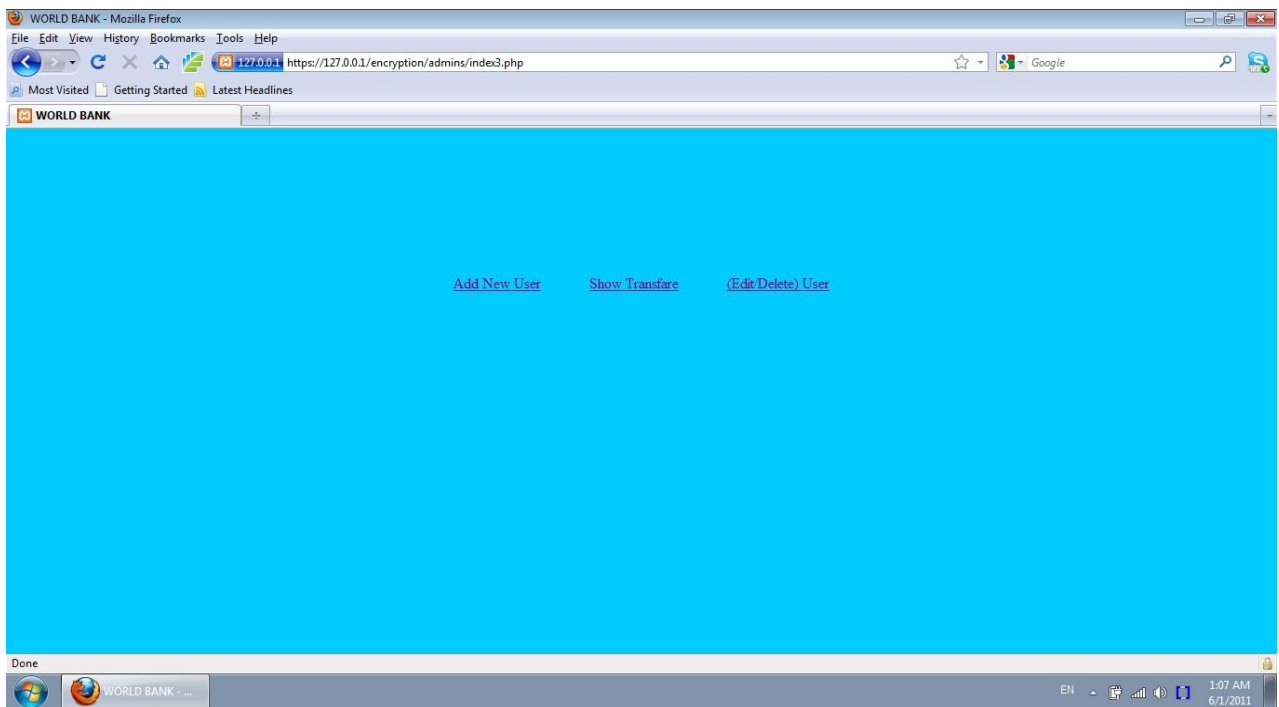


Figure 34: Bank Screen Shot Shows That The Addition Process Is Successfully Completed.

Figure 35 shows the additional process for execution.

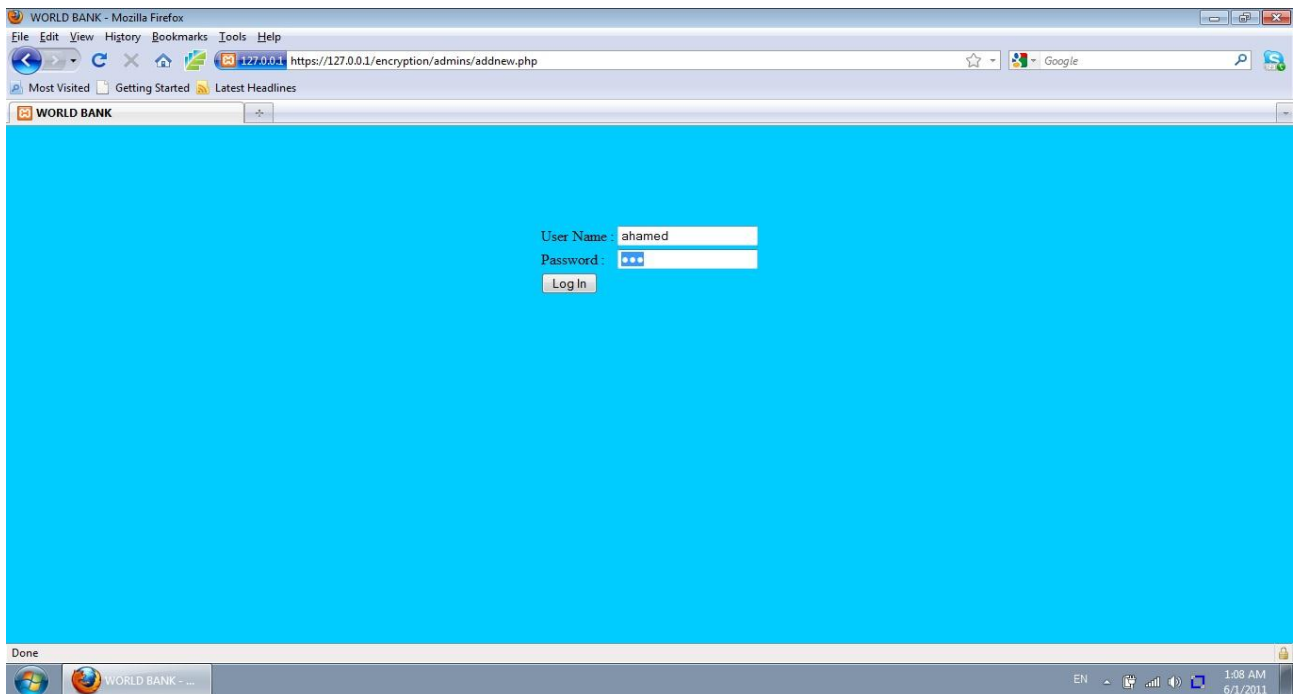
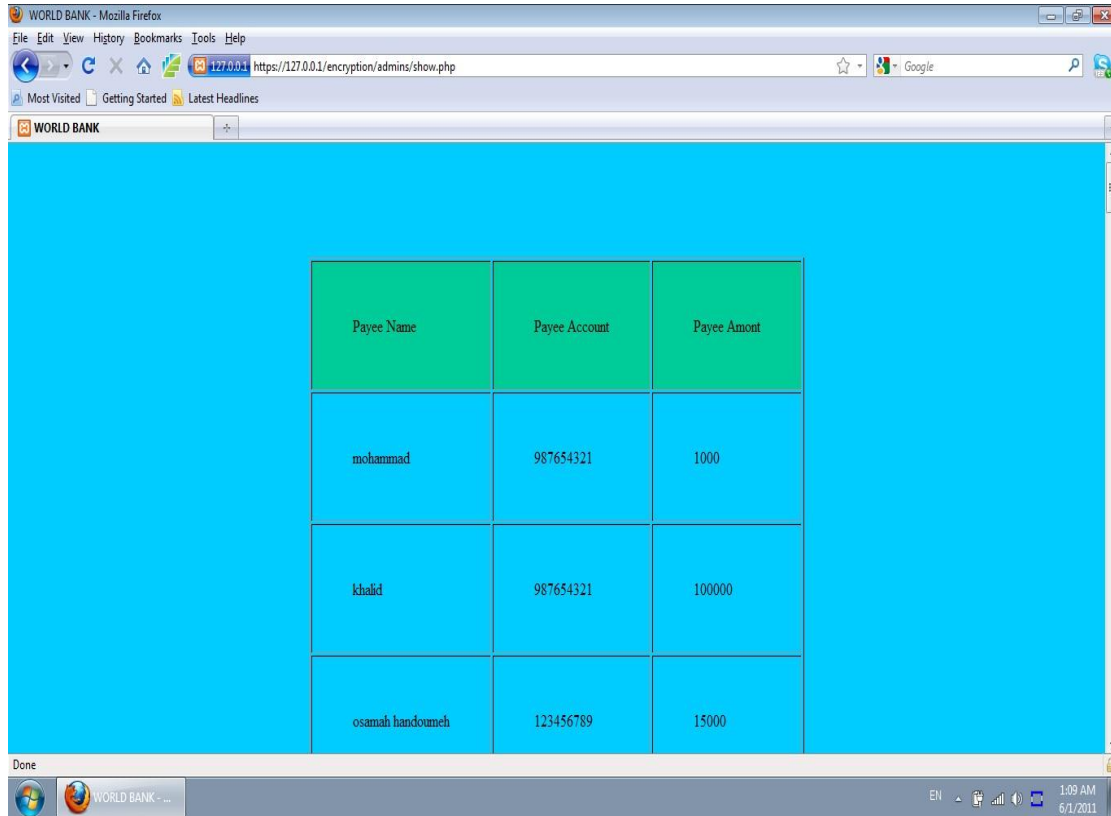


Figure 35: The Addition Process execution through the Bank Server.

4.2.3.2 The Bank Money-Tranfaring Process from the clients' account and according to the determined ammount is shown in Figure 36.



The screenshot shows a Mozilla Firefox browser window displaying a web application interface. The address bar shows the URL <https://127.0.0.1/encryption/admins/show.php>. The page content features a table with three columns: 'Payee Name', 'Payee Account', and 'Payee Amont'. The table contains three rows of data. The background of the page is a solid light blue color.

Payee Name	Payee Account	Payee Amont
mohammad	987654321	1000
khalid	987654321	100000
osamah handouneh	123456789	15000

Figure 36: The Bank Money-Tranfaring Process.

4.2.4 The proof for the Lock Status

Locking the Bank WAS Layers is shown inside the status bar of the web page –as in Figure 37.

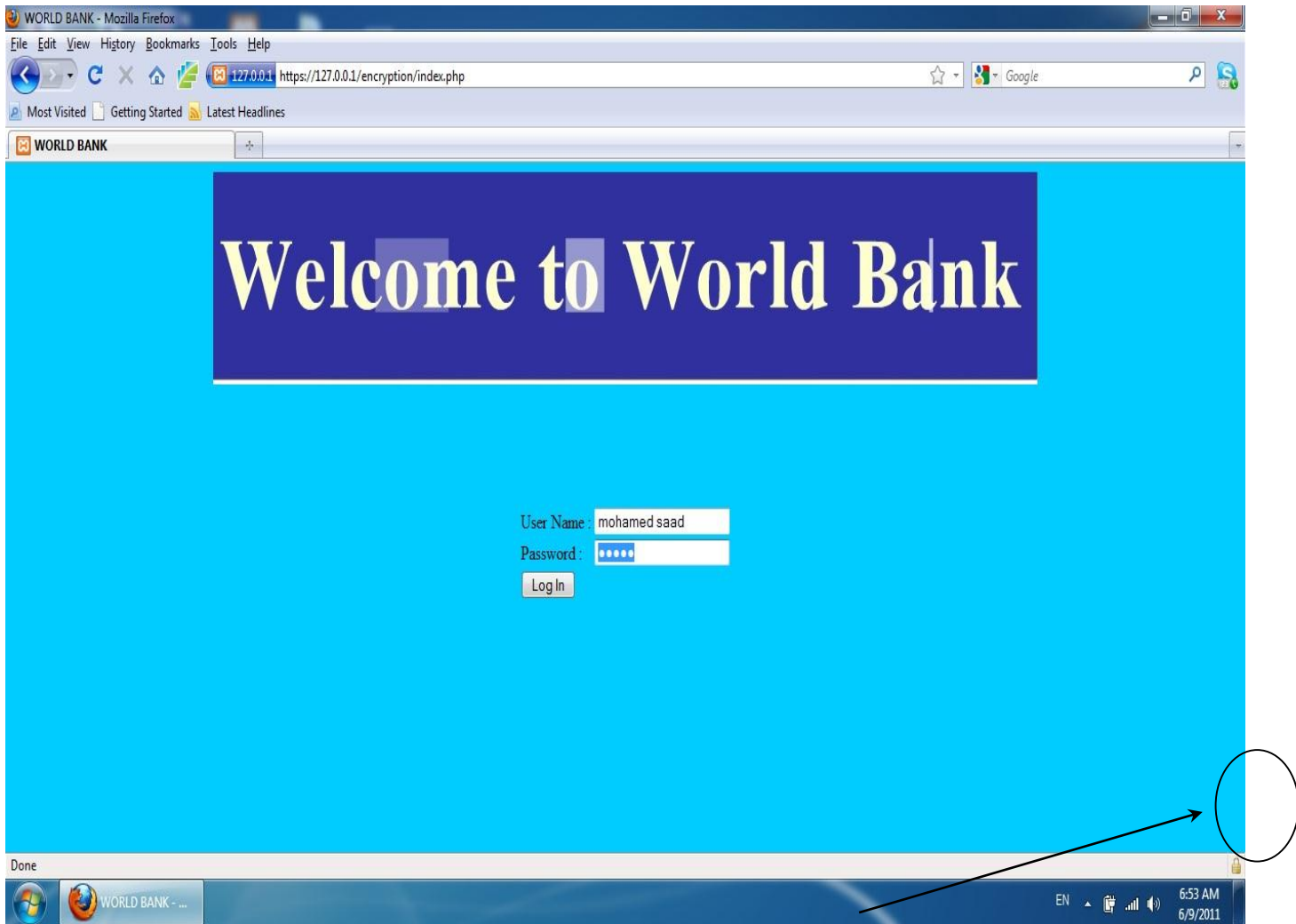


Figure 37: The Lock Status Proof.

In case of penetration, The Lock Status will detect the penetration as sending a message of “try again later” as shown below in Figure 38 and 39. In case of the Locking is in-exists, the SilentBanker won't be detected and that shows the SilentBanker successfully Stealth the client' credential information of both username and password.



Figure 38: "Try Again Later".

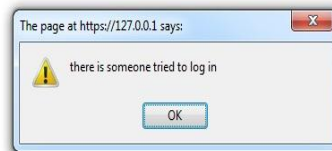


Figure 39: The Penetration Status was detected from the Bank WAS and a message was sent showing it.

Chapter five **Conclusion**

5.1 Conclusion:

This thesis discussed one of most dangerous threats that e-money transfer process are facing nowadays, called the “SilentBanker” and showed related attach process while the transferring process occurred and recommended a proposed solution for preventing this threat attack and so transferring process will executes safely. The Solution was presented through recommending a combined solution- as knowing that such threat can work through using the injection coding attach tool; locking the browser was the first defense line to use, the RSA encryption and decryption coding was the second one that uses for indentifying the payee identity- where Banking side presents major role in the detection process as checking whether the transfer process was successful so to successfully transfer the amount and without any error through, or to inquire both bank and client sides about the failures if not. This thesis tried to combine mixed stages for introducing best results in preventing the SilentBanker attack.

5.2 Future Work

Future work is recommended to implement this work on a large-scale for the commercial market usage; in order to do so, using Quality Function Deployment (QFD) in business analysis for determining the quality standards under a certain quality standards' portfolio for all of structure, procedures and control - should be used

for giving sufficient assurance for the customer in market in having Trusted Computing Base (TCB) documentation that can guarantee security for this application.

Work on developing the Voice over Internet Protocol (VoIP) technology usage as a one-authentication communication channel was another recommended solution for log-in procedures –without going back to the client' computer assistant in logging- in where the SilentBanker install itself in its browser. This similar-to sell phone technique was a smart way to avoid the user browser usage.

Bearing in mind that such e-Criminals' intelligence hackers will continue to innovate more capable attack tools -especially in money-making businesses and whatever security developer tried to invent more systems security problems will not stop nor disappear despite of all tries to! And as the new innovative trends attacks specification the react innovative solution will be recommend.

References

[1] Adhikari, R. 2008. Beating Online Fraud with A Phone Latest Effort To Revisit The Authentication Security Token, InternetNews.com, Copyright 2011 QuinStreet Inc., Available at: <http://www.internetnews.com/mobility/article.php/3750501/Beating-Online-Fraud-With-a-Phone.htm>, Accessed in: [Jan 28th, 2011].

[2] ZDNet website Available at: <http://www.zdnet.com/topics/web+application+security> Accessed in: [May 14th, 2011].

[3] Arnold, B.; Chess, D.; Morar, J.; Segal, A.; and Swimmer, M. 2009. An Environment for Controlled Worm Replication and Analysis.

Available at: <http://www.verisign.com.au/whitepapers/enterprise/ecommerce/infra5.shtml>

Accessed in: [Jan 20th, 2011].

[4] Federal Aviation Administration FAA. 2009. Review of Web Applications Security and Intrusion Detection in Air Traffic Control Systems. Report no.: FI-2009-049.

[5] Krebs, B. 2010. N.Y. Firm Faces Bankruptcy from \$164,000 E-Banking Loss. KrebsOnSecurity, Online Crime Investigations.

Available at: <http://www.krebsonsecurity.com/2010/02/n-y-firm-faces-bankruptcy-from-164000-e-banking-loss/>

Accessed in: [Jan 26th, 2011].

[6] Theerthagiri, D. 2009. Reversing Malware: A detection Intelligence with In-depth SecurityAnalysis. Institutionen för systemteknik, Department of Electrical Engineering, Final Thesis; Linköpings universitet SE-581 83 Linköping, Sweden, LITH-ISY-EX--09/4291—SE,

Available at:

http://cipherstormgroup.com/research/cswp/reversing_malware_detection_intelligence.pdf

[7] Reed, B. 2008. New Trojan Intercepts Online Banking Information. NetworkWorld, PCWorld Communications, Inc.

Available at:
http://www.pcworld.com/article/141364/new_trojan_intercepts_online_banking_information.html Accessed in: [Feb. 5th, 2011].

[8] Maguire, J.R. and Miller, H.G. 2010. Web-Application Security: From Reactive to Proactive. © Copyright 2011 IEEE Computer Society, Report Number: 1520-9202/10.

[9] Virus Spyware Removal Guide, 2010. Remove Trojan SilentBanker B. © Cleanpcguide.com,

Available at: <http://www.cleanpcguide.com/remove-trojan-silentbanker-b-how-to-remove-trojan-silentbanker-b/>

Accessed in: [Feb., 5th, 2011].

[9] Grossman, J. 2001. Web Application Security; The Land that Information Security Forgot. BlackHat Europe 2001, WhiteHat Security © 2001WhiteHat Security, Inc.

[10] Gutiérrez, C.^{a,b}; Rosado, D.G.^a; Fernández-Medina, E.^a 2009. The Practical Application Of A Process For Eliciting And Designing Security In Web Service Systems. Information and Software Technology 51 (2009) 1712–1738.

^aDepartment of Information Technologies and Systems, ALARCOS Research Group– Institute of Information Technologies and Systems, University of Castilla-La Mancha, Ciudad Real, Spain; ^b Correos Telecom, Madrid, Spain.

[11] Sima, C. and Liu, V. 2011. Effective Controls for Attaining Continuous Application Security throughout the Web Application Development Life Cycle.
http://voices.washingtonpost.com/securityfix/2009/10/fbi_cyber_gangsstole_40mi.html

Accessed in: [Jan, 26th, 2011].

[12] Cremers, A.B.; Spalka, A. and Langweg, H. 2010. IFIP/Sec'01: Protecting the Creation of Digital Signatures with Trusted Computing Platform Technology against Attacks by Trojan Horse Programs. Department of Computer Science III, University of Bonn, Germany.

[13] IEEE Security & Privacy. 2008. Web Application Security Assessment Tools. Published by the IEEE Computer Society 1540-7993/06,

Available at: <http://www.computer.org/security>.

[15] McAfee Labs. 2009. PWS-Banker.cz -a good comprehensive description given by McAfee security about this sample (sdra64.exe), they named this SilentBanker as PWS-Banker.cz .

Available at: <http://www.mcafee.com/threat-intelligence/malware/default.aspx?id=157491>,

Is also available at: http://vil.nai.com/vil/content/v_157491.htm,

Accessed in: [Feb., 6th, 2011].

[16] Boardman, K. 2004. A Critical Analysis of Electronic Commerce Security Measures. Short Paper Submitted in partial fulfillment of the requirements for the Degree of Bachelor of Science (Honors) in Computer Science and Information Systems at Rhodes University.

[17] Zhu, D. 2002. Security Control in Inter-Bank Fund Transfer", Journal of Electronic Commerce Research, Vol. 3, No. 1, 2002, Page 15.

[18] Verisign. 2004. Building the Infrastructure for secure Electronic Commerce. Published in 2004.

Available at:
<http://www.verisign.com.au/whitepapers/enterprise/ecommerce/infra5.shtml>

Accessed in: [May 10th, 2004].

[19] Spalka, A.; Armin, B.; Cremers, and Langweg, H. 2002. Trojan Horse Attacks On Software For Electronic Signatures. Department of Computer Science III, University of Bonn, Germany, Informatica 26 (2002) 191–203.

[20] Ghosh, A. 1998. *E-Commerce Security: weak links, best defences*, Wiley Computer Publishing, Canada.

[21] Entrust. 2010. Defeating Man-in-the-Browser: How to Prevent the Latest Malware Attacks against Consumer & Corporate Banking. Entrust 24002/3-10,

Available at: <http://www.entrust.com>.

Accessed in: [Feb, 12th, 2011].

[22] Krebs, B. 2009. FBI: Cyber Crooks Stole \$40MM from US Small, mid-sized Firms. The Washington Post: Security Fix,

Available at:
http://voices.washingtonpost.com/securityfix/2009/10/fbi_cyber_gangsstole_40mi.html

Accessed in: [Jan, 26th, 2011].

[23] Sarbanes-Oxley Act of 2002.

Available at: <http://www.soxlaw.com>

[24] The Federal Information Security Management Act of 2002,

Available at: <http://csrc.nist.gov/groups/SMA/fisma>,

Accessed in: [Feb. 5th, 2011].

[25] Schaufler, C. and Schiller, F.V. 2009. Chapter 9: Banking and Bookkeeping: Computers are not (yet?) Capable of being reasonable any more than is a Second Lieutenant. Security Engineering: A Guide to Building Dependable Distributed Systems.

[26] Reshef, E. and Bar-Gad, I. 2000. Web Application Security. TISC 2000, Perfecto Technology- the Web Application Security Company.

[27] Kashiwagi, Y. 2008. PHP. University of Texas – Austin. <http://msdn.microsoft.com/en-us/library/ff647073.aspx>

[28] Standing, C. 2002. Methodologies for Developing Web Applications. Elsevier- Information and Software Technology 44 (2002) 151-159,

Available at: <http://www.elsevier.com/locate/infsof>

Accessed in: [Nov.15th, 2010].

[28] Delgado¹ O. Fúster-Sabater¹, Sierra² A. 2008. Analysis of New Threats to Online Banking Authentication Schemes. J.M., ACTAS DE LA X RECSI, SALAMANCA, DELGADO et al.: ANALYSIS OF NEW THREATS TO ONLINE BANKING.

[29] Symantec Corporation. 2007. Symantec Internet Security Threat Report Trends for July–December 06. Copyright © 2007 Symantec Corporation. 03/07 12078591. Available at: <http://www.verisign.com.au/whitepapers/enterprise/ecommerce/infra5.shtml>

[30] CENZIC Survey. 2009. Making Web Applications Hacker Proof.

[31] Dimitris M. Karakoidas, Vassilios, Spinellis, and Diomidis, 2009. Web Application Security: Fortifying Applications Against Xpath Injection Attacks. Athens University of Economics and Business.

[32] Currie, M.W. 2009. In-The-Wire Authentication: Protecting Client-Side Critical Data Fields In Secure Network Transactions. Ziliant Systems, Durban, South Africa, © IEEE.

[33] Ford, M. 1998. Identity Authentication and 'E-Commerce' in the Journal of Information. Law and Technology (JILT), Issue 3.

Available at: <http://andytson.com/blog/2009/07/php-public-key-cryptography-using-openssl/>

[34] ValidSoft, 2007. Are your Electronic Transactions Safe?. © ValidSoft Ltd. 2003–2008.

[35] Radha, V. 2004. Preventing Technology Based Bank Frauds” in Journal of Internet Banking and Commerce. vol. 9, no.1.

[36] Mannan, M. 2009. Authentication and Secured Personal Information in an Un trusted Internet. A thesis submitted to the Faculty of Graduate Studies and Research in partial fulfillment of the requirements for the degree of Doctor of Philosophy- School of Computer Science at Carleton University, Ottawa, Ontario, USA.

[37] Trojan and Virus Research News. Silentbanker Reloaded. ThemeShark.com.

Available at: <http://www.pctrojan.com/content/289-silentbanker-reloaded>

Accessed in: [May 30th, 2011].

[38] Microsoft. 2011. Research Trojan Downloader: Win32/Silentbanker.A. Malware Protection Center, Threat Research and Response, © 2011 Microsoft Corporation.

Available at: <http://www.microsoft.com/Security/portal/Threat/Encyclopedia/Entry.aspx?Name=TrojanDownloader%3AWin32%2FSilentbanker.A&ThreatID=-2147366907>

Accessed in: [May 30th, 2011].

[39] Certification course for Information System Security Professionals CISSP. 2000. Security Architecture and Models Module. Security Models, ISC2, Finland, 30.3.2000.

[41] Masalin, S. 2001. Software Security Design and Testing. A Master Thesis in Industrial Engineering and Management. real projects in Nokia Networks Ltd.

[42] Meier J.D. Mackman, A. Dunner, M. Vasireddy, S. Escamilla, R. and Murukan, A. 2003. Improving Web Application Security: Threats and Countermeasures: How To Create a Custom Encryption Permission”, msdn, Microsoft Pattern and Practice, © Microsoft Corporation.

Available at: <http://msdn.microsoft.com/en-us/library/ff647073.aspx>

Accessed in: [May 30th, 2011].

[43] Thompson, A. 2009. PHP public key cryptography using OpenSSL. Webtatic.com, © Andy Thompson 2009 – 2011.

Available at: <http://andytson.com/blog/2009/07/php-public-key-cryptography-using-openssl/>

Accessed in: [June 01st, 2011].

[44] Murchu, L.O. 2009. Trojan.Silentbanker Decryption. © Symantec Corporation

Available at: <http://www.symantec.com/connect/blogs/trojansilentbanker-decryption>

Accessed in: [May 30th, 2011].

[45] 2008. How to Make Strong Encryption Easy to Use. BACKBLAZE, Copyright 2011.

Available at: <http://blog.backblaze.com/2008/11/12/how-to-make-strong-encryption-easy-to-use/>

Accessed in: [June 01st, 2011].

[46] Szydowski, M. Kruegel, Ch. and Kirda, E. 2007. Secure Input for Web Applications. Secure Systems Lab, Technical University Vienna, Vienna, Austria.

[47] Ahuja, V. 1997. *Secure Commerce on the Internet*. AP Professional. London.

Available at:

http://cipherstormgroup.com/research/cswp/reversing_malware_detection_intelligence.pdf

Appendixes

Appendix A: SilentBanker Installation Codes

□ %SysDir%\lowsec\local.ds – configuration file

□ % SysDir%\lowsec\user.ds – log file

□ HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Run\ "userinit" = "%System%\ sdra64.exe"

□ KEY_USERS\DEFAULT\Software\Microsoft\Windows\CurrentVersion\Run\ "userinit" = "%System%\ sdra64.exe"

It alters the following registry entries so that it executes every time Windows starts:

□ HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\WindowsNT\CurrentVersion\Winlogon\ "Userinit" = "%System%\userinit.exe, System%\sdra64.exe"

Next the listed processes are injected by the malicious code:

□ WINLOGON.EXE

□ SVCHOST.EXE

”

Appendix B: “Algorithm involved in initialising, communicating in and ending a SSL session between a client and a server based on Gosh” [1998].

“Below is listed the algorithm involved in initialising, communicating in and ending a SSL session between a client and a server based on Gosh (1998) [31].

1. Client Hello: Contains a suite of secure protocols that the client browser supports and a random challenge string generated by the browser. The challenge string is unique to the session and will be used at the end of the initialization process to make sure the secure channel has been established. The suite of secure protocols consists of key exchange algorithms for agreeing to a private session key, private key encryption protocols for transaction confidentiality, and hashing algorithms for data integrity.

2. Server Hello : Contains the servers X.509 i standard certificate, an acknowledgement that the server can support the protocols requested by the client and a random connection identifier used, as with the random challenge string, at the close to determine if the protocol has been set up.

3. The server’s certificate will then be authenticated by the client’s web browser. The client will then generate a master secret to be shared between the client and server. This master secret serves as a seed to generate a number of keys used for both symmetric encryption and data integrity. This master secret is encrypted with the server’s public key and sent to the server.

After this public key encryption is no longer necessary for this session and private key algorithms such as RC2 (40 Bit encryption)

and RC4 (128 Bit encryption) can be used to secure subsequent messages. From the master secret both server and client will generate to identical symmetric key pairs.

One key pair is used to encrypt outgoing messages from the client and i X.509 is a standard for PKI that specifies standard formats for digital certificates. Decrypt incoming messages to the server. In other words the clients outgoing write key is the same as the servers incoming read key.

4. Client finish: Client encrypts the server's random connection ID with client write key. The server will know the connection is set up if this decrypts to the same as the original connection ID.

5. Server Finish: Server encrypts the clients challenge string with the servers write key. The client then decrypts this with the clients read key and compared it to the original challenge string. Now both client and server know the connection has been set up.

5.2.4 Secure Payments and SET

There exist many payment schemes that use a variety of payment protocols and implementations to provide secure payment services. Ghosh (1998) indicates that the key difference between secure payment protocols and secure web sessions is that secure payment protocols provide a method for guaranteeing that merchants receive payments while keeping payment details such as credit card number confidential. Secure web sessions however leave payment details up to the merchant. An advantage of secure payment protocols is that credit card details are not available to the merchant and are therefore safeguarded from a potential unsafe merchant. Examples of secure payments protocols include the iPK protocols by IBM and Secure Electronic Transaction (SET) [31].

SET is a technical standard for secure payments over the Internet that focuses on credit cards and was developed by MasterCard and Visa. According to Ghosh (1998) SET does not specify actual implementation and therefore does not specify the ordering process, the payment method selection and the platform or security procedures needed to secure the SET client and host machines. SET does however specify the following requirements [31]:

□□Confidentiality: SET is only concerned with the confidentiality of payment information such as credit card numbers and is not concerned with order information. The securing of payment details is achieved through encryption.

If confidentiality of order details was needed SET could conceptually use the lower level SSL to create a secure channel thereby encrypting the entire web session.

□□Data integrity: Digital signatures are used to guard against data corruption or malicious tampering of data [39].

□□Client Authentication: Ahuja (1997) indicates that the client must be authenticated to be a legitimate user of a valid bank-card account number [39].

This can be achieved by a digital certificate issued to the client by the credit card issuer or by digital envelopes which contain the client digital signature.

□□Merchant Authentication: The client must have a method of verifying that the merchant has a relationship with a banking institution that permit the merchant to accept bank-card payments [39]. This is achieved through merchant digital certificates. The following is an example of the steps involved in a typical SET transaction based on Ghosh [1998] [31]:

1. Consumer sends request for transaction to merchant.
2. Merchant acknowledges request. The consumer and merchant exchange their digital signatures in these first 2 steps.
3. Consumer digitally signs a message digest of the order and encrypts the credit card number. The digital signature can be used by the merchant to provide authentication, non-repudiation and data integrity. The card number is encrypted with the merchant's bank's public key in order to hide the number from the merchant.
4. The merchant sends the purchase amount to be approved and the credit card number to the merchant bank. The merchant's bank then uses traditional backend structures to check credit.
5. The approval or denial is sent back to the merchant.
6. The merchant confirms the purchase with the consumer stating that the request has been approved or denied.
7. The consumer can then request the status of the order i.e.: if the merchant has received payment from the bank the merchant can start delivering the product.
8. The merchant can respond to the status enquiry – the merchant only wants to release the product once payment has been received and can report back on the status of the order. If the order is processed in batch cycles instead of real time the order payment may take longer to receive and therefore the order longer to deliver.
9. The merchant requests payment from the bank (in real time or batch cycles).
10. The bank will send confirmation of the transfer from the consumers credit card account to the merchant.

It can be seen that in the SET transaction payment of the merchant is included in the transaction. This differs from the SSL transaction which is only concerned with securing the channel and does not consider payments.

5.2.5 Pseudo Card Numbers

Pseudo credit card numbers are card numbers that are non permanent and can be used once only. Users are able to use a onetime credit card number for each purchase made online and therefore do not have to submit their real credit card numbers online.

According to Clark [2001] the purchases made by the pseudo credit card numbers are recorded against the user's real credit card number and once a purchase has been made with the number it expires and any attempts to use that number are rejected. Clark [2001] identifies pseudo credit card numbers as the easiest fraud reduction system to implement. Benefits offered to the consumer include increased fraud prevention through client authentication techniques for example: entering a password before receiving a pseudo card number [40]. Benefits to the merchant identified by Clark [2001] include the fact that the entire process is transparent to the merchant, meaning that the merchant need not alter his existing systems to support the pseudo card numbers. Another benefit addresses the insecure merchant problem. Because the pseudo card numbers expire after the transaction, if a merchant stores these credit card number and a hacker gains access to them they will be of no use [40].

Another advantage is that this approach seems to be gaining acceptance with card issuers. Clark [2001] identifies American Express and MasterCard as card issuers that seem to have implemented this type of systems to a degree [40].

The following is a typical process involved when making payment with a pseudo card number based on those identified by Clark [2001]. The interaction between the cardholder, merchant, acquirer and issuer is represented in the following steps [40]:

1. Cardholder authenticates himself with the digital wallet
2. Cardholder request a pseudo card number through a digital wallet.
3. The wallet server issues a pseudo number to the cardholder.
4. Customer purchases an item using the pseudo number.
5. Merchant sends payment request to acquirer through an internet payment gateway.
6. The acquirer sends a payment authorisation message to the issuer via and inter-bank network.
7. The issuer replaces the pseudo card number with the real credit card number.
8. The transaction is authorised.
9. The issuer sends the response to the acquiring bank containing the pseudo card number not the real number.
10. The acquirer sends a response to the merchant.
11. The Merchant sends a transaction response to the consumer.””

Appendix C: “SilentBanker Installation and Web Injection

“SilentBanker Installation Analysing the SilentBanker dropper with MD5 that will install a Browser-Helper-Object (BHO) and register its payload dll into the Internet Explorer. The payload was in our case mscorewr.dll (in c:\windows\system32\ folder) as comes with a hard-coded C&C server. **Usermode hooks for attack** Once the SilentBanker Trojan is active in memory (basically when the Internet Explorer starts), it will setup export hooks, so that it gets access to all transmitted internet traffic and to much more information. As all sophisticated Trojans will hook core windows functions to compromise the system, SilentBanker Trojan hooked (or redirected) among others the following core windows functions [37]: HttpOpenRequestA/W; HttpSendRequestA/W; InternetConnectW; InternetReadFile; InternetReadFileExA/W; InternetWriteFile; and CommitUrlCacheEntryA/W. As can see, it basically hooks all Internet related functions to get access to the Internet Traffic (even though it might be encrypted with SSL or EV-SSL!), These usermode hooks enable the Trojan to do its dirty work [37].

3.2.2.3 SilentBanker Installation Method When run, TrojanDownloader: Win32/Silentbanker.A may perform the following actions [38]:

- Drops and runs a copy of itself to the %Temp% folder as 'wscnfy32.exe'
- Drops an encrypted configuration data file as <system folder>\wuasirvy.dll. This file name may vary. The configuration file contains URLs used by the downloader to download a copy of Trojan: Win32/Silentbanker.A.

- Drops a library file into the Windows folder named 'msacm32.drv' and executes it. The instructions in this file perform the main installation routine. This library file is detected as 'TrojanDownloader: Win32/Silentbanker.A.dll'.
- The component TrojanDownloader:Win32/Silentbanker.A.dll may perform the following actions:
 - Checks Internet connectivity by connecting to the domain 'google.com'
 - Searches for the dropped encrypted configuration files by one of these file names, assumed to contain the obfuscated ULRs to retrieve
 - Trojan:Win32/Silentbanker.A:rasqervy.dll
 - sdfinacs.dll
 - hidrwupd.dll
 - wuasirvy.dll
- Parses the configuration data file to get the URL required to download Trojan:Win32/Silentbanker.A

Uses the same URL information to notify the attacker about the trojan's installation information; for example, id, socks port, http port, up time, uid, version, language, etc.

3.2.2.4 HTML Web injection The SilentBanker Trojan has also the capability to inject any arbitrary HTML code into a website and it makes use of this mainly to get additional information from the user. The disturbing fact is however that this is also possible with HTTPS together with EV-SSL certificates. This way, the website looks legitimate from all angles. The URL is correct, the SSL certificate is

fine and the green bar is shown. The reason is that the website actually comes from the legitimate site; however the SilentBanker Trojan will locally inject its malicious HTML code to the site. The code depends for each financial institution and is part of the configuration file” [37].”

Appendix D: “SilentBanker Encryption/Decryption

Trojan.SilentBanker's configuration files have always been protected, ever since the first version of the Trojan that we encountered. The reason for this protection is to make it difficult to understand what the Trojan is doing, and in particular, to hide which sites the Trojan is targeting. The original version targeted over 400 banking pages. Although, the actual list of pages being targeted was only clearly visible after the protection had been removed from the configuration files [44].

In order to discover the list of sites being targeted by any version of the Trojan the protection needs to be removed from the configuration files first. The old version of the Trojan used some simple tricks to hide its configuration; however, upon inspecting the new version of the Trojan, it is immediately obvious that something has changed. The protected configuration files look very different from previous versions [44].

Here a description for the new configuration files and the steps necessary to view these files in plain text. The old protection technique was to first use character translation (a=x, b=s, c=f, etc.) on the plain text configuration files and then to compress the resulting character translated text to a smaller binary format that could be downloaded quickly [44]. Some examples of what the old configuration files looked like-I'm showing the fully decrypted configuration file format first and working backwards towards the encrypted file that was downloaded by the Trojan. This is the plain text configuration file; it contains URLs to send the stolen data to,

URLs to download updates from, and the URLs of targeted bank sites (the data shown has been sanitized to remove dangerous or targeted URLs): Figure D.1 shows an “Example of a Decrypted Config File” [44].

```
[dfgdf]
bg1=blahblah.com/eur/index.php
[nbmx]
bg1=blahblah.com/eur/index.php
[kjew]
bg1=blahblah/download/blah.exe
[qweq]
bg1=somebank.com
bg2=anotherbank.net
bg3=targettedsite.com
bg4=otherbank.cz
bg5=othersite.com
```

Figure D.1: “Example of a Decrypted Config File” [44].

Each new section of the configuration file starts with. So, the first section is [dfgdf]. Inside each section there is a list of data for the Trojan to use. Each string of data to be used is stored after an identifier (e.g. bg1=, bg2= , bg3=, etc.) [44].

Presented below is the text configuration file after character translation has been carried out. The character translation is only used on strings that are preceded by an "=" sign. An example of the translation is [44]:

"b"	was	changed	to	"o"
"l"	was	changed	to	"y"
"a"	was	changed	to	"n"
"h"	was	changed	to	"u",

So that "blah" becomes "oynu".

(b=o,l=y,a=n,h=u, .=8,c=p,o=b,m=z,/=#,e=r,u=h,r=e,/=#,j=v,n=a,d=q,e=r,x=k, .=8,p=c,h=u,p=c). Figure D.2 and Figure D.3 shows “The Config File after Character Translation”.

```
[dfgdf]
bg1=oynuoynu8pbz#rhe#vaqrk8cuc
[nbmx]
bg1=oynuoynu8pbz#rhe#vaqrk8cuc
[kjew]
bg1=oynuoynu#qbjaybnq#oynu8rkr
[qweq]
bg1=fgfronax8pbz
bg2=nabtureonax8arg
bg3=gnetrggrqfvgr8pbz
bg4=bgureonax8pm
bg5=bgurefvgr8pbz
```

Figure D.3: “The Config File after Character Translation” [44].

The file shown above is not exactly what the Trojan downloads though, what the Trojan downloads is a compressed binary configuration file, shown in figure D.4. We can see that the compressed binary format starts with FF and on the right we can still make out some of the character translated text, too. The fact that the file starts with FF is one tell tale sign that this is a Trojan.Silentbanker configuration file, and the Trojan also stores its configuration files in files named [9-11 digits].cpx.

```
FF5B6466 6764665D 0DFF0A62 67313D6F ỳ[dfgdf].ỳ.bg1=ó
796EFF75 6F796E75 387062FF 7A237268 ynyuoynu8pbýz#rh
65237661 FF71726B 38637563 0D3F0A5B e#vaýqrk8cuc.?.[
6E626D78 F4FF060F EF6B6A65 771C042E nbmxóý...ikjew...
```

Figure D.4: “Compressed Binary Config File” [44].

The current Trojan.Silentbanker configuration files look different though- as shown in figure D.5. As we can see, there is no 0xFFh at the start. So, the Trojan must be using some other type of encryption on these new configuration files.

```
14C0234C 0171AFDA 3A52D25C 8FD2E7BD .À#L.ç`Ú:RÒ\|Òç½
67A2B5E3 537BA7AA B17369DF 2CBACE4C gçµãS{S²±siB,ºÍL
427945A7 9533737A F7EA7CF5 E7032A4C ByES|3sz÷é|öç.*L
800FDB43 13CFB03E E203A685 332A7975 |.ÛC.Í`>á.!!3*yu
9FA25ABA E6165FD2 C5BE0FEE AB753456 |çZºæ. ÖÁ¾.i<<u4V
FA4897F2 4E3E5294 194B919D F66F6E6D úH|òN>R|.K`|öonm
```

Figure D.5: “New Config File Format” [44].

After some analysis of the Trojan we come to the routine that described in figure D.6 [44].

```
mov     edi, [eax]
mov     eax, [eax+4]
mov     [ebp+arg_0], 0C6EF3720h
mov     [ebp+var_C], edi
mov     [ebp+var_10], eax
mov     [ebp+arg_4], 20h

loop_Decrypt:
mov     edi, [ebp+var_4] ; CODE
xor     edi, edx
add     edi, [ebp+var_8]
mov     eax, edx
shr     eax, 5
xor     eax, [ebp+arg_0]
mov     ebx, edx
add     edi, eax
shl     ebx, 4
add     ebx, edi
mov     edi, [ebp+var_C]
sub     ecx, ebx
mov     eax, ecx
xor     edi, ecx
add     edi, [ebp+var_10]
shr     eax, 5
xor     eax, [ebp+arg_0]
add     [ebp+arg_0], 61C88647h
mov     ebx, ecx
shl     ebx, 4
add     edi, eax
add     ebx, edi
sub     edx, ebx
dec     [ebp+arg_4]
jnz     short loop_Decrypt
```

Figure D.6: “Decryption Loop” [44].

This is the decryption routine for the configuration files. Now we can start to make some sense of the configuration file shown above. Searching online for the constants used in the code above - namely C6EF3720 and 61C88647- shows us that the code used is probably a Tiny Encryption Algorithm (TEA) encryption routine or a modified version of it. After running the decryption routine shown above, we end up with a file that looks like figure D.7. This is, in fact, the same format as the Trojan was using previously. Notice that it starts with FF. (The "x"s on the right were an IP address that has been removed.) The latest version of the Trojan has just added a layer of encryption on top of the old protection layers. Once the TEA encryption layer has been bypassed, we can decode the configuration files in exactly the same way as for the older version of the Trojan [44].

FF5B325D	0D0A3338	31FF3D78	782E7878	ÿ[2]..381ÿ=xx.xx
782EFF78	782E7878	2F0D0A7B	5B36F0F0	x.ÿxxx.xx/..{[688
34343D30	0400FD37	08033232	31323038	44=0..ÿ7..221208
D7393037	040031F0	F03230FF	3D313630	×907..18820ÿ=160

Figure D.7: “Config. File After Decryption” [44].

Joining Public and Private Keys using Symmetric Keys

Through the PHP Public Key Cryptography, using “OpenSSL” trade for data encryption and decryption with a symmetric-key system using mcrypt for PHP for managing some security aspects in both servers’ backend and frontend gathered or separately. The asymmetric-key system is needed for public-key cryptography for the necessary php OpenSSL extension documentation and systems for the implementations. Using the PHP OpenSSL extension it is fairly easy to sort out a secure system for encrypting data with one key that only can be decrypted with another; first, you need to generate your private and public keys programmatically using PHP-Next, you can load the public key, and encrypt the data when you need to get the sensitive data again, you can load the private key and decrypt, Alternatively you can use the private key to encrypt data, sign data or seal it against multiple other public keys” [43].

Appendix E: Coding

(E.1) Client Entrance-Code

```
<?
if($_POST['ok'])
{
session_start();
$UN=strtoupper($_POST['username']);
require_once("ss.php");
$upass=$_POST['pass'];
$cryptastic = new cryptastic;
$encrypted = $cryptastic->encrypt($upass, $key);
$encrypt_un=$cryptastic->encrypt($UN, $key);
$_SESSION['enpass']=$encrypted;
$_SESSION['enuser']=$encrypt_un;
header("Location: https://127.0.0.1/encryption/index2.php");
}
else
{
?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-
8" />
<title>WORLD BANK</title>
<center>

</center>
</head>
<body bgcolor="#00ccff"><br /><br /><br /><br /><br /><center>
<form action="index.php" method="post">
<table>
<tr><td><label>User Name : </label>
</td>
```

```

<td> <input type="text" name="username" value="" /></td>
</tr>
<tr><td><label> Password : </label></td>
<td> <input type="password" name="pass" value="" /></td></tr>
<tr><td><input type="submit" value="Log In" name="ok" />
</td></tr>
</table>
</form>
</body>
</html>
<?
}
?>

```

(E.2) Bank Check-Code

```

<?
    session_start();
    require_once("connect.php");
    $username=strtoupper($_COOKIE['username']);
    $pass2=$_SESSION['enpass'];
    require_once("ss.php");
    $username=$_SESSION['enuser'];
    $cryptastic = new cryptastic;
    $decpass_ser = $cryptastic->decrypt($pass2, $key);
    $decuser_ser = $cryptastic->decrypt($username, $key);
    $query1="select * from mytable where
    UserName='".$decuser_ser.'"";
    $result=mysql_query($query1);
    $row=mysql_fetch_object($result);
    $db_user=$row->UserName;
    $db_pass=$row->PassUser;
    if($db_pass==$decpass_ser)
    {
    $userid=$row->Id;
    $userid=$row->Id;
    $session_id=session_id();
    $_SESSION['ses']=$session_id;
    $_SESSION['userid']=$userid;
    $result2=mysql_query("select * from session_id_s where
    UserId='".$userid'"");
    $row2=mysql_fetch_object($result2);
    if($row2==0)

```

```

{
mysql_query("insert          into          session_id_s
VALUES('$session_id','$userid','1','0')");
echo right;
}
else
{
mysql_query("update session_id_s set trylogin='1' where
UserId='".$userid."'");
header("location : index.php");
?>
try Agian Later.
<meta content="5;http://127.0.0.1/encryption/index.php" http-
equiv="refresh">
<?
}
?>
<meta content="6;https://127.0.0.1/encryption/index3.php" http-
equiv="refresh">
<?
}
else
{
?>
wrong Username Or Password
<meta content="5;http://127.0.0.1/encryption/index.php" http-
equiv="refresh">
<?
}
/*
if($decrypted==$row->PassUser    &&    $username==$row-
>UserName)
{
}
*/
?>

```

(E.3) Admin Code

```
<?
session_start();
$_SESSION['ses'];
require_once("checkuser.php");
if(isset($_SESSION['ses']))
{
$_POST['yes'];
if($_POST['yes'])
{
require_once("ss.php");
$Cryptastic = new cryptastic;
$_POST['pname'];
$_POST['pname'] = $Cryptastic->encrypt($_POST['pname'], $key);
$_SESSION['pname'] = $_POST['pname'];
$_POST['paccount'];
$_POST['paccount'] = $Cryptastic->encrypt($_POST['paccount'], $key);
$_SESSION['paccount'] = $_POST['paccount'];
$_POST['pcc'];
$_POST['pcc'] = $Cryptastic->encrypt($_POST['pcc'], $key);
$_SESSION['pcc'] = $_POST['pcc'];
$_POST['pamont'];
$_POST['pamont'] = $Cryptastic->encrypt($_POST['pamont'], $key);
$_SESSION['pamont'] = $_POST['pamont'];
$_POST['currncy'];
$_POST['currncy'] = $Cryptastic->encrypt($_POST['currncy'], $key);
$_SESSION['currncy'] = $_POST['currncy'];
?>
<meta content="0;index4.php?pa=<? echo $_SESSION['pamont'] ?>" http-
equiv="refresh" />
<?
}
else
{
?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-
8" />
```

```

<title>WORLD BANK</title>
<center>

</center>
</head>
<body bgcolor="#00ccff"><br /><br /><br /><br /><br /><center>
<form action="index3.php" method="post">
<table>
<tr><td><label>Payee Name </label></td>
<td> <input type="text" name="pname" /></td></tr>
<tr><td><label> Payee Account </label></td>
<td><input type="text" name="paccount" /></td></tr>
<tr><td> <label>Payee Cart Code</label></td><td><input
type="password" name="pcc" maxlength="4"/></td></tr>
<tr><td><label>Payee Amont</label></td><td><input type="text"
name="pamont" /></td>
<td>
<select name="currncy" >
<option value="US">$ US</option>
<option value="RS">RS</option>
<option value="JO">JO</option>
</select>
</td></tr>
<tr><td><input type="submit" value="Send" name="yes" />
</td>
</tr>
</table>
</form>
<a href="logout.php" >LogOut</a>
</body>
</html>
<?
}
}
else
{
?>
<meta content="0;index.php" http-equiv="refresh" />
<?
}
?>

```

(E.4) Decryption and verification Code

```
<?
require_once("checkuser.php");
session_start();
$session_id=$_SESSION['ses'];
if(isset($session_id))
{
require_once("ss.php");
require_once("connect.php");
$cryptastic = new cryptastic;
$name=$_SESSION['pname'];
$account=$_SESSION['paccount'];
$pcc=$_SESSION['pcc'];
$pamont=$_SESSION['pamont'];
$currency=$_SESSION['currncy'];
$currency_d=$cryptastic->decrypt($currency,$key);
$name_d=$cryptastic->decrypt($name, $key);
```

(E.5) Decryption Code

```
<?
$ok=$_POST['ok'];
if($ok)
{
session_start();
$UN=strtolower($_POST['username']);
require_once("../ss.php");
$upass=$_POST['pass'];
$cryptastic = new cryptastic;
$encrypted = $cryptastic->encrypt($upass, $key);
$encrypt_un=$cryptastic->encrypt($UN, $key);
$_SESSION['adminpass']=$encrypted;
$_SESSION['adminuser']=$encrypt_un;
header("Location:
https://127.0.0.1/encryption/admins/index2.php");
}
else
{
?>
```

```

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-
8" />
<title>WORLD BANK</title>
<center>

</center>
</head>
<body bgcolor="#00ccff"><br /><br /><br /><br /><br /><center>
<form action="index.php" method="post">
<table>
<tr><td><label>User Name : </label>
</td>
<td> <input type="text" name="username" value="" /></td>
</tr>
<tr><td><label> Password : </label></td>
<td> <input type="password" name="pass" value=""/></td></tr>
<tr><td><input type="submit" value="Log In" name="ok" />
</td>
</tr>
</table>
</form>
</body>
</html>
<?
}
?>

```

(E.6) Admin Coding

```

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-
8" />
<title>WORLD BANK</title>
<center>

```

```


</center>
</head>
<body bgcolor="#00ccff"><br /><br /><br /><br /><br /><center>
<table align="center" cellspacing="50">
<tr><Td><a href="addnew.php">Add New User</a></Td>
<td><A href="show.php">Show Transfare</A></td>
<td><a href="editdel.php">(Edit/Delete) User</a></td>
</tr>
</table>
</body>
</html>

```

(E.6.1) Add New User Code

```

<?
session_start();
require_once("../connect.php");
require_once("../ss.php");
$username=$_SESSION['addnewuser'];
$pass2=$_SESSION['addnewpass'];
$Cryptastic = new cryptastic;
echo $username=$Cryptastic->decrypt($username,$key);
echo $pass2=$Cryptastic->decrypt($pass2,$key);
mysql_query("insert          into          mytable
VALUES('$username','$pass2')") or die(mysql_error());
?>

```

(E.6.2) e-money Transferring Process

```

<?
require_once("../connect.php");
require_once("../ss.php");
$Cryptastic = new cryptastic;
$query="select * from trans";
$result=mysql_query($query);
?>

```



```

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-
8" />
<title>WORLD BANK</title>
<center>

</center>
</head>
<body bgcolor="#00ccff"><br /><br /><br /><br /><br /><center>
<table cellspacing="" cellpadding="50" align="center" border="1">
<tr bgcolor="#00CC99"><td>Payee Name </td>
<td> Payee Account </td>
<td>Payee Amont</td>
</tr>
<?
while($row=mysql_fetch_object($result))
{
echo "<tr><td>";
echo $row->pname."</td><td>";
echo $row->paccount."</td><td>";
echo $row->pamont."</td></tr>";
}
?>
</table>
</body>
</html>

```

(E.6.3) Add/Delete Users Code

```

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-
8" />
<title>WORLD BANK</title>
<center>

```

```


</center>
</head>
<body bgcolor="#00ccff"><br /><br /><br /><br /><br /><center>
<table align="center" cellspacing="50">
<tr><Td><a href="addnew.php">Add New User</a></Td>
<td><A href="show.php">Show Transfare</A></td>
<td><a href="editdel.php">(Edit/Delete) User</a></td>
</tr>
</table>
</body>
</html>

```

(E.6.4) The Lock Code

```

<?
class cryptastic {
/** Encryption Procedure
*
* @param mixed msg message/data
* @param string k encryption key
* @param boolean base64 base64 encode result
*
* @return string iv+ciphertext+mac or
* boolean false on error
*/
public function encrypt( $msg, $k, $base64 = false ) {
# open cipher module (do not change cipher/mode)
if ( ! $td = mcrypt_module_open('rijndael-256', '', 'ctr', '' ) )
return false;
$msg = serialize($msg); # serialize
$iv = mcrypt_create_iv(32, MCRYPT_RAND); # create iv
if ( mcrypt_generic_init($td, $k, $iv) !== 0 ) # initialize buffers
return false;
$msg = mcrypt_generic($td, $msg); # encrypt
$msg = $iv . $msg; # prepend iv
$mac = $this->pbkdf2($msg, $k, 10, 32); # create mac
$msg .= $mac; # append mac
mcrypt_generic_deinit($td); # clear buffers
mcrypt_module_close($td); # close cipher module
if ( $base64 ) $msg = base64_encode($msg); # base64 encode?

```

```

return $msg;                                # return iv+ciphertext+mac
}
/** Decryption Procedure
*
* @param string msg output from encrypt()
* @param string k encryption key
* @param boolean base64 base64 decode msg
*
* @return string original message/data or
* boolean false on error
*/
public function decrypt( $msg, $k, $base64 = false )
if ( $base64 ) $msg = base64_decode($msg);      # base64
decode?
# open cipher module (do not change cipher/mode)
if ( ! $td = mcrypt_module_open('rijndael-256', '', 'ctr', '' ) )
return false;

$iv = substr($msg, 0, 32);                    # extract iv
$mo = strlen($msg) - 32;                      # mac offset
$em = substr($msg, $mo);                      # extract mac
$msg = substr($msg, 32, strlen($msg)-64);     # extract ciphertext
$mac = $this->pbkdf2($iv . $msg, $k, 10, 32);  # create mac
if ( $em !== $mac )                           # authenticate mac
return false;
if ( mcrypt_generic_init($td, $k, $iv) !== 0 ) # initialize buffers
return false;
$msg = mdecrypt_generic($td, $msg);           # decrypt
$msg = unserialize($msg);                   # unserialize
mcrypt_generic_deinit($td);                  # clear buffers
mcrypt_module_close($td);                    # close cipher module
return $msg;                                  # return original msg
}
/** PBKDF2 Implementation (as described in RFC 2898);
*
* @param string p password
* @param string s salt
* @param int c iteration count (use 1000 or higher)
* @param int kl derived key length
*
* @param string a hash algorithm
* * @return string derived key

```

```

*/      public function pbkdf2( $p, $s, $c, $kl, $a = 'sha256' ) {
$hl = strlen(hash($a, null, true)); # Hash length
$kb = ceil($kl / $hl);           # Key blocks to compute
$dk = "";                        # Derived key
# Create key
for ( $block = 1; $block <= $kb; $block ++ ) {
# Initial hash for this block
$ib = $b = hash_hmac($a, $s . pack('N', $block), $p, true);
# Perform block iterations
for ( $i = 1; $i < $c; $i ++ )
# XOR each iterate
$ib ^= ($b = hash_hmac($a, $b, $p, true));
$dk .= $ib; # Append iterated block
}
# Return derived key of correct length
return substr($dk, 0, $kl);
}
}
$pass = '10';
$salt = '10';
$cryptastic = new cryptastic;
$key = $cryptastic->pbkdf2($pass, $salt, 10, 32);
?>

```

