Volume 8 • Issue 1 • February 2014

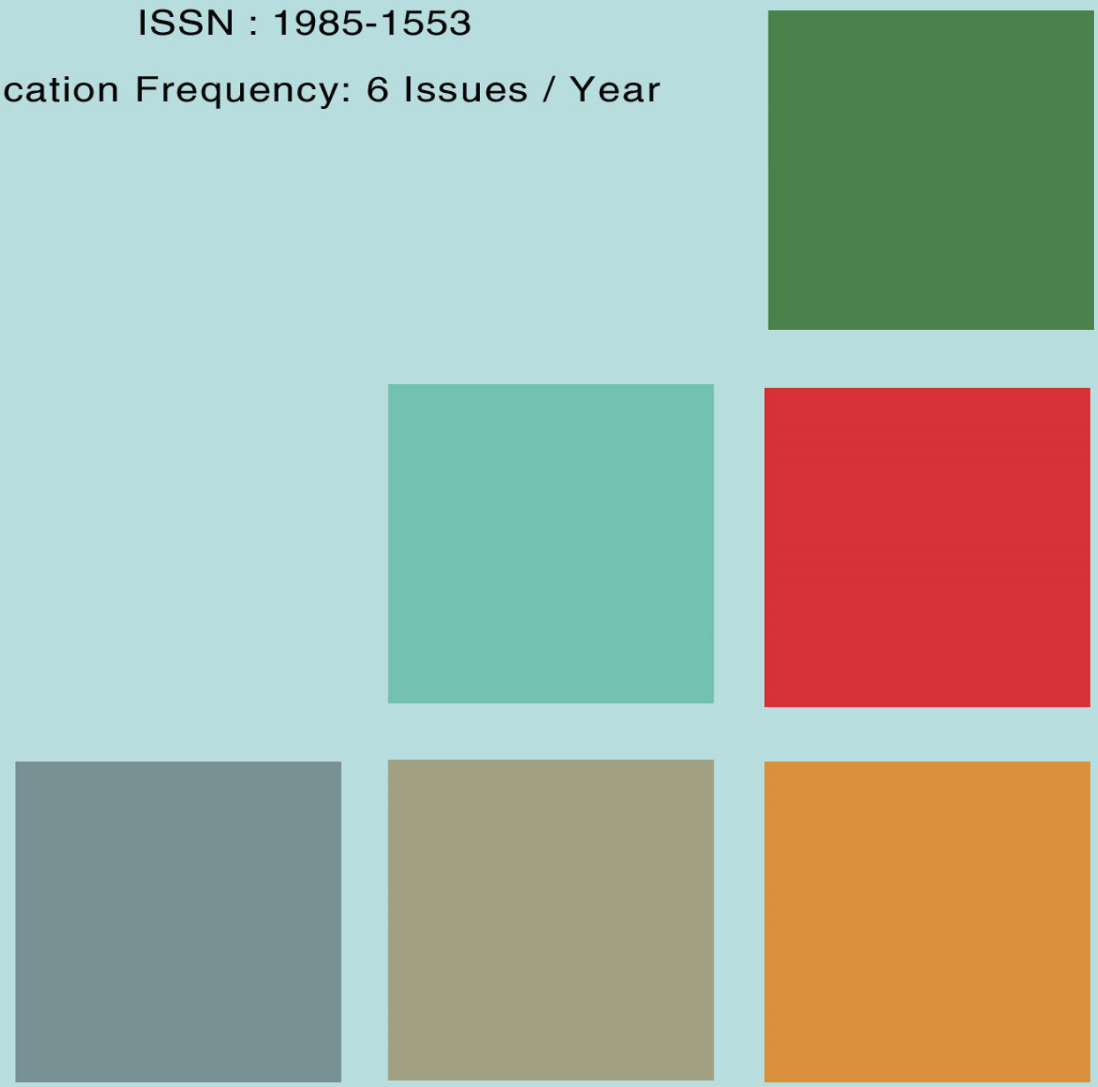
Editor-in-Chief
Dr. Chen-Chi Shing

INTERNATIONAL JOURNAL OF

COMPUTER SCIENCE AND SECURITY (IJCSS)

ISSN : 1985-1553

Publication Frequency: 6 Issues / Year



CSC PUBLISHERS
http://www.cscjournals.org

INTERNATIONAL JOURNAL OF COMPUTER SCIENCE AND SECURITY (IJCSS)

VOLUME 8, ISSUE 1, 2014

EDITED BY
DR. NABEEL TAHIR

ISSN (Online): 1985-1553

International Journal of Computer Science and Security is published both in traditional paper form and in Internet. This journal is published at the website http://www.cscjournals.org, maintained by Computer Science Journals (CSC Journals), Malaysia.

IJCSS Journal is a part of CSC Publishers
Computer Science Journals
http://www.cscjournals.org

INTERNATIONAL JOURNAL OF COMPUTER SCIENCE AND SECURITY (IJCSS)

Book: Volume 8, Issue 1, February 2014
Publishing Date: 11 - 2 - 2014
ISSN (Online): 1985 -1553

IJCSS Journal is a part of CSC Publishers
http://www.cscjournals.org

Published in Malaysia

Typesetting: Camera-ready by author, data conversation by CSC Publishing Services – CSC Journals, Malaysia

CSC Publishers, 2014

EDITORIAL PREFACE

This is First Issue of Volume Eight of the International Journal of Computer Science and Security (IJCSS). IJCSS is an International refereed journal for publication of current research in computer science and computer security technologies. IJCSS publishes research papers dealing primarily with the technological aspects of computer science in general and computer security in particular. Publications of IJCSS are beneficial for researchers, academics, scholars, advanced students, practitioners, and those seeking an update on current experience, state of the art research theories and future prospects in relation to computer science in general but specific to computer security studies. Some important topics cover by IJCSS are databases, electronic commerce, multimedia, bioinformatics, signal processing, image processing, access control, computer security, cryptography, communications and data security, etc.

The initial efforts helped to shape the editorial policy and to sharpen the focus of the journal. Started with Volume 8, 2014, IJCSS appears with more focused issues. Besides normal publications, IJCSS intend to organized special issues on more focused topics. Each special issue will have a designated editor (editors) – either member of the editorial board or another recognized specialist in the respective field.

This journal publishes new dissertations and state of the art research to target its readership that not only includes researchers, industrialists and scientist but also advanced students and practitioners. The aim of IJCSS is to publish research which is not only technically proficient, but contains innovation or information for our international readers. In order to position IJCSS as one of the top International journal in computer science and security, a group of highly valuable and senior International scholars are serving its Editorial Board who ensures that each issue must publish qualitative research articles from International research communities relevant to Computer science and security fields.

IJCSS editors understand that how much it is important for authors and researchers to have their work published with a minimum delay after submission of their papers. They also strongly believe that the direct communication between the editors and authors are important for the welfare, quality and wellbeing of the Journal and its readers. Therefore, all activities from paper submission to paper publication are controlled through electronic systems that include electronic submission, editorial panel and review system that ensures rapid decision with least delays in the publication processes.

To build its international reputation, we are disseminating the publication information through Google Books, Google Scholar, Directory of Open Access Journals (DOAJ), Open J Gate, ScientificCommons, Docstoc and many more. Our International Editors are working on establishing ISI listing and a good impact factor for IJCSS. We would like to remind you that the success of our journal depends directly on the number of quality articles submitted for review. Accordingly, we would like to request your participation by submitting quality manuscripts for review and encouraging your colleagues to submit quality manuscripts for review. One of the great benefits we can provide to our prospective authors is the mentoring nature of our review process. IJCSS provides authors with high quality, helpful reviews that are shaped to assist authors in improving their manuscripts.

Editorial Board Members
International Journal of Computer Science and Security (IJCSS)

EDITORIAL BOARD

Assistant Professor Vishal Bharti
Maharishi Dayanand University
India

Dr. Parvinder Singh
University of Sc. & Tech
India

Assistant Professor Vishal Bharti
Maharishi Dayanand University,
India

TABLE OF CONTENTS

Volume 8, Issue 1, February 2014

Pages

Efficient Coercion Resistant Public Key Encryption

Maged Hamada Ibrahim *mhii72@gmail.com*
Department of Electronics, Communications and Computers Engineering,
Faculty of Engineering, Helwan University
1, Sherif St., Helwan, Cairo; P.O. 11792, Egypt

Abstract

The notion of deniable encryption has been known in the literature since its introduction in [1] as coercion resistant encryption schemes that allow the user (sender and/or receiver) to escape a coercion attempted by a coercive adversary. The schemes allow the user to open fake message(s) to the coercer that when verified gives the same ciphertext as the true message, while the receiver is always able to decrypt for the true message. In this paper we focus on sender-incoercible encryption. The contribution of this paper is two-fold. First, we introduce a new classification of services that could be provided by coercion-resistant encryption showing that all previously proposed deniable PKE schemes fall in the category of unplanned incoercible PKE assuming the user is non-collaborative and do not satisfy the requirements for deniable encryption. Then we inspect, refine and improve the sender-incoercible PKE introduced in [2]. Our new scheme achieves constant transmission rate where the size of the plaintext may be calibrated to be sufficiently large i.e. the scheme encrypts arbitrary length messages without a blowup expansion in the ciphertext while the size of the ciphertext grows linearly with the number of fake messages.

Keywords: Network Security, Coercion-resistance, Deniable Encryption, Incoercible Encryption, Receipt-freeness, Hybrid Encryption, Electronic Voting.

1. INTRODUCTION

Consider the scenario of standard public key encryption where the receiver $\mathcal{R}$'s public key is known to everyone and could be used by arbitrary many senders (potentially unknown to the receiver at the time the public key is published). For an arbitrary sender $\mathcal{S}$ to send a message m to $\mathcal{R}$ without any further interaction, he computes and sends the ciphertext c which is essentially a function of $\mathcal{R}$'s public key pk and the message m. On the reception of c, $\mathcal{R}$ is able to decrypt for m using the corresponding secret key sk. Now, assume that there is an adversary that is able to eavesdrop on the channel between $\mathcal{S}$ and $\mathcal{R}$, of course, the one-way security property of the encryption scheme prevents a polynomial time adversary from reaching the plaintext given the ciphertext in hand without knowing the secret key. However, the one-way security of the encryption function is not enough since such a weak notion does not hide the statistical properties of the plaintext message (unless the message is truly random parameter with sufficiently large bit-length). Therefore, the semantic security notion (probabilistic encryption) or equivalently, indistinguishable chosen plaintext attack (IND-CPA notion) which emphasizes on the necessity of hiding the plaintext statistical properties using random coins has been introduced in [3]. Other active attacks that could be attempted by an adversary (such as tampering with the ciphertext in a way undetectable by the receiver) have been introduced (through several notions of security) in [4, 5, 6, 7] and many others. In all these notions, incorporating random coins are essential in performing the encryption securely.

1.1 Coercion

Now, consider the situation where the adversary $\mathcal{A}$ (in addition of being able to eavesdrop on the channel between $\mathcal{S}$ and $\mathcal{R}$) has some coercive power against the sender $\mathcal{S}$. Such a *coercive adversary* cannot corrupt $\mathcal{S}$ (i.e. cannot replace $\mathcal{S}$) and hence she is weaker than a corruptive

adversary, yet she only has some coercive power against $\mathcal{S}$ which allows her to force $\mathcal{S}$ to perform as she wishes. This adversary approaches $\mathcal{S}$ commanding him to perform some actions (e.g. sending a particular vote to $\mathcal{R}$) against $\mathcal{S}$'s desire. Notice that $\mathcal{A}$ is able to record all communications sent from $\mathcal{S}$ to $\mathcal{R}$. Now, the following question arises: Does a standard public key encryption scheme allows $\mathcal{S}$ to escape such coercion? In standard PKE, $\mathcal{S}$ computes the ciphertext on the form $c = \mathcal{E}_{pk}(r, m)$ where pk is $\mathcal{R}$'s public key, r is $\mathcal{S}$'s random coins and m is the message that $\mathcal{S}$ wishes to send. Now $\mathcal{A}$ approaches $\mathcal{S}$ after transmission and asks him to reveal r and m (in practice, $\mathcal{A}$ asks $\mathcal{S}$ to deliver his laptop, hard-drive or even the whole server to her). Let m' be the message satisfactory to $\mathcal{A}$. Notice that c commits $\mathcal{S}$ to both r and m i.e. $\mathcal{S}$ is not able to come up (in polynomial time) with a different random coins r' satisfying $\mathcal{E}_{pk}(r, m) = \mathcal{E}_{pk}(r', m')$. Also, $\mathcal{S}$ cannot claim that he erased his memory (specially the random coins) since in this case, $\mathcal{A}$ assumes that $\mathcal{S}$ didn't obey her desire and takes actions against him. Therefore, if $\mathcal{S}$ is to lie safely without being suspected as a liar, the encryption scheme must allow him to open the encryption without the need to claim the erasure of any of his local randomness that are known to exist.

1.2 An Overview of Existing Adversary Models

For seeking completeness, in this subsection we review the essential models of an adversary: There are several models of a corruptive adversary: *Stationary (non-mobile) adversary*, *Mobile adversary*, *Static adversary* and *Adaptive adversary*. In a stationary adversary, the adversary may attack a number of parties (minority), and this number is assumed not to exceed a certain value (the threshold) along the life time of the private inputs.

A mobile adversary [8, 9] is able to jump from one party to the other (mobile virus attacks), collecting as much information as she can, she has the whole life-time of a secret to do so. Hence, the assumption that the adversary will not exceed a certain threshold no more holds. To withstand such type of an adversary, the parties must pro-actively renew their private inputs (cooperatively) through proactive security techniques and erase any previously shared information.

In a static adversary [10], the parties that the adversary is to corrupt are defined prior to the multiparty protocol execution, and remains unchanged during execution, that is, the adversary does not adapt her behavior during execution of the protocol whenever (for example) she finds that some party did not erase previous information after pro-actively renew her private inputs.

An adaptive adversary is a stronger adversary [10, 11]. This adversary is not only able to jump from one party to another, but she do that in a wise manner, according to her view of the communications among parties and her view of the computations of the already corrupted parties. Withstanding such type of adversaries is not an easy task especially in the existence of dishonest parties (non-erasing parties that are not trusted to erase their sensitive information).

The type of an adversary that we deal with in this paper cannot corrupt a party, yet, she has a coercive power that allows her to coerce a party to do as she wishes. This type of an adversary is known as a *coercive adversary* [1, 2, 12, 13]. The notion of *deniable* encryption deals with this type of an adversary in the sense that it allows a party to open any plaintext message that when verified gives the same ciphertext observed by this adversary.

1.3 Deniability and Adaptive Security

Adaptively secure encryption represents the tool (plug and play) to achieve adaptive security in multiparty protocols. This tool is a.k.a *non-committing* encryption. However, the term non-committing is misleading because such encryption is indeed committing as notified in [14]. This encryption is committing in the sense that an honest sender cannot later pretend that an alternate message was sent. That is, these cryptosystems are non-committing in the existence of the simulator (the ideal world, not the real world). A distinguish between deniable encryption [1, 2, 12] and non-committing (adaptively secure) encryption [14, 15, 16, 17, 18, 19] is as follows. Deniable

encryption is a true non-committing encryption which faces a type of an adversary known as a coercive adversary. This adversary is weaker than a corruptive adversary in the sense that, she cannot corrupt a party, yet, she has some power that allows her to coerce this party to do as she wishes. In deniable encryption, a sender can generate a ciphertext that appears as an encryption of many messages. Whereas, in non-committing encryption, the ciphertext that could be opened as an encryption of any message is generated by the simulator (in the ideal world) while the honest parties are still committed to the encryption. The simulator has many advantages over the sending and the receiving parties, for example, the simulator knows the public as well as the secret keys of all parties and all auxiliary information used to perform the encryption which makes non-committing encryption easier than coercion-resistant encryption. Yet, for a non-committing encryption scheme to be useful (i.e. achieves adaptive security in a multiparty environment), it must withstand corruption of the sender and the receiver simultaneously which is a strong assumption, whereas, coercion-resistant encryption could be designed under weaker assumptions (e.g. to achieve sender-only deniability or receiver-only deniability) where one of the two parties is beyond the reach of the adversary. However, when it comes to the design of a sender-and-receiver coercion-resistant encryption, it is much harder than non-committing encryption. In this paper we focus on the sender side and assume that the receiver is beyond coercion.

2. PREVIOUS WORK

The notion of deniable encryption has been introduced in [1] where a sender-incoercible public-key encryption scheme based on trapdoor permutations has been constructed. However, the scheme falls short of achieving an appropriate level of deniability, due to several reasons: i) To achieve a high level of incoercibility, the size of the ciphertext corresponding to a one bit encryption is super-polynomial and hence inefficient. ii) There is a significant probability of decryption errors. iii) A collaborative sender is able to setup the encryption in a way that allows him to later prove the decrypted message to the coercer.

The recent work in [2] introduced a scheme for a one-move sender-incoercible public-key encryption which is built using any trapdoor permutation. The scheme also allows the sender to lie safely but still the true message is provable by the sender. The communication complexity is in the order of $\mathcal{O}(k)$ bits for one bit plaintext, where k is a security parameter. To encrypt more than one bit, the scheme deviates from practicality as also notified in [20]. The same communication complexity applies to the public-key encryption scheme in [12].

3. MOTIVATIONS AND CONTRIBUTIONS

3.1 Motivations

We find that the notion of coercion-resistant encryption schemes (previously known as deniable encryption) needs to be refined from the point of view of classification of services provided, in the sense that – for example – the encryption scheme to be coercion-resistant, it is not necessarily be deniable, it is enough to allow the user to lie safely without claiming the erasure of his random inputs. The idea introduced in [2] is good as to provide an incoercible PKE scheme (not deniable PKE), however the construction was inefficient leading to high computations and communications complexity making the scheme deviates from practice as the plaintext bit-length increases [20].

3.2 Our Contributions

First, we introduce a new classification of services that could be provided by coercion-resistant encryption showing that all previously proposed schemes fall in the category of unplanned incoercible encryption and do not satisfy the requirements for deniable encryption. Then we inspect, refine and improve the sender-incoercible PKE introduced in [2]. Our new scheme achieves constant transmission rate where the size of the plaintext may be calibrated to be sufficiently large i.e. the scheme encrypts arbitrary length messages without blowup expansion in the size of the ciphertext. Also, the size of the ciphertext grows linearly with the number of fake messages.

4. COERCION-RESISTANT ENCRYPTION: DENIABLE vs. INCOERCIBLE

In this section we introduce the new classifications of services we want to escalate for coercion resistant encryption schemes. Consider – for example – the Yes/No e-voting scheme where the sender is supposed to submit his Yes or No vote against the coercer in an encrypted way to the authority. We have two cases here according to the time of coercion, either before or after transmission. In case the coercer approaches the sender before transmission, then the coercer has the chance to perform the encryption and generates the ciphertext by himself (since he knows the receiver's public-key), delivers the ciphertext to the sender forcing him to send this particular ciphertext (notice that the coercer is able to eavesdrop on the channel), consequently, the sender is trapped and has nothing to do but sending this ciphertext to the receiver. Approaching the sender before transmission traps the sender regardless of the type of encryption scheme used.

Now we are left with the case where the coercer approaches the sender after transmission that is the first contact between them is after the ciphertext has been placed on the channel. If the sender's claim that he erased all or part of his random inputs is accepted to a coercer then any standard PKE is indeed incoercible but not deniable since still the sender has the choice to open all his random inputs to the coercer (as a receipt) and prove the encrypted plaintext. However, the claim of the sender that he erased all or part of his random inputs will be taken against him. Here comes the difference we want to escalate between deniability and incoercibility: *Incoercible encryption is not receipt-free but allows the sender to lie safely without claiming the erasure of any of his random inputs while Deniable encryption is receipt-free i.e. disables the ability of the sender to prove to the coercer the plaintext that will be decrypted by the receiver.*

According to the instant of coercion, we have one of the following two situations:

1. *Unplanned coercion*: In this type we assume that there is absolutely no contact between the sender and the coercer before transmission of the ciphertext. The first contact between them is after the ciphertext has been transmitted and probably recorded by the coercer.

2. *Planned coercion*: In this type, the coercer may have one or more contacts with the sender before transmission which allows arrangements for the encryption and ciphertext transmission process.

In planned coercion, since the coercer may contact the sender before transmission, the coercer may deliver the sender the ciphertext he wants to see on the channel (i.e. the ciphertext is setup by the coercer himself), consequently, the sender is trapped and has nothing to do but obeying the coercer's order. For this reason, we have strong feeling that, theoretically, planned coercion is impossible to realize in practice, however, on the proof of this statement we make no claim, because some technical assumptions may open the door for realizing such a scheme.

We make another categorization according to the sender's will to cooperate with the coercer as follows:

1. *Collaborative-sender*: In this type, the sender is willing to cooperate with the coercer (e.g. knowing which message satisfactory to the coercer) so that, the sender is able to setup the encryption in a way that allows him later to prove the encrypted message to the coercer (i.e. the message that will be decrypted by the receiver) to benefit from such transmission (e.g. gaining some cash in return).

2. *Non-collaborative-sender*: The sender wants to perform the encryption and setup the ciphertext in a way that allows him later to *lie safely* to the coercer about the encrypted message without being proved as a liar.

In Collaborative-sender, notice that, the sender is willing to perform actions against his own beliefs, as long as such action satisfies the coercer. However, to do so, the sender needs to know exactly what message satisfies the coercer before transmission takes place so that he will be able to setup the encryption in a way that allows him later to prove to the coercer that this particular message will be decrypted by the receiver. For the sender to be able to do that (i.e. knowing what satisfies the coercer), there must be some sort of contact between him and the coercer prior to transmission. Hence, we conclude that *collaborative-sender is equivalent to planned coercion* which, again, is almost impossible to withstand in practice, since the sender can ask the coercer for the ciphertext that he wants to see on the channel.

In Non-collaborative sender, the sender is not willing to collaborate with the coercer, he just wants to perform actions satisfying his own beliefs and at the same time, is able to lie safely about his encrypted message (without claiming that he erased any of his local parameters such as random coins) to escape a coercive actions (e.g. violent actions) that could be attempted by the coercer. Again, we must emphasize that although the sender is non-collaborative, if the coercer approaches him before transmission, then the coercer will force him to send a particular ciphertext and the sender will not be able to lie in this case. Almost all previously proposed sender-Incoercible encryption and also the scheme introduced in this paper are unplanned Incoercible, that is, the coercer approaches the sender after transmission. *We want to emphasize that our sender-incoercible PKE (SI-PKE) scheme assumes absolutely no contact of any type between the coercer and the sender prior to the transmission of the ciphertext*. Under this assumption, after transmission, whenever the coercer approaches the sender, our SI-PKE allows the sender to open any message satisfactory to the coercer, and safely claims this fake message as the one that will be decrypted by $\mathcal{R}$ without the need to claim the erasure of any of his local randomness (e.g. random coins). In this case, the coercer is unable to prove that the sender's claim is false, and hence, Incoercibility holds.

New terminology distinction: In the literature, and up to this point, deniability and incoercibility were used alternatively for the same meaning. Here we make a new distinction between *Deniability* and *Incoercibility*. Our proposed scheme is in fact sender-incoercible but not sender-deniable. We introduce the following classification of coercion resistant encryption:

• *Incoercible encryption*: In this class of coercion resistant encryption, the sender still able to prove at least one of the encrypted messages to the coercer if he wants to, i.e., he can perform the encryption and setup the ciphertext in a way that allows him to prove to the coercer the message that will be decrypted by the receiver. Since the encryption in this case is still provable, we bring incoercibility as a new terminology for the encryption scheme that is still provable yet allows the sender to lie whenever he desires without claiming the erasure of any of his local randomness.

• *Deniable encryption*: In this class of coercion resistant encryption, the sender who sets up the ciphertext cannot prove (even to himself) the encrypted message. We notify that no PKE exists that satisfies this property in the non-erasure model. The only well-known fully deniable encryption scheme is the one-time-pad.

Figure 1 summarizes our new terminology distinction and categorization of services provided by coercion resistant schemes. It shows (by dark blocks) which services our SI-PKE provides. By inspecting previous schemes such as [1, 2] and those surveyed and reviewed in [20], we found that none of these encryption schemes are deniable in the sense mentioned above, since the sender is always able to prove to the coercer the plaintext that will be decrypted by the receiver. However, in the encryption schemes of [1, 2] the sender is able to lie safely to the coercer and hence, the schemes are incoercible.

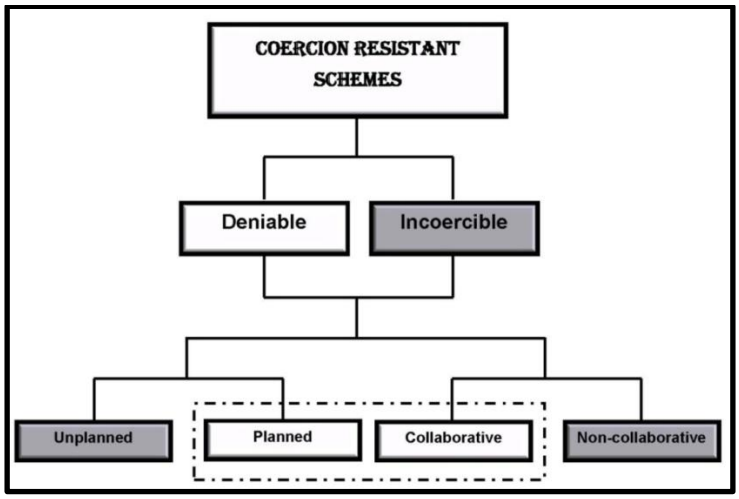


FIGURE 1: Our View of the Services Provided by Coercion Resistant Schemes.

5. EXISTING COERCION RESISTANT SCHEMES

In this section we review several schemes and protocols to declare the distinction between incoercibility and deniability. First we review the SI-PKE scheme of [2] and show why this scheme is incoercible but not deniable. Next, we discuss the one-time-pad as a perfectly deniable encryption scheme and also discuss the perfectly secure message transmission (PSMT) techniques as a method to achieve deniability under assumptions that are too strong to be realized in practice.

5.1 Sender-Incoercible PKE

In this section we review the SI-PKE introduced in [2]. The scheme could be built given any trapdoor permutation (f, f^{-1}), where, f is the receiver's public function and f^{-1} is its trapdoor inverse known only to the receiver. By $(b_{k-1}^{(x)} \dots b_0^{(x)})$ we denote the binary representation of $x \in \{0,1\}^k$. Let $f^{(j)}(y_0) = f(f(\dots f(f(y_0))\dots))$ be the process of encrypting y_0 (i.e. applying f to y_0) j times where $y_0 \in \mathcal{D}_f$. Let d be the maximum number of decryptions that will be performed by the receiver (i.e. the receiver will apply f^{-1} no more than d times). Let $H: \{0,1\}^* \to \{0,1\}^\ell$ be a hash function with digest (output) bit-length ℓ. The pair (f, d) is the receiver's public key. Let b_t be the true bit to be encrypted while b_f be the fake bit. The scheme is described next.

Encryption: To encrypt one bit, the sender proceeds as follows:

- *Honest encryption* $(b_t = b_f)$.
  - Picks y_0 at random from the domain of f such that, $b_t = \bigoplus_{i=0}^{k-1} b_i^{(y_0)}$.
  - Picks a small integer $0 < r < d$.
  - Computes $C = f^{(r)}(y_0)$.
  - Picks $e \in_R \{0,1\}$, sets $R_e = H(y_0)$ and $R_{1-e} \in_R \{0,1\}^\ell$.
  - Sends (C, R_0, R_1) to the receiver.

- *Dishonest encryption* $(b_t = \bar{b}_f)$.
  - Picks y_0 at random from the domain of f such that, $b_t = \bigoplus_{i=0}^{k-1} b_i^{(y_0)}$.
  - Keeps on applying f to y_0, that is, to compute $y_j = f^{(j)}(y_0), (j = 1,2,\dots) < d$ until there exists some y_j with its binary representation satisfying, $b_f = \bigoplus_{i=0}^{k-1} b_i^{(y_j)}$.
  - Applies f at least one more time to compute $C = f^{(r)}(y_0), j < r \leq d$.
  - Picks $e \in_R \{0,1\}$, sets $R_e = H(y_0)$ and $R_{1-e} = H(y_j)$.
  - Sends (C, R_0, R_1) to the receiver.

Decryption: On the reception of (C, R_0, R_1), $\mathcal{R}$ starts decrypting by computing $y_{i-1} = f^{-1}(y_i)$ for $i = d \cdots 1$, arranges the resulting parameters as the tuple $\mathcal{Y} = \langle y_{j_1}, \cdots, y_{j_d} \rangle$, he inspects $\mathcal{Y}$ for the least index j_i such that $H(y_{j_i})$ matches either R_0 or R_1. In this case (when the match is found) then $y_{j_i} = y_0$. Finally, $\mathcal{R}$ uses y_0 to decrypt for $b_t = \bigoplus_{i=0}^{k-1} b_i^{(y_0)}$.

Opening the encryption: In order to open the encryption honestly, the sender reveals y_0. To open dishonestly, the sender reveals y_j , claims that y_j is picked at random from the domain of f and that R_e is random.

Incoercibility. In the dishonest encryption, after transmission, when the coercer approaches $\mathcal{S}$, $\mathcal{S}$ lies safely by opening y_j. It is infeasible for the coercer to perform decryptions without the knowledge of f^{-1} and hence cannot reach y_0. In this case, from the properties of the hash function, the claim of $\mathcal{R}$ that R_e is random cannot be proven false by the coercer.

Undeniability. It is obvious that if $\mathcal{S}$ opens honestly (i.e. y_0) then he proves to the coercer that y_0 and hence b_t is the bit decrypted by $\mathcal{R}$ since in this case the coercer is able to reach y_j and knows that both R_0 and R_1 are not random and that y_0 is indeed with the smallest index, and consequently the scheme satisfies sender-incoercibility but not deniability

5.2 Perfectly Deniable Schemes

In the following we describe the one-time-pad and the PSMT as perfectly deniable schemes, yet, their assumptions are impractical.

One-time-pad. In a one-time-pad, the sender $\mathcal{S}$ and the receiver $\mathcal{R}$ are sharing a secret parameter k which is used only once. $\mathcal{S}$ prepares his message m where $|k| = |m|$ and sends the ciphertext $c = k \oplus m$. The receiver $\mathcal{R}$ decrypts by computing $m = k \oplus c$. Although one-time-pad encryption is impractical, we describe it here as the only perfectly deniable encryption scheme known. Given any ciphertext c, $\mathcal{S}$ or $\mathcal{R}$ may open the encryption of a fake message $m_f \neq m_t$ and $|m_f| = |m_t|$ as the pair (k_f, m_f) where $k_f = c \oplus m_f$. For the one-time-pad to be sender-and-receiver deniable encryption, both $\mathcal{S}$ and $\mathcal{R}$ must coordinate their stories for m_f. It is obvious that, neither $\mathcal{S}$ nor $\mathcal{R}$ are able to prove the encryption even to themselves. Given any ciphertext c', then pick a pair (k', m') such that $|c'| = |k'| = |m'|$ and $c' = k' \oplus m'$.

Perfectly secure message transmission. In *perfectly secure message transmission* (PSMT) first introduced in [21], and improved in subsequent contributions (e.g. [22, 23, 24, 25]), a sender $\mathcal{S}$ and a receiver $\mathcal{R}$ are connected by $n = 3t + 1$ channels with at most t channels are corrupted by the adversary, while the remaining $n - t$ channels are beyond the reach of the adversary (physically secured). Under these assumptions (in one round of communication) $\mathcal{S}$ is able to transmit a message m to $\mathcal{R}$ in a perfectly secure way using polynomial sharing. Finally, $\mathcal{R}$ decodes for the message using the well-known BerleKamp-Welch decoder [26]. In three rounds of communication (assuming that $\mathcal{S}$ is the party that always start the communication) connectivity could be improved to $n = 2t + 1$. Such perfect secure transmission is indeed non-committing and hence, deniable and adaptively secure, since it could be easily shown that the parties will always be able to fake a conversation as long as at least $n - t$ channels are beyond the reach of the adversary. Under the assumption that physically secure channels exist between every pair of nodes in the network, coercion resistant encryption is useless. Yet, the assumption that some channels are physically secured is impractical in most network applications (e.g. the internet and wireless connectivity) and hence, standard encryption techniques are employed to protect from an adversary with eavesdropping capacities that extend to the whole network. In this case, privacy is preserved (in the cryptographic sense) and correctness is achieved by assuming that the adversary corrupts a fraction of the network paths. However, in addition to the fact that the system becomes cryptographically secure, employing standard encryption techniques gives rise to adaptive and coercive vulnerabilities.

6. OUR IDEA AND BASIC TOOLS

Since we focus in this paper on sender incoercibility, we mention what a sender-incoercible encryption scheme is required to satisfy as follows:

- The sender must be able to open any information the coercer asks for. On the other hand, the coercer will not ask for something that does not exist or that cannot be proven to exist.
- The information opened by the sender must be consistent (or appear to be consistent) with the transmitted ciphertext.
- The sender must not claim the erasure of any of his local randomness, since such claim will not be accepted by the coercer as long as such local randomness is proven to exist (e.g. random coins in conventional PKEs).

We remark that $\mathcal{S}$ may claim that some computed parameters are picked at random as long as the cryptographic assumption prevents the coercer from detecting such lie. For example, given a hash function H, $\mathcal{S}$ may pick x and compute $y = H(x)$ and claims to the coercer that y is picked at random. Here, from the one wayness of H the claim of $\mathcal{S}$ cannot be proven false by the coercer and hence the coercer finds no reason to ask for any x since in this case he cannot prove the existence of this x.

The idea to improve the complexity of our previously proposed scheme in [2] is to make use of hybrid encryption. In hybrid encryption the public key of the receiver is used to encrypt a symmetric key for a symmetric encryption scheme. The symmetric key is used to encrypt an arbitrary length plaintext. In our scheme, the public key encryption algorithm is used in its OW security (as a one way trapdoor permutation) to encrypt several symmetric keys (we call this process "folded encryption"). The symmetric keys are used to encrypt the true and the fake messages using any available symmetric key encryption algorithm. This allows the encryption of arbitrary length messages (unlike the schemes in [2]). On the other hand, with the help of a strong hash function, the receiver will be able to reach the symmetric key intended to encrypt the true message and bypass all other fake keys/messages.

6.1 Asymmetric Encryption

An asymmetric encryption scheme [e.g. 3, 27, 28, 29], $\mathcal{AE} = (\mathcal{K}, \mathcal{E}, \mathcal{D}, \mathrm{COINS}, \mathrm{MSPC})$ is a triple of algorithms, associated with finite sets, $\mathrm{COINS}(k)$ and $\mathrm{MSPC}(k) \subseteq \{0,1\}^*$, for $k \in \mathbb{N}$, where:

•The algorithm $\mathcal{K}$, called the key-generation algorithm, is a probabilistic algorithm which on input $1^k \in \mathbb{N}$ outputs a pair of strings $(pk, sk) \leftarrow \mathcal{K}(1^k)$.

•The algorithm $\mathcal{E}$, called the encryption algorithm, is a probabilistic algorithm that takes a pair of strings, pk and x, and a string $r \leftarrow \mathrm{COINS}(k)$, and produces a string $y = \mathcal{E}_{pk}(x; r)$.

•The algorithm $\mathcal{D}$, called the decryption algorithm, is a deterministic algorithm that takes a pair of strings, sk and y, and returns a string $x = D_{sk}(y)$.

It is required that, for any $k \in \mathbb{N}$, if $(pk, sk) \leftarrow \mathcal{K}(1^k)$, $x \in \mathrm{MSPC}$, and $y \leftarrow \mathcal{E}_{pk}(x)$, then $\mathcal{D}_{sk}(y) = x$. In our proposed scheme, we need an asymmetric encryption algorithm which is one-way secure, hence we need the weakest security notion of asymmetric encryption defined next.

One-way Encryption: Let $\mathcal{AE}$ be an asymmetric encryption. For $\mathcal{AE}$, we consider an algorithm, $\mathcal{A}$, called an adversary, that, taking a public-key, pk, outputted by $\mathcal{K}$, and an encryption, y, of a random plaintext in MSPC tries to decrypt y. The probability of $\mathcal{A}$'s success, denoted by the advantage of $\mathcal{A}$, depends on $\mathcal{A}$, $\mathcal{AE}$, and the random choice of a plaintext from MSPC. $\mathcal{A}$ does not have any decryption oracle (while an encryption oracle doesn't matter because chosen-

plaintext attacks are clearly unavoidable in an asymmetric encryption scheme). For $k \in \mathbb{N}$, define the advantage of $\mathcal{A}$ by

$$Adv^{OW}_{\mathcal{A},\mathcal{AE},MSPC}(k) = Pr[(pk, sk) \leftarrow \mathcal{K}(1^k); x \leftarrow MSPC(k); y \leftarrow \mathcal{E}_{pk}(x): \mathcal{A}(pk, y) = \mathcal{D}_{sk}(y)]$$

It is said that an adversary $\mathcal{A}$ (t, ε)-breaks $\mathcal{AE}$ in the sense of OW'ness if $\mathcal{A}$ runs in at most time t and achieves $Adv^{OW}_{\mathcal{A},\mathcal{AE},MSPC}(k) \geq \varepsilon$. It is said that $\mathcal{AE}$ is (t, ε)-secure in the sense of OW'ness if there is no adversary that (t, ε)-breaks $\mathcal{AE}$ in that sense. A OW-secure $\mathcal{AE}$ is spoken of as "one-way trapdoor permutation".

6.2 Symmetric Encryption

A symmetric encryption scheme, $\mathcal{SK} = (E, D, \mathrm{KSPC}, \mathrm{MSPC})$ is a pair of algorithms associated with finite sets, $\mathrm{KSPC}(k)$ and $\mathrm{MSPC}(k)$, $\subseteq \{0,1\}^*$, for $k \in \mathbb{N}$, where:

• E, called the encryption algorithm, is a deterministic algorithm that takes a pair of strings, K and x, and produces $y = E_K(x)$.

• D, called the decryption algorithm, is a deterministic algorithm that takes a pair of strings, K and y, and outputs a string $x = D_K(y)$.

It is required that, for any $k \in \mathbb{N}$, if $K \in \mathrm{KSPC}(k)$, $x \in \mathrm{MSPC}$ and $y = E_K(x)$, then $D_K(y) = x$.

6.3 Important Notations for Our Scheme

The following notations are important to clearly understand our SI-PKE scheme:

- We denote by $y_j = \mathcal{E}^{(j)}_{pk}(y_0)$ the process of encrypting $y_0 \in \mathrm{MSPC}$ j-times $(j \geq 1)$ that is, $y_j = \mathcal{E}^{(j)}_{pk}(y_0) = \mathcal{E}_{pk}(\mathcal{E}_{pk}(\cdots \mathcal{E}_{pk}(y_0)))$
- We denote by $y_{j-i} = \mathcal{D}^{(i)}_{sk}(y_j)$ the process of decrypting y_j i-times $(1 \leq i \leq j)$ that is $y_{j-i} = \mathcal{D}^{(i)}_{sk}(y_j) = \mathcal{D}_{sk}(\mathcal{D}_{sk}(\cdots \mathcal{D}_{sk}(y_j)))$. Notice that $y_0 = \mathcal{D}^{(j)}_{sk}(y_j)$.

7. OUR SI-PKE SCHEME

We assume that the receiver $\mathcal{R}$ who is beyond coercion has a public/private key pair (pk, sk) for a OW-secure asymmetric encryption scheme $\mathcal{AE}$ (eg. RSA one-way trapdoor permutation). Let $H: \{0,1\}^* \rightarrow \{0,1\}^\ell$ be a strong hash function [30] with digest (output) bit-length ℓ. Let $(d > 2) \in \mathbb{N}$ be a small integer picked by $\mathcal{R}$. The pair (pk, d) is known to everyone including the sender $\mathcal{S}$ as $\mathcal{R}$'s public-key for the SI-PKE while sk is kept private to $\mathcal{R}$.

7.1 SI-PKE for One Fake Message

The sender $\mathcal{S}$ has two arbitrary length messages $m_t \in \mathrm{MSPC}^{\mathrm{sym}}$ and $m_f \in \mathrm{MSPC}^{\mathrm{sym}}$ where m_t is his true message aimed to be decrypted by $\mathcal{R}$ while m_f is the fake message that he may wish to open to a coercer later after transmission. Of course, we must have $|m_t| = |m_f|$. The SI-PKE scheme operates as follows:

Encryption: $\mathcal{S}$ proceeds as follows:

- *Honest Encryption* $(m_t = m_f)$:
  - Picks r at random such that $1 \leq r \leq d$.
  - Picks $K_0 \in_R \mathrm{MSPC}^{\mathrm{asym}}$.
  - Computes $K_r = \mathcal{E}^{(r)}_{pk}(K_0)$.
  - Picks $e \in_R \{0,1\}$.
  - Sets $(R_e, C_e) = (H(K_0), E_{K_0}(m_t))$ while $R_{1-e} \in_R \{0,1\}^\ell$ and $C_{1-e} \in_R \mathrm{MSPC}^{\mathrm{sym}}$.
  - Sends the tuple $\mathcal{C} = \langle K_r, (R_0, C_0), (R_1, C_1) \rangle$.

- *Dishonest Encryption* ($m_t \neq m_f$):
  - Picks r at random such that $1 \leq r \leq d$.
  - Picks $K_0 \in_R \text{MSPC}^{\text{asym}}$.
  - Computes $K_r = \mathcal{E}_{pk}^{(r)}(K_0)$.
  - Picks any K_j, $(0 < j < r)$.
  - Picks $e \in_R \{0,1\}$.
  - Sets $(R_e, C_e) = (H(K_0), E_{K_0}(m_t))$ and $(R_{1-e}, C_{1-e}) = (H(K_j), E_{K_j}(m_f))$.
  - Sends the tuple $\mathcal{C} = \langle K_r, (R_0, C_0), (R_1, C_1)\rangle$.

Decryption: On the reception of $\mathcal{C}$, $\mathcal{R}$ starts decrypting by computing $K_{i-1} = D_{sk}(K_i)$ for $i = d \cdots 1$, arranges the resulting keys as the tuple $\mathcal{K} = \langle K_{j_1}, \cdots, K_{j_d}\rangle$, he inspects $\mathcal{K}$ for the least index j_i such that $H(K_{j_i})$ matches either R_0 or R_1. In this case, when the match is found, then $K_{j_i} = K_0$. Finally, $\mathcal{R}$ uses K_0 to decrypt for $m_t = D_{K_0}(C_e)$.

Opening the encryption: In order to open the encryption honestly, the sender reveals K_0. To open dishonestly, the sender reveals K_j , claims that K_j is picked at random from $\text{MSPC}^{\text{asym}}$ and that C_e and R_e are random.

Incoercibility. In the dishonest encryption, after transmission, when the coercer approaches $\mathcal{S}$, $\mathcal{S}$ lies safely by opening K_j. Since from the OW security of the asymmetric encryption scheme and assuming $\mathcal{R}$ is beyond coercion/corruption, it is infeasible for the coercer to perform decryptions without the knowledge of sk and hence cannot reach K_0. In this case, from the properties of the hash function, the claim of $\mathcal{R}$ that R_e is random cannot be proven false by the coercer. From the chosen ciphertext/plaintext security of the symmetric encryption scheme, even if the coercer knows m_t and given the corresponding C_e, he cannot prove that C_e is not random without the knowledge of the encryption key K_0. Hence, by using our SI-PKE, $\mathcal{S}$ can safely lie to the coercer without being caught.

We remark that opening K_0 allows the coercer to reach K_j by performing $K_j = \mathcal{E}_{pk}^{(j)}(K_0)$, therefore the coercer will easily detect that C_{1-e} and R_{1-e} are not random. This does not threaten incoercibility since the claim of $\mathcal{S}$ that K_0 is the key with the smallest index is true. We emphasize that when $\mathcal{S}$ opens K_j and claims that K_j is the key with the smallest index (i.e. K_0), the coercer given pk and the ciphertext cannot reach any K_i with $i < j$.

7.2 SI-PKE for Multiple Fake Messages

We simply extend our SI-PKE for one fake message to the case where the sender needs to prepare for more than one fake message. Here, the complexity will grow linearly with the number of fake messages. Let $m_t \in \text{MSPC}^{\text{sym}}$ be the true message while $m_f^{(1)}, \cdots, m_f^{(n)} \in \text{MSPC}^{\text{sym}}$ be the $n \geq 1$ possible fake messages that the sender may want to open any of them later to the coercer. We have $|m_t| = |m_f^{(i)}|, \forall i$ and the number of fake messages (n) must be fixed and does not change from one encryption to another. It is required that $d > n$ and this could be easily satisfied by allowing $\mathcal{R}$ to choose d alittle bit larger than the case for one fake message encryption. Here, we may call n and d, *the faking capacity* of the SI-PKE scheme. The SI-PKE scheme is as described next.

Encryption: $\mathcal{S}$ proceeds as follows:

- *Honest encryption*:
  - Picks $K_0 \in_R \text{MSPC}^{\text{asym}}$.
  - Picks a random r such that $n \leq r \leq d$.
  - Computes $K_r = \mathcal{E}_{pk}^{(r)}(K_0)$.
  - Picks $e \in_R \{0, \cdots, n\}$.

- Sets $(R_e, C_e) = (H(K_0), E_{K_0}(m_t))$ while $R_{v\neq e} \in_R \{0,1\}^{\ell}$ and $C_{v\neq e} \in_R \text{MSPC}^{\text{sym}}$ for $v = 0 \cdots n$.
- Sends the tuple, $\mathcal{C} = \langle K_r, (R_0, C_0), \cdots, (R_n, C_n)\rangle$ to $\mathcal{R}$.

• *Dishonest Encryption* ($m_t \neq m_f^{(1)} \neq \cdots \neq m_f^{(n)}$):
- Picks $K_0 \in_R MSPC^{\text{asym}}$.
- Picks a random r such that $n \leq r \leq d$.
- Computes $K_r = \mathcal{E}_{pk}^{(r)}(K_0)$.
- Picks any K_j's, $(0 < j < r)$ and arranges them as the tuple $\langle K_{j_1}, \cdots, K_{j_n}\rangle$.
- Picks $e \in_R \{0, \cdots, n\}$.
-Sets $(R_e, C_e) = (H(K_0), E_{K_0}(m_t))$ and assigns each other $(R_{v\neq e}, C_{v\neq e})$ a value $\left(H\left(K_{j_{v\neq 0}}\right), E_{K_{j_{v\neq 0}}}\left(m_f^{(v\neq 0)}\right)\right), v = 0, \cdots, n$.
- Sends the tuple, $\mathcal{C} = \langle K_r, (R_0, C_0), \cdots, (R_n, C_n)\rangle$ to $\mathcal{R}$.

Decryption: On the reception of $\mathcal{C}$, $\mathcal{R}$ starts decrypting by computing $K_{i-1} = D_{sk}(K_i)$ for $i = d \cdots 1$, arranges the resulting keys as the tuple $\mathcal{K} = \langle K_{j_1}, \cdots, K_{j_d}\rangle$, he inspects $\mathcal{K}$ for the least index j_i such that $H(K_{j_i})$ matches any of the R_v's for $v = 0 \cdots n$. In this case, when the match is found, then $K_{j_i} = K_0$ and $R_v = R_e$. Finally, $\mathcal{R}$ uses K_0 to decrypt for $m_t = D_{K_0}(C_e)$.

Opening the encryption: In order to open the encryption honestly, the sender reveals K_0. To open dishonestly, the sender opens the fake message he wants (say $m_f^{(v)}$) and reveals the corresponding key K_{j_v} claiming that K_{j_v} is picked at random from $\text{MSPC}^{\text{asym}}$ (i.e. $K_{j_v} = K_0$).

Incoercibility. The sender $\mathcal{S}$ picks any fake message $m_f^{(v)} \in \{m_f^{(1)}, \cdots, m_f^{(n)}\}$ and lies safely by opening the corresponding key K_{j_v} used to encrypt this message. Since from the OW security of the asymmetric encryption scheme and assuming $\mathcal{R}$ is beyond coercion/corruption, it is infeasible for the coercer to perform decryptions without the knowledge of sk and hence cannot reach K_0 or any key $K_{j_1}, \cdots, K_{j_{v-1}}$. In this case, from the properties of the hash function, $\mathcal{S}$'s claim that $R_0, \cdots, R_{v-1}$ are random cannot be proven false by the coercer. From the chosen ciphertext/plaintext security of the symmetric encryption scheme, even if the coercer knows m_t and all fake messages and given $C_0, \cdots, C_n$, he cannot prove that $C_0, \cdots, C_{v-1}$ are not random without the knowledge of the encryption keys $K_0, K_{j_1} \cdots, K_{j_{v-1}}$. Hence, by using our SI-PKE, $\mathcal{S}$ is able to safely lie to the coercer without being caught. Notice that the coercer given K_{j_v} is able to find all keys $K_{j_{v+1}}, \cdots, K_{j_r}$ and hence he knows that $R_{v+1}, \ldots, R_n$ and $C_{v+1}, \ldots, C_n$ are not random. We emphasize that the main claim of $\mathcal{S}$ is that the opened key K_{j_v} is the key with the smallest index (i.e. $j_v = 0$) which still cannot be proven false by the coercer and therefore incoercibility still holds.

Finally, from previous discussions in this paper, it is clear by inspection that our SI-PKE scheme is unplanned incoercible assuming the sender is non-collaborative and the receiver is beyond coercion, yet, the scheme is undeniable.

8. CONCLUSIONS

In this paper we introduced a new classification of services in the area of coercion resistant encryption showing that coercion resistant encryption schemes are classified as either incoercible encryption or deniable encryption where: incoercible encryption allows the sender to lie without claiming the erasure of any of his random inputs but still committed to the encryption and is able (under his own choice) to prove to the coercer the message that will be decrypted by the receiver, while in deniable encryption, the sender cannot prove even to himself the decrypted message. We also showed (heuristically) that planned coercion-resistant encryption, where the coercer is assumed to approach the sender before transmission is impossible to realize in practice. Then we proposed an improvement to our previously proposed sender incoercible encryption to achieve constant transmission rate where the size of the plaintext may be calibrated to be sufficiently

large. Unlike previous schemes where the size of the ciphertext is super polynomial, our scheme encrypts arbitrary length messages without a blowup expansion in the ciphertext while the size of the ciphertext grows linearly with the number of fake messages.

9. REFERENCES

[1] R. Canetti, C. Dwork, M. Naor, R. Ostrovsky, “Deniable Encryption,” in CRYPTO’97, 1997, pp.90-104.

[2] M. H. Ibrahim, “A Method for Obtaining Deniable Public-Key Encryption,” international journal of network security (ijns), Vol. 8, No. 1, 2009, pp. 1–9.

[3] S. Goldwasser, S. Micali, “Probabilistic Encryption,” Journal Computer System Science, 28(2), 1984, pp. 270–299.

[4] M. Naor, M. Yung, “Public-key Cryptosystems Provably Secure against Chosen Ciphertext Attacks,” in proceedings of STOC 1990, pp. 427–437.

[5] C. Rackoff, D. R. Simon, “Non-Interactive Zero-Knowledge Proof of Knowledge and Chosen Ciphertext Attack,” CRYPTO 1991, pp. 433–444.

[6] D. Dolev, C. Dwork, M. Naor, “Non-Malleable Cryptography,” STOC 1991, pp. 542–552.

[7] A De Santis, G. D. Crescenzo, R. Ostrovsky, G. Persiano, A. Sahai, “Robust Non-interactive Zero Knowledge,” CRYPTO 2001, pp. 566–598.

[8] A. Herzberg, S. Jarecki, H. Krawczyk, M. Yung, "Proactive secret sharing or: How to cope with perpetual leakage," LNCS 963, Proc. Crypto’95, Springer Verlag, 1995, pp. 339–352.

[9] R. Ostrovsky, M. Yung: How to Withstand Mobile Virus Attacks, (Extended Abstract), PODC 1991, pp. 51–59.

[10] R. Canetti, I. Damgard, S. Dziembowski, Y. Ishai, T. Malkin: On Adaptive vs. Non-adaptive Security of Multiparty Protocols. EUROCRYPT 2001: 262–279.

[11] R. Canetti, U. Feige, O. Goldreich, M. Naor, “Adaptively Secure Multi-Party Computation,” STOC 1996, pp. 639-648

[12] M. H. Ibrahim, “Receiver-Deniable Public-Key Encryption,” international journal of network security (ijns), Vol. 8, No. 2, 2009, pp. 159–165.

[13] R. Canetti, R. Gennaro, “Incoercible multiparty computation,” in Proceedings of the 37th Annual Symposium on Foundations of Computer Science, 1996, pp. 504–513.

[14] D. Beaver, “Plug and Play Encryption,” in proceedings of CRYPTO 1997, pp. 75–89.

[15] R. Canetti, S. Halevi, J. Katz, “Adaptively-Secure, Non-interactive Public-Key Encryption,” in proceedings of TCC 2005, pp. 150–168.

[16] R. Canetti, U. Feige, O. Goldreich, M. Naor, “Adaptively Secure Multi-Party Computation,” STOC 1996, pp. 639–648.

[17] J. B. Nielsen, “Separating Random Oracle Proofs from Complexity Theoretic Proofs: The Non-committing Encryption Case,” CRYPTO 2002, pp. 111–126.

[18] I. Damgard, J. B. Nielsen, “Improved Non-committing Encryption Schemes Based on a General Complexity Assumption,” CRYPTO 2000, pp. 432–450.

[19] S. Jarecki, A. Lysyanskaya, “Adaptively Secure Threshold Cryptography: Introducing Concurrency, Removing Erasures,” EUROCRYPT 2000, pp. 221–242.

[20] B. Meng, “A Critical Review of Receipt-Freeness and Coercion-Resistance,” Information Technology Journal, Volume 8 Issue 7, 2009.

[21] D. Dolev, C. Dwork, O. Waarts, and M. Yung, “Perfectly secure message transmission,” Journal of the ACM, 40(1), Jan. 1993, pp. 17-47.

[22] K. Srinathan, A. Patra, A. Choudhary, C. P. Rangan, “Probabilistic Perfectly Reliable and Secure Message Transmission - Possibility, Feasibility and Optimality,” INDOCRYPT 2007, pp. 101-122.

[23] A. Patra, A. Choudhary, K. Srinathan, C. P. Rangan, “Constant Phase Bit Optimal Protocols for Perfectly Reliable and Secure Message Transmission,” INDOCRYPT 2006: pp. 221-235.

[24] M. Fitzi, M. K. Franklin, J. A. Garay, S. H. Vardhan, “Towards Optimal and Efficient Perfectly Secure Message Transmission,” TCC 2007, pp. 311-322

[25] K. Kurosawa, K. Suzuki, “Truly Efficient 2-Round Perfectly Secure Message Transmission Scheme,” EUROCRYPT 2008, pp. 324-340.

[26] E. Berlekamp and L. Welch, “Error correction of algebraic block codes.” US Patent, 4,633,470, USA.

[27] T. El-Gamal, “A public key cryptosystem and a signature scheme based on discrete logarithms,” IEEE Transactions on Information Theory, 31(4), 1985, pp. 469–472.

[28] R. Cramer, V. Shoup, “A Practical Public Key Cryptosystem Provably Secure Against Adaptive Chosen Ciphertext Attack,” CRYPTO 1998, pp. 13–25.

[29] P. Paillier, “Public-Key Cryptosystems Based on Composite Degree Residuosity Classes,” EUROCRYPT 1999, pp. 223–238.

[30] M. Bellare, P. Rogaway, “Random Oracles are Practical: A Paradigm for Designing Efficient Protocols,” ACM Conference on Computer and Communications Security, 1993, pp. 62–73.

E-payment Security Analysis In Depth

Adam Ali.Zare Hudaib
(Certified Ethical Hacker) *adamhudaib@gmail.com*
"Two Mas" ltd Krs414007
Network Security Research
Amman, Jordan. Poland

Abstract

Technology is the basis of our lives. The growth of the Internet has made it an ease for consumers to find items for purchase, but no longer is cash a viable way for payment. This increase in e-commerce has driven the need to create an online payment system. Unfortunately there are a lot of flaws and internet frauds that people are facing.

In this research we will review different payment protocols and security methods that are being used to run online payment systems. We will survey some of the popular systems that are being used today, with a deeper focus on the PayPal system, NFC and bitcoins. In addition, we will also discuss the weaknesses in the systems that can compromise the customer's trust.

Keywords: E-payment, Security Protocol, Bitcoin, NFC.

1. INTRODUCTION

Progress in web technologies has led to rapid growth of hybrid web applications that combine the Application Programming Interfaces (APIs) of multiple web services (e.g., search APIs, map APIs, payment APIs, etc.) into integrated services like personal financial data aggregations and online shopping websites. The pervasiveness of these applications, however, brings in new security concerns. The web programming paradigm is already under threat from malicious web clients that exploit logic flaws caused by improper distribution of the application functionality between the client and the server (e.g., relying on client logic to validate user privileges). The program logic of a hybrid web application is further complicated by the need to securely coordinate different web services that it integrates: failing to do so leaves the door wide open for attackers to violate security invariants by inducing inconsistencies among these services.

Intuitively, logic bugs related to multiple web services can be much more difficult to avoid than those in traditional single-service web applications – it is analogous to real-life experiences that when multiple parties discuss a subject by making individual one-on-one phone calls, it is generally difficult for each party to comprehend the whole picture. So in this research we will review different payment protocols and security methods that are being used to run online payment systems.

2. E-PAYMENT SECURITY ANALYSIS IN DEPTH

There are a lot of payment systems that are being used today. But every system has its pros and cons. We will focus on this further.

2.1 Payment Systems

There is payment by instruction type of systems, when a payer basically orders the bank to move a sum of money from her account into a payee's account. Examples in this category are credit and debit cards as well as many forms of cheques. The moment at which the money is actually moved from the payer's account into the payee's account depends on the system, but at all times banks and credit card companies will try to prevent discrepancies between accounts. The central

security aspect in these systems is to ensure that only legitimate account holders are able to issue payment instructions. Of course, digital signatures are the solution for doing this over a large, open network such as the Internet. Since digital signatures only make sense if there is an infrastructure for certifying public keys, a lot of effort is devoted to just this. See, for instance, the SET (Secure Electronic Transaction) proposal, a joint effort by MasterCard, VISA, and other influential partners, which specifies a hierarchy of certification authorities on top of the payment protocols [1].

Prepaid systems are conceptually close to electronic equivalents of cash. Telephone cards, smart card based systems, as well as e-cash fall into this category. The user's account is debited as soon as the card or device is reloaded with electronic cash. During payments the electronic cash is released again, and only then the payee's account will be credited. In the meantime the issuer keeps a float corresponding to the outstanding cash. The central security aspect in this type of system is to ensure that cards or representations of cash cannot be forged. When forgery happens, the float will ultimately be insufficient to credit all of the payees' accounts for received payments. Of course, it should also be ensured that only legitimate account holders can reload cash from their accounts. However, this security aspect is now limited to the infrequent withdrawal protocol, and is no part anymore of the more frequent payment protocol.

Although the payment protocol is functionally a protocol between two parties (payer and payee) many payment systems require that the payee contacts a third party (e.g., the bank or the credit-card company acting as an acquirer) before accepting a payment. If that is the case, the system is called an on-line payment system; the communication between a payee and its acquirer may be using any communication medium (not necessarily the Internet). If such a contact with a third party is not required during the payment protocol, the system is called off-line. In an off-line system payees are required to contact their acquirer on a regular basis for clearing all received payments.

A basic requirement of a payment protocol is that it allows a payee to receive payments from any payer. A payment can be seen as some sort of authentication of the payer towards the payee (to show that the payment is authentic). Authentication can be based on secret key cryptography or on public key cryptography. In the latter case, the payee only needs to have a public key available in order to verify incoming payments. Although the costs of equipping smart cards with crypto co-processors are expected to become marginal, it is important to note that the property of public verifiability can be obtained using simple smart cards only, provided one applies a method of what we call signature transport. In such a system, signatures are created by the issuer only, and later endorsed by the payer during the payment protocol, depending on a challenge from the payee. The trick is to achieve that sufficiently many payments can be made between successive reloads, which requires optimal use of the limited amount of EEPROM available on simple smart cards. The added advantage is that the secret key for creating signatures is only used by the issuer. In case authentication is based on secret key (symmetric) cryptography, how- ever, the payer and payee must have a shared secret key available in order to complete a payment. A straightforward solution is to give all users the same secret key, but this is generally considered insecure, as this would mean that breaking a single smart card (i.e., extracting its secret key) will suffice to break the complete system. The standard solution is therefore to break the symmetry between payers and payees by equipping the merchants with a highly secure tamper-proof box called a SAM that contains a master key. The payers' keys are derived from this master key in a process called diversification by applying a cryptographic hash (e.g., SHA-1) to the concatenation of the master key and the payer's card number. The idea is that the SAM is more difficult to break than a smart card, and also that it is possible to routinely check (as part of the maintenance) if the SAMs have not been tampered with. In the EMV standard (developed by Europay, MasterCard, and Visa) a first step is made toward including public key authentication. To prevent frauds in which cards with fake card numbers are introduced, each card carries a fixed RSA certificate that shows the validity of the card number. At the start of each payment, the certificate can be verified against the public key stored in the POS terminal. The remainder of the payment protocol again relies on a secret master key stored in the SAM of the POS terminal.

Checkout based on using electronic payment system consists of some typical steps. For example, if using PayPal, it starts when the button "Check out with PayPal" on page of the merchant website is clicked. Then user is directed to page on PayPal, where he can click the "Pay Now" button to pay. Then, the shopper's browser is redirected back to the merchant's website to finish the order, which usually does not require the shopper's actions. Finally, the shopper gets the confirmation page. The checkout process is arranged in this way to ensure that all three parties – the shopper, the e-payment system, and the merchant, stay consistent despite their different locations across the Internet.

Dynamic web are invoked through HTTP requests: the client sends an HTTP request through a URL with a list of arguments and receives an HTTP response (often a web page) dynamically constructed by the server as the outcome of the call.

These responses serve as the building blocks for the workflows of various checkout solutions offered by different payment systems service providers (Amazon, PayPal, and Google). Some of the solutions, such as PayPal Standard and Amazon Simple Pay, are entirely based upon HTML, while the others, like PayPal Express and Checkout By Amazon, implement SOAP and NVP APIs. Also e-payment systems websites communicate exclusively over HTTPS to guarantee end-to-end security [2].

2.2 Securing Checkout Processes

The main security goal of a checkout system is to maintain the following payment-completion invariant: Merchant M changes the status of an item I to "paid" with regard to a purchase being made by Shopper S if and only if:

1. M owns I;
2. a payment is guaranteed to be transferred from an account of S to that of M in the e-payment system;
3. the payment is for the purchase of I, and it is valid for only one piece of I;
4. the amount of this payment is equal to the price of I [3].

This invariant, though intuitive, implies a set of intertwined binding relations that should be respected in every step of the transaction. These bindings unequivocally link the merchant to a piece of the item being sold, the price of the item to the payment the merchant receives, and the payment for this specific purchase to the shopper. Complexity in preserving the invariant. To achieve this security goal, a checkout system is expected to preserve the aforementioned invariant throughout a transaction. This turns out to be nontrivial, particularly in the presence of two web services. Specifically, the challenges in keeping both servers in consistent states include, but are not limited to, the following:

1) Confusion in coordination. Given their incomplete views of a transaction, the merchant and the payment system need to work together to preserve the invariant. This, however, is often hindered by the partial knowledge each party has about the other: the code of their systems is often off-limits to each other; the payment system typically provides nothing but vague descriptions of its operations. As a result, misunderstanding often arises on the security assurance either party offers. For example, a merchant may assume that every notification of a payment completion from the payment system must be about one of his transactions, but the payment system may not have this guarantee and may expect a merchant to verify it by itself.

2) Diversity in the adversary's roles. The merchant and the payment system expose their APIs to the public, which enables the adversary to play more diverse roles than just the shopper, and thus to gain a deeper involvement in the checkout process than he

could in a more traditional client-server interaction. The shopper can directly invoke a merchant's APIs, which mimics the behavior of the payment system; the shopper can also mimic a merchant to register with the payment system a callback API.

3) Parallel and concurrent services. Both the merchant website and the payment system need to serve many customers, and a shopper can concurrently invoke multiple purchase transactions. This further complicates the trilateral interactions, opening avenues for cross-transaction attacks.

4) Authentication and data integrity. Compared with the two-party web applications, authentication in a payment system-based checkout system involves three parties and is thus more difficult in avoiding authentication and data integrity breaches.

2.3 Technologies Used for Online Payment Security

There are a few different protocols that are used for online security today. The most common security mechanism is SSL. Some of the others include TLS, and SET.

Secure Sockets Layer, more commonly known as SSL, is a protocol that is used to maintain client and server authentication. A site is easily identified as using SSL if it has the small yellow padlock at the bottom of the browser.

In SSL, communication between the server and the client is encrypted using their certificates. This encryption creates virtual information that is not hackable by others. The steps of how SSL works is shown in the following diagram:

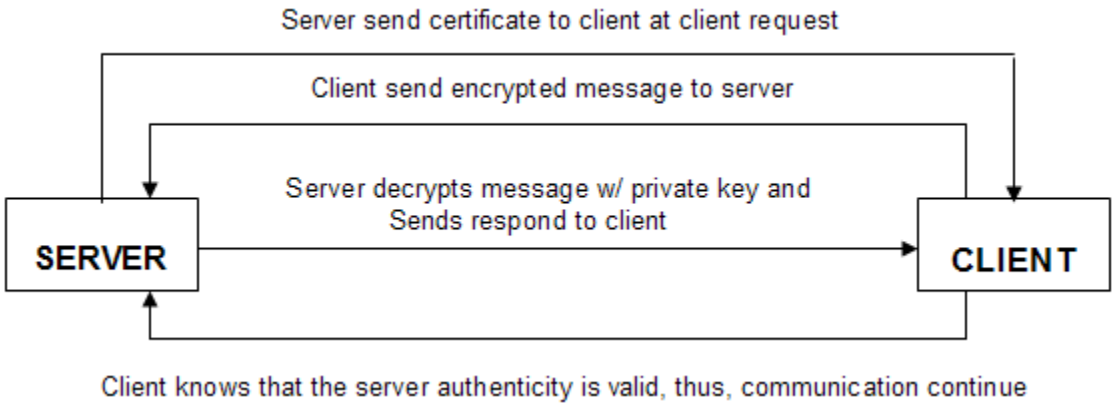


FIGURE 1: SSL Algorithm.

SSL Version 1, a test version, was quickly replaced by SSL version 2, which was the first version, released to the public and was shipped with the Netscape Navigator browser. Today version 2 is still supported despite having some security problems. Later, Microsoft came out with its own version of SSL called PCT. SSL Version 3 is a complete redesign of SSL and fixes the problems found in previous versions as well as having additional features [4].

The purpose of SSL is to provide a means to allow secure communication between two parties. However, one party must have a certificate trusted by the other in order to help prevent man in the middle attacks. SSL also supports authentication, encryption and key exchange.

SSL uses a handshake protocol. Suppose a client wants to make a purchase from a website server, but this server does not know anything about the client.

The first step is for the client to send a message to the server. After the server receives the message, it acknowledges it by sending the client a message in return. The server also sends the client its certificate and asks for the client's certificate. The client sends its certificate, a client key exchange message, and a certificate verification message. Both the client and server send change cipher spec messages and then send finished messages to end the handshake [5].

A website implements SSL by using HTTPS, which stands for Hypertext Transfer Protocol over Secure Socket Layer. This web protocol was developed by Netscape to encrypt and decrypt page requests as well as the pages that are returned by the web server. HTTPS uses port 443 instead of port 80, which is used for HTTP.

SSL uses a key size of 40-bits for the RC4 stream encryption algorithm. This is considered a sufficient degree of encryption for commercial exchange. Both HTTPS and SSL support the use of X.509 digital certificates from the server. This way, the user can authenticate the sender if needed [6].

One of SSL's strengths is its ability to help prevent some common attacks. SSL is strong against the brute force attack because it uses 128 bits. The dictionary attack which tends to be more efficient than a brute force attack is where an attack tries every word in a dictionary as a possible password for an encrypted message. This attack is also avoidable because SSL has very large key spaces. The replay attack which reruns messages that were sent earlier is prevented since SSL uses 128-bit nonce value to indicate a unique connection. And as mentioned earlier, the Man-In-the-Middle Attack is prevented by using signed certificates to authenticate the server's public key.

Despite the fact that SSL has the ability to prevent some common attacks, it still has some weaknesses. One of the weaknesses found in SSL is the brute force attack against weak ciphers. This weakness was forced by the US export on Netscape. This weakness still remains one of the most obvious weaknesses of the SSL protocol and it has broken many times [7].

Another weakness in SSL is the renegotiation of the master key. It is known that after a connection has been established, the same master key gets used all the way through the connection. This could be a serious security flaw if SSL are layered underneath a long running connection. One possible solution for this flaw is to force renegotiation of the master key at different times. This way, the difficulty and the cost of the any brute force attack will be multiplied by the number of times that the master key has changed [8].

The Transaction Layer Security protocol, commonly known as TLS, is based on SSL and will soon become its successor. TLS has some changes in its MAC, has clearer and more precise specifications, cleaner handling because of not having a client certificate, and more flexibility.

Secure Electronic Transaction, SET, provides a way for the client's credit card number to be sent to authorizing banks. However, there was not enough market acceptance of SET to make it commonplace.

2.4 Example of Using SSL by PayPal

With today's technology, the Internet has become the most popular place for people who want to buy goods and services. In order for people to do such kind of trading, they need a safe online payment system that they can trust. PayPal is one of the world's largest online payment systems. All you need is an account with PayPal and you will be ready to send and receive payments online securely.

The technologies used by PayPal consist of the main security mechanisms that most sites would employ. PayPal uses HTTPS and SSL to encrypt the data stream when a user establishes a session with the PayPal site. It is unknown what security mechanisms are used to protect their databases containing information about their customers.

According to the PayPal website, PayPal encrypts information sent to their website using SSL. It uses an encryption key that is 128-bits long, which is currently the most secure level being used today. Before proceeding, the server checks whether or not the user's browser uses SSL 3.0 or higher.

PayPal also uses an electronic firewall to protect its data from the Internet. Their servers are behind the firewall and not directly connected to the Internet in order to protect private information from unauthorized computers [9].

It is unclear what other forms of technology PayPal uses to ensure security, because of their avoidance to divulge too much information to the public as to put them at a security risk to attackers.

Popular companies such as Microsoft and PayPal have been attacked by hackers from all around the world because of some security flaws in their systems. Despite the fact that PayPal is one of the world's largest online payment systems, it has some security flaws and weak points.

Because of PayPal's heavy reliance on SSL as a means to achieve security, many of their weaknesses are therefore the weaknesses of SSL.

One of the attacks on PayPal was done by few experienced hackers from Russia who discovered a serious security flaw in the "address confirmation process" of PayPal's members' accounts. The hacking process was exposed to everyone on the internet in Russian language. It has been confirmed that PayPal had some technical difficulties in fixing the problem mentioned above, which means that a lot of PayPal accounts with confirmed address could be hacked into [10].

Email scams have been the source of security problems for PayPal. Many users have received emails appearing to be from PayPal urging them to click on a link and log in to their accounts.

This scam is dangerous because hackers obtain passwords by using false e-mail messages. Third parties hack PayPal accounts, using the passwords obtained, and then login to the accounts to steal money. They send false e-mails to PayPal users leading them to think that the e-mails were sent by PayPal when they are not [11]

Another email scam shows up directly in the email inbox and does not require the user to click on any links. This email scam is a bit more advanced as it shows a form directly in the email. The email asks the user to give information in order to confirm the information in PayPal's database; not doing so may risk cancellation of their account.

The problem with this email is not only is it not from PayPal, but it leads the naïve user to a website that appears to be an authentic PayPal website. The act of pretending to be somebody else's website is referred to as web spoofing.

In the email scam can provide links to not a PayPal site. Although the link shown is a PayPal site, after clicking on it, the address for the link appears to be different. This may go unnoticed to naïve users because the images and content are very similar to the real PayPal's website. The problem with this is when the user logs in to their account to prevent their account from being cancelled. Since this website is not authentic, the user submits their username and password to an unknown third party. This third party is then able to store their username and password into a database to cause damage to these users' accounts.

The use of client certificates would help stop web spoofing. However, client side certificates are hardly used, so the practicality of this is not very great. PayPal should also protect the images that are used on their website such that it cannot be saved or used by the public. This would help prevent attackers from stealing the images to create pages that mirror PayPal in its appearance.

PayPal's security relies heavily on user passwords. Although they limit the number of attempts to login, the limit seems pretty high. PayPal limits login attempts to ten accesses before locking the account [12]. While the chances of attackers using brute force to break into an account is rather slim, the opportunities are greater than other websites in which a user is limited to three attempts. This is a very significant security issue. If an attacker is able to access a user's account, he can wreak havoc by stealing money from the user.

An initial solution to this problem is to decrease the limit of login attempts. In addition, PayPal should have another layer of protection after entering a correct password, such as a question that requires a correct answer similar to what is done for password retrieval.

Other solutions that would help with online payment security involve the user to be alert at all times. One possible way to prevent thieves from stealing your password is to never trust any e-mail coming from PayPal and never click on a link that would take you to PayPal directly from an e-mail. So instead of clicking on the link it is better to go to the website by typing the website's URL. Also, it is a good idea to make sure the URL entered contains https://.

It should be remained that according to a report by a team of researchers from Indiana University and Microsoft Corp. major e-payment systems, e-retailers and e-commerce platforms—including Google Checkout, PayPal, Amazon Payments, Buy.com, JR.com, nopCommerce and Interspire—have payment system security software flaws that can be exploited to confirm payment to an illegitimate web site or to receive products for free or at reduced prices. The team's research so far has only touched on what it calls "the simplest trilateral interactions" among payment services providers, e-commerce platforms and online retailers.

XiaoFeng Wang, however, says payment service providers should take a more effective and proactive role in ensuring their systems are properly deployed. Also the researcher adds that payment services providers should also give merchant software developers tools to analyze their service integration. In the research it was also shown how someone could manipulate the e-mail payment notification system within Amazon Payments to make a payment on one e-commerce site result in a payment confirmation to a different site.

2.5 Bitcoin

Bitcoin is a peer-to-peer payment network and digital currency based on an open source protocol, which makes use of a public transaction log. Bitcoin was introduced in 2009 by pseudonymous developer "Satoshi Nakamoto". It is called a cryptocurrency because it uses public-key cryptography. When paying with bitcoin, no actual monetary exchange takes place between buyer and seller. Instead, the buyer requests an update to a public transaction log, the blockchain. This master list of all transactions shows who owns what bitcoins currently and in the past and is maintained by a decentralized network that verifies and timestamps payments using a proof-of-work system. The operators of this network, known as "miners", are rewarded with transaction fees and newly minted bitcoins [13].

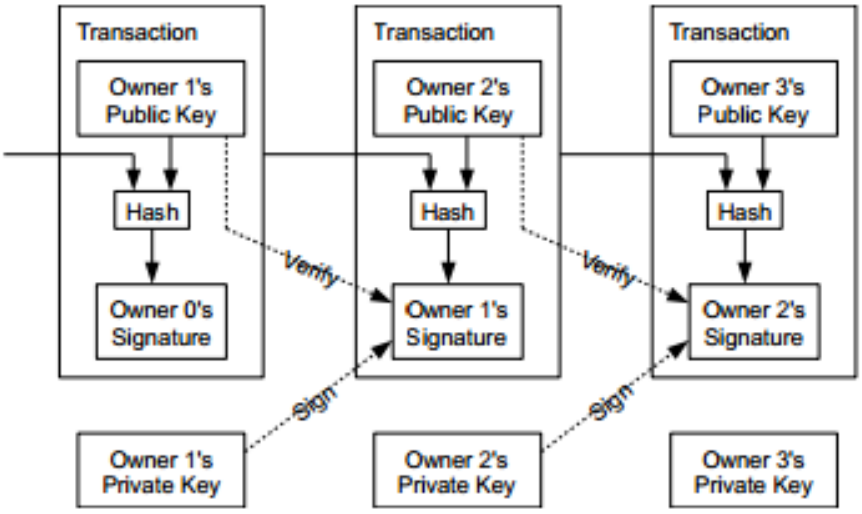


FIGURE 2: Bitcoin Payment Processing.

In order to make a payment, a user requests an update to the master transaction list, the blockchain, and the transaction is validated by the network. Although transactions can be validated instantly, it takes bitcoin miners approximately 10 minutes to record the payment within the blockchain and confirm it was not spent twice. In addition, transactions that pay a fee may be processed more quickly [14].

Bitcoin payment processing fees are optional and generally substantially lower than those of credit cards or money transfers. Currently, doing the work of payment processing is rewarded with newly created bitcoins. But this reward is halved every few years eventually phasing out all together when the total number of bitcoins have been released. Once the Bitcoin ceiling of 21 million units is reached, payment processing will only be incentivized with transaction fees.

Bitcoin functions using public-key cryptography, in which a user generates a pair of cryptographic keys: one public and one private. Only the private key can decode information encrypted with the public key; therefore the keys' owner can distribute the public key openly without fear that anyone will be able to use it to gain access to the encrypted information. An example public wallet (owned by the FBI) demonstrates its structure, a string of 34 numbers and letters (the private key, however, must be kept secret and secure) [15]. The public key can be used as an "address" to which other users can send bitcoins. Anyone wishing to use Bitcoin can create one or more Bitcoin addresses, which are collected and tracked in "wallets". Anyone can send bitcoins to the public address provided by the owner of the wallet, while the private key must be entered by the wallet owner to send bitcoins. Securing and protecting the private key is the essence of wallet security. If the private key for an address is not kept secret, the bitcoins may be stolen; theft has been documented on numerous occasions, and the practical day-to-day security of Bitcoin wallets remains an on-going concern.

Wallets allow a user to complete transactions between addresses by requesting an update to the blockchain, the public transaction log. Wallets come in a variety of forms: apps for mobile devices and computers, hardware devices, and paper tokens. When making a purchase with a mobile device, the use of QR codes to simplify transactions is ubiquitous.

2.6 NFC

Near field communication (NFC) is a set of standards for smartphones and similar devices to establish radio communication with each other by touching them together or bringing them into proximity, usually no more than a few inches.

Near Field Communication is used mostly in paying for purchases made in physical stores or transportation services. A consumer using a special mobile phone equipped with a smartcard waves his/her phone near a reader module. Most transactions do not require authentication, but some require authentication using PIN, before transaction is completed. The payment could be deducted from a pre-paid account or charged to a mobile or bank account directly [16].

Mobile payment method via NFC faces significant challenges for wide and fast adoption, due to lack of supporting infrastructure, complex ecosystem of stakeholders, and standards. Some phone manufacturers and banks, however, are enthusiastic.

Although the communication range of NFC is limited to a few centimeters, NFC alone does not ensure secure communications. In 2006, Ernst Haselsteiner and Klemens Breitfuß described different possible types of attacks, and detail how to leverage NFC's resistance to man-in-the-middle attacks to establish a specific key. Unfortunately, as this technique is not part of the ISO standard, NFC offers no protection against eavesdropping and can be vulnerable to data modifications. Applications may use higher-layer cryptographic protocols (e.g., SSL) to establish a secure channel.

NFC attack examples:
The RF signal for the wireless data transfer can be picked up with antennas. An attacker can typically eavesdrop within 10m and 1m for active devices and passive devices, respectively. With the use of a patch loop antenna it is possible to place a receiver close to the target and disguise it. This is much like ATM skimming in that it needs to be near the location however in this case no contact with the device or reader is required [17].

It is easy to destroy data by using a jammer. There is no way currently to prevent such an attack. However, if NFC devices check the RF field while they are sending, it is possible to detect attacks.

It is much more difficult to modify data in such a way that it appears to be valid to users. To modify transmitted data, an intruder has to deal with the single bits of the RF signal. The feasibility of this attack, (i.e., if it is possible to change the value of a bit from 0 to 1 or the other way around), is amongst others subject to the strength of the amplitude modulation. Transmitting Manchester-encoded data with a modulation ratio of 10% permits a modification attack on all bits. Because NFC devices usually include ISO/IEC 14443 protocols, the relay attacks described are also feasible on NFC. For this attack the adversary has to forward the request of the reader to the victim and relay back its answer to the reader in real time, in order to carry out a task pretending to be the owner of the victim's smart card [18].

Losing the NFC RFID card or the mobile phone will open access to any finder and act as a single-factor authenticating entity. Mobile phones protected by a PIN code acts as a single authenticating factor.

Lawfully opened access to a secure NFC function or data is protected by time-out closing after a period of inactivity. Attacks may happen despite provisions to shut down access to NFC after the bearer has become inactive. Additional features to cover such an attack scenario dynamically shall make use of a second wireless authentication factor that remains with the bearer in case of the lost NFC communicator. Relevant approaches are described as an electronic leash or its equivalent, a wireless key.

Conclusions
We presented our analysis for E-payment, as an example of security challenges in third-party service integration. We found serious logic flaws in leading merchant applications, popular online stores and payment providers (i.e., PayPal). We discussed the weaknesses in the systems that can compromise the customer's trust.

Web applications increasingly integrate third-party services. The integration introduces new security challenges due to the complexity for an application to coordinate its internal states with those of the component services and the web client across the Internet.

Online payments through are relatively safe because they use SSL technology which is the safest mechanism being used today or another secure methods (for example, using public-key cryptography). But the problem is the SSL protocol is not flawless, and users who see the yellow padlock at the bottom of the browser may get a false sense of security. Also there are always some flaws in security methods.

But in reality, the security of online payment also depends on the customer himself. He should gain knowledge in how to use the internet so that he can be more aware of email scams and website URLs that may not be from payment system website. For example for PayPal users, the lack of knowledge and common sense appears to have caused more problems than insecurity. However, there probably is no best way to be fully secured other than to just avoid online purchases altogether.

We believe that our study takes some steps in the security problem space that web applications have brought. In future work we are considering the security challenges that come with web service integrations in other scenarios, e.g., social networks and web authentication services, cancel, return flows. Fundamentally, we believe that the emergence of this new web programming paradigm demands new research efforts on ensuring the security quality of the systems it produces.

3. REFERENCES

[1] S. Murdoch and R. Anderson. “Verified by Visa and MasterCard SecureCode: or, How Not to Design Authentication”. Financial Cryptography and Data Security, Jan. 2010, pp. 42-45.

[2] “PayPal. PayPal - Data Security and Encryption”. Internet: http://www.paypal.com/cgi-bin/webscr?cmd=p/gen/security-outside [Dec. 10, 2013].

[3] 3. Rui Wang, Shuo Chen, XiaoFeng Wang, Shaz Qadeer. ”How to Shop for Free Online Security Analysis of Cashier-as-a-Service Based Web Stores”. Internet: http://research.microsoft.com/pubs/145858/caas-oakland-final.pdf [Dec. 1, 2013].

[4] 4. “The Secure Sockets Layer Protocol”. Internet: http://www.cs.bris.ac.uk/~bradley/publish/SSLP/chapter4.html [Nov. 22, 2013].

[5] 5. “SSL: Intercepted today, decrypted tomorrow”. Netcraft, pp. 10-12, May 25, 2013.

[6] 6. “SSL/TLS in Detail“. Microsoft TechNet, July 31, 2003.

[7] 7. “Description of the Secure Sockets Layer (SSL) Handshake“. Internet: http://www.support.microsoft.com [Dec. 1, 2013].

[8] 8. “Secure electronic transaction”. Internet: http://en.wikipedia.org/wiki/Secure_Electronic_Transaction [Dec. 12, 2013].

[9] 9. “The Secret PayPal Hack Method – 100% Guaranteed!”. Hack Expert, Nov. 11, 2003. Internet: http://www.astronomysight.com/_message/00000465.htm [Dec. 12, 2013].

[10] 10. “PayPal Email Scam – Web Site Version”. Internet: http://www.fightidentitytheft.com/paypal_scam.html [Nov. 25, 2013].

[11] 11. “PayPal Email Scam – Email Form”. Internet: http://www.fightidentitytheft.com/paypal_scam_email_form.html [Nov. 25, 2013].

[12] 12. “SearchSecurity.com”. Internet: http://searchsecurity.techtarget.com/sDefinition/0%2C%2Csid14_gci214006%2C00.html [Dec. 12, 2013].

[13] 13. “Bitcoin”. Internet: http://en.wikipedia.org/wiki/Bitcoin.html [Dec. 10, 2013].

[14] 14. “Bitcoins”. Internet: http://www.weusecoins.com/en/ [Dec. 8, 2013].

[15] 15. Alex Hern. “Bitcoin me: How to make your own digital currency”. Internet: http://www.theguardian.com/technology/2014/jan/07/bitcoin-me-how-to-make-your-own-digital-currency [Dec. 5, 2013].

[16] 16. “Near field communication”. Internet: http://en.wikipedia.org/wiki/Near_field_communication [Dec. 10, 2013].

[17] 17. Mike Clark. “Inside Secure adds sales agents”. Internet: http://www.nfcworld.com/2012/12/05/321436/inside-secure-adds-sales-agents, Dec. 5, 2012 [Dec. 10, 2013].

[18] 18. “NFC and Contactless Technologies”. Internet: http://nfc-forum.org/what-is-nfc/about-the-technology/ [Dec. 1, 2013].

INSTRUCTIONS TO CONTRIBUTORS

The *International Journal of Computer Science and Security (IJCSS)* is a refereed online journal which is a forum for publication of current research in computer science and computer security technologies. It considers any material dealing primarily with the technological aspects of computer science and computer security. The journal is targeted to be read by academics, scholars, advanced students, practitioners, and those seeking an update on current experience and future prospects in relation to all aspects computer science in general but specific to computer security themes. Subjects covered include: access control, computer security, cryptography, communications and data security, databases, electronic commerce, multimedia, bioinformatics, signal processing and image processing etc.

To build its International reputation, we are disseminating the publication information through Google Books, Google Scholar, Directory of Open Access Journals (DOAJ), Open J Gate, ScientificCommons, Docstoc and many more. Our International Editors are working on establishing ISI listing and a good impact factor for IJCSS.

The initial efforts helped to shape the editorial policy and to sharpen the focus of the journal. Started with Volume 8, 2014, IJCSS is appearing in more focused issues. Besides normal publications, IJCSS intend to organized special issues on more focused topics. Each special issue will have a designated editor (editors) – either member of the editorial board or another recognized specialist in the respective field.

We are open to contributions, proposals for any topic as well as for editors and reviewers. We understand that it is through the effort of volunteers that CSC Journals continues to grow and flourish.

IJCSS LIST OF TOPICS

The realm of International Journal of Computer Science and Security (IJCSS) extends, but not limited, to the following:

- Authentication and authorization models
- Computer Engineering
- Computer Networks
- Cryptography
- Databases
- Image processing
- Operating systems
- Programming languages
- Signal processing
- Theory
- Communications and data security
- Bioinformatics
- Computer graphics
- Computer security
- Data mining
- Electronic commerce
- Object Orientation
- Parallel and distributed processing
- Robotics
- Software engineering

CALL FOR PAPERS

Volume: 8 - **Issue: 3**

i. Submission Deadline : April 5, 2014 **ii. Author Notification:** May 5, 2014

iii. Issue Publication: May 2014

CONTACT INFORMATION

Computer Science Journals Sdn BhD

B-5-8 Plaza Mont Kiara, Mont Kiara

50480, Kuala Lumpur, MALAYSIA

Phone: 006 03 6204 5627

Fax: 006 03 6204 5628

Email: cscpress@cscjournals.org

CSC PUBLISHERS © 2014
COMPUTER SCIENCE JOURNALS SDN BHD
B-5-8 PLAZA MONT KIARA
MONT KIARA
50480, KUALA LUMPUR
MALAYSIA

PHONE: 006 03 6204 5627
FAX: 006 03 6204 5628
EMAIL: cscpress@cscjournals.org