

CRYPTANALYSIS ON DENIABLE ENCRYPTION

Yaga Bamunu Mudiyansele Sandaruwan Jayasinghe

(118213E)



University of Moratuwa, Sri Lanka.
Electronic Theses & Dissertations
Degree of Master of Science
www.lib.mrt.ac.lk

Department of Computer Science and Engineering

University of Moratuwa

Sri Lanka

May 2015

CRYPTANALYSIS ON DENIABLE ENCRYPTION

Yaga Bamunu Mudiyansele Sandaruwan Jayasinghe

(118213E)



University of Moratuwa, Sri Lanka.
Electronic Theses & Dissertations

Thesis submitted in partial fulfillment of the requirements for the Degree of MSc in
www.lib.mrt.ac.lk

Computer Science

Department of Computer Science and Engineering

University of Moratuwa

Sri Lanka

May 2015

“I declare that the work included in this report was done by me, and only by me, and this project report does not incorporate without acknowledgment any material previously submitted for a Degree or Diploma in any other University or institute of higher learning to the best of my knowledge. And to my belief it does not contain any material previously published or written by another person except where the acknowledgment is made in the text.

Also, I hereby grant to University of Moratuwa the non-exclusive right to reproduce and distribute my thesis, in whole or in part in print, electronic or other medium. I retain the right to use this content in whole or part in future works (such as articles or books)”.

Signature:

Date:



University of Moratuwa, Sri Lanka.
Electronic Theses & Dissertations
www.lib.mrt.ac.lk

“I certify that the declaration made above by the candidate is true to the best of my knowledge and she has carried out research for the Masters thesis under my supervision.”.

Signature of the supervisor:

Date:

Abstract

The notion of Deniable Encryption is a cryptographic primitive, which enables legitimate users to face coercion by dynamic adversaries without revealing true secret internals of the cryptosystem. Deniable Encryption provides a way to generate fake internals that correctly explain the cipher text.

When considering existing deniable schemes, two major variations can be found; schemes based on the concept of Deniable crypto-systems introduced by R. Canetti *et al.* and plausible deniable schemes. The schemes based on plausible deniability are not always depending on cryptographic systems, but rather use different approaches such as steganography or hardware level hidden volumes. With the objective of cryptanalysis, this research has been focused on deniable crypto-systems.

The existing deniable encryption schemes proposed provide different levels and types of deniability, which makes it difficult to find a common model for the cryptanalysis. Therefore, This research has narrowed down the cryptanalysis to full-sender-deniable encryption, which is the strongest notion in sender deniability.

In order to evaluate the real world implementation of full-sender-deniable encryption, this research has implemented a crypto-system using sparse-set. This research has also introduced a new type of sparse-set generation, which provides better performance compared to the two sparse-set generation methods proposed by Canetti *et al.*

Based on the common model of full-sender-deniable encryption, our cryptanalysis has been focused on three main areas; deniability limitation already given by Canetti *et al.*, statistical cryptanalysis and cryptanalysis based on faking algorithm. Since the encryption function of full-sender-deniable encryption is a public parameter, the adversary can coerce the sender to generate randomness by further faking and have additional data to detect the original faking. This is a new scenario that has been considered in this research, where it can be applicable in situation like rubber hose cryptanalysis.

Keywords: Deniable encryption, Cryptanalysis

Acknowledgment

First of all, I would like to offer my uppermost gratitude to my supervisor, Dr Chandana Gamage for his immense guidance, supervision and consistent encouragement through out the research work. I would also like to thank my MSc course coordinator Dr. Shehan Perera and research coordinator Dr. Malaka Walpola for their encouragement, insightful comments and guidance.

I wish to express my gratitude to all my MSc batch mates for their friendly encouragement given throughout the MSc program.

Last but not least, I would like to thank my family, especially to my wife for the encouragement, support and quite patience which was crucial for the completion of this thesis.



University of Moratuwa, Sri Lanka.
Electronic Theses & Dissertations
www.lib.mrt.ac.lk

Contents

Declaration	i
Abstract	ii
Acknowledgment	iii
Contents	iv
List of Tables	vii
List of Figures	viii
1 INTRODUCTION	1
1.1 Background	1
1.1.1 Violation of semantic security and coercion	1
1.1.2 Deniable encryption in practice	2
1.2 Research Problem	2
1.3 Objectives and Scope of the Research	2
1.4 Research Contribution	3
2 LITERATURE REVIEW	4
2.1 Evolution of Deniable Encryption	4
2.2 Review on Deniable Encryption	5
2.2.1 Shared key deniable encryption	9
2.2.2 Public key deniable encryption	10
2.2.3 Sender deniable encryption	10
2.2.4 Receiver deniable encryption	10
2.2.5 Bi-deniable encryption	11
2.2.6 Multi-distribution deniable encryption	11
2.2.7 Fully-deniable encryption	11

2.2.8	Plan-ahead deniable encryption	11
2.2.9	Plausible deniability	12
2.2.10	Publicly deniable encryption	12
2.2.11	Universal deniable encryption	12
2.2.12	Non-committing encryption	12
2.3	Applications of Deniable Encryption	13
2.4	Public Key Deniable Encryption Schemes	15
2.4.1	Schemes based on sparse set	16
2.4.2	Schemes based on samplable encryption	19
2.4.3	Schemes based on mediated RSA	23
2.4.4	Scheme based on simulatable encryption	25
2.4.5	Scheme based on indistinguishability obfuscation	28
2.4.6	Transforming sender deniable encryption to receiver deniable encryption or vice versa	32
2.4.7	Transforming a sender/receiver deniable encryption to bi-deniable encryption	32
2.5	Shared Key Deniable Encryption Schemes	33
2.6	Cryptanalysis	35
2.6.1	Ciphertext indistinguishability	35
2.6.2	Deterministic encryption vs probabilistic encryption	37
2.6.3	Malleability of encryption	37
2.6.4	Methods of cryptanalysis	37
3	IMPLEMENTATION OF FULL SENDER DENIABLE ENCRYPTION	40
3.1	Implementaion	40
3.1.1	Keygen	40
3.1.2	Sender	40
3.1.3	Receiver	43
3.1.4	Adversary	45
3.2	Sparse Set Generation using Probabilistic Encryption	48
3.3	Performance Comparison of the Implementations	49



University of Moratuwa, Sri Lanka.
 Electronic Theses & Dissertations
www.lib.mrt.ac.lk

3.3.1	Encryption	49
3.3.2	Decryption	49
4	CRYPTANALYSIS	52
4.1	Common Model for Full-Sender Deniable Encryption	52
4.2	Cryptanalysis Based on $1/n$ -deniability	55
4.3	Cryptanalysis Based on Statistics	56
4.4	Cryptanalysis Based on Faking Algorithm	58
4.4.1	Coercing faking algorithm	58
4.4.2	Detecting faking	59
4.4.3	Deriving true randomness	59
4.5	Side Channel Attacks	61
5	Conclusion	62
	References	65



University of Moratuwa, Sri Lanka.
 Electronic Theses & Dissertations
www.lib.mrt.ac.lk

List of Tables

2.1 Summary of the existing deniable encryption schemes 34



University of Moratuwa, Sri Lanka.
Electronic Theses & Dissertations
www.lib.mrt.ac.lk

List of Figures

2.1	Types of deniable encryption	9
2.2	Pseudorandom number generation	34
2.3	Encryption of shared key deniable encryption using pseudorandom generation	36
3.1	Number of PKC encryption for 1 bit of deniable encryption vs bit length of random V	50
3.2	Message length /Cipher-text ratio vs bit length of V	50
3.3	Number of PKC decryption for 1 bit of deniable encryption vs bit length of random V	51
4.1	Sender's view vs adversary's view of the encryption	54
4.2	Sender faking as shared key encryption	55
4.3	Faking against random number generator	57



University of Moratuwa, Sri Lanka.
Electronic Theses & Dissertations
www.lib.mrt.ac.lk

1 INTRODUCTION

1.1 Background

1.1.1 Violation of semantic security and coercion

The notion of semantic security is based on the limitations of the adversary's capability to interfere the communication between the sender and the receiver. The confidentiality of the encryption depends on the secret keys/randomness owned by the legitimate parties involved in the communication where the adversary does not have access to the private internals of the sender and/or the receiver. If the keys/randomness is compromised, the security of the communication will also get compromised. Hence, the legitimate parties are responsible for keeping the private keys/randomness without exposing to untrusted parties.

However, this is not a valid assumption in current practice. In scenarios like rubber hose cryptanalysis, the adversary is capable of extracting private internals directly from the sender and/or the receiver by coercion. Moreover, one of the key characteristics of the semantic security is the committing nature with the internals of encryption/decryption where legitimate parties are bound to the keys been used. This committing nature of encryption/decryption is problematic in situation like e-voting/e-auction systems when an adversary is capable in forcing the legitimate parties to reveal their internals.

Deniable encryption is a strong notion, which enables the legitimate parties to provide fake internals to preserve the confidentiality of the communication. The main objective of the deniable encryption is to survive at coercion by exposing fake internals with a valid explanation for the cipher text generated.

1.1.2 Deniable encryption in practice

Consider the scenario where Alice is a secret government agent who has an alliance with a terrorist named Bob. Alice was working on a mission to arrest terrorist leader Carl and was using Bob's support to retrieve the evidence against Carl. The communication between Alice and Bob was done via encrypted channels over the agency network where all the communication logs including encrypted data were stored securely in agency storage. The mission was successfully completed and the terrorist leader has been taken into the custody. However, Alice had agreed with Bob that she would not reveal the secret alliance. After few years, Carl was released from the prison and becomes the minister of the same government agency. Now Carl is capable of coercing Alice to reveal the internals of the encryption to retrieve the Bob's identity from the stored cipher.

Deniable encryption is one of the strong notions of cryptographic security that can be used in above situation where Alice or/and Bob can give fake internals to generate fake messages for the adversary.



University of Moratuwa, Sri Lanka.
Electronic Theses & Dissertations
www.lib.mrt.ac.lk

1.2 Research Problem

In the last few decades, number of deniable encryption schemes have been introduced and cryptanalysis of those schemes was considered by practitioners as an interesting research problem. Based on a wide ranging literature survey done, it appears that a comprehensive cryptanalysis has not been carried out on the existing deniable encryption schemes.

1.3 Objectives and Scope of the Research

The main objective of the research is to put forward the cryptanalysis on full sender deniable encryption. To accomplish the main objective, following steps were completed at the initial stage of the research.

- Studying and analyzing the existing deniable encryption schemes

- Studying on cryptanalysis
- Identifying the similar properties to derive higher-level abstraction that will be a general model of full sender deniable encryption schemes
- Identifying possible cryptanalysis on the common model of sender deniable encryption

With the knowledge gained at initial stage, it was realized that though there are number of schemes proposed, complete practical implementation of full-sender-deniable encryption has not been proposed. Therefore, rest of the research was completed with below objectives,

- Providing full-sender-deniable encryption implementation based on the parity based scheme proposed by Canetti *et al.* [1]
- Evaluating the implementation to achieve further cryptanalysis

1.4 Research Contribution



University of Moratuwa, Sri Lanka.
 Electronic Theses & Dissertations
www.lib.mrt.ac.lk

This research work presented in this thesis has successfully crypt-analysed the full sender deniable encryption introduced by Canetti *et al.* [1].

2 LITERATURE REVIEW

2.1 Evolution of Deniable Encryption

The notion of deniable encryption was first explored in detail by Ran Canetti, Cynthia Dwork, Moni Naor, and Rafail Ostrovsky in 1996 [1]. The authors have introduced different notions of deniable encryption including sender deniability, receiver deniability and bi-deniability. They have proposed a mechanism to transform a sender deniable scheme into a receiver deniable scheme and vice versa. They have also proposed a method to transform a sender/receiver deniable scheme to a bi-deniable scheme.

M. Dürmuth and D. M. Freeman have proposed a method to obtain a deniable encryption scheme using samplable encryption [2], in 2011. They have constructed two encryption schemes based on quadratic residuosity and trapdoor permutation. Based on samplable encryption [2], another deniable encryption scheme [3] was proposed by B. M. David and A. C. A. Nascimento in 2011. The scheme is based on McEliese cryptosystem.

A. O'Neill, C. Peikert and B. Waters have proposed a bi-deniable encryption scheme [4] which provides security against coercing of both sender and receiver, simultaneously. The constructed scheme is non-interactive and does not require any third party involvement.

M. H. Ibrahim has introduced another sender deniable encryption scheme [5] based on quadratic residuosity of two primes in 2008. He has also proposed a new approach to derive a sender deniable public key encryption scheme from any trapdoor permutation.

Another receiver deniable encryption scheme [6] based on mediated RSA [7] and

oblivious transfer [8] was proposed by M. H. Ibrahim. This is an interactive construction where a separate entity called a security mediator should be present.

In 2013, A. Sahai and B. Waters have derived a deniable encryption [9] using punctured programs which is based on indistinguishability obfuscation. Using the concept of punctured programs [9], the proposed scheme provides full sender deniability. They also have introduced two new notions of deniability called publicly deniable encryption and universal deniable encryption.

2.2 Review on Deniable Encryption

In the current context of deniable schemes, two main variations can be identified as the schemes based only on cryptographic primitives and the schemes that provide plausible deniability derived from non-cryptographic primitives.

The notion of cryptographic deniable encryption that was introduced by Canetti *et al.* [1] is based on providing fake internals to prevent further coercion. The sender generates cipher text C from the plaintext M using public key K_{pk} and randomness R_s . The receiver derives the M using the secret key K_{sk} and R_r . The underlying goal of the deniability encryption is to generate fake M'_s (at sender) and M'_r (at receiver) that can be opened with same cipher text C while using fake keys R'_s and K'_{sk} . When considering sender deniable encryption, only M'_s and R'_s are used for the faking. In receiver deniable encryption schemes, only M'_r and K'_{sk} are used for the faking. In bi-deniable encryption schemes, M'_s , R'_s , M'_r and K'_{sk} are used for faking. If the coercer is capable to force the sender and the receiver simultaneously, it should be possible to produce same fake message (i.e. $M'_s = M'_r$) at both ends.

One of the fundamentals requirement of deniable encryption is that the coercible parties should be able to generate fake messages (M_s and/or M_r) with the content of limited number of variations. In practical context, the fake messages should be meaningful text that will satisfy the coercer. If the fake messages are dummy

messages with no meaning related to the situation/application, the coercer does not have any means to believe the coercible party. This can be considered as the basic proof for the coercer to ensure the thrust worthiness of the coercible parties.

Another property involves with deniable encryption is that when is it possible to generate the fake messages and keys. The best option is to generate the fake messages at the point of coercion where the coercible parties have the control over the faking based on the present situation. However, some of particular deniable encryption allows the coercible parties to generate the fake message only at initial stage. Generating fake messages and keys at the initial stage of encryption is known as plan-ahead deniability. Though this is a weaker notion of deniability, in practical applications like e-voting, where number of possible or required plain text messages are limited, it provides adequate security against coercion.

Another key discussion area of deniable encryption is that what parameters can be changed at faking. The best option is changing only the keys/randomness. If we consider a sender (same for the receiver as well) with the encryption algorithm E and cipher text C generated as $C = E(M, K_{pk}, R_s)$, fake M'_s and R'_s should be generated such that $C = F(M'_s, K_{pk}, R'_s)$. Though one can simply change two parameters M_s and R_s to achieve the same C , he/she do not have complete flexibility to generate M'_s . As discussed above, M'_s has limited number of options or is a constant value depending on the application scenario to satisfy the adversary. The deniability achieved only by changing the key/randomness is known as fully deniability and construction is significantly difficult than multi-distributional deniability. With multi-distributional deniability, separate fakable algorithm E' is used. The obvious question is that why adversary believes that coercible parties have used non-fakable encryption. Thereby in practice, the adversary should not be aware of the fakable encryption to defend the coercion successfully. With this setup, coercible parties can use fakable algorithm E' for the encryption/decryption, while presenting non-fakable algorithm E for the adversary with a fake message and a fake keys.

Another variation of multi-distributional deniable encryption is that hiding the real values inside the fake plain text presented to the adversary. The simple example for this kind of encryption scheme is having double encryption/decryption. Another example can be demonstrated using El-Gammal [10] PKE. The cipher text of El-Gammal has two parts: g^r and $m.y^r$ (where g is the generator, m is the message, r is a random value and y is the public key). Deniable encryption proposed by Marek *et al.* [11], the real message is hidden inside the g^r instead of m using symmetric key algorithm such as AES and the fake message is used as the m . At a coercion, the receiver can reveal key related to El-Gammal encryption while preserving the privacy. However, if the adversary is capable of knowing the real internal implementation of encryption/decryption, the scheme does not provide deniability.

When considering plausible deniability, two variations can be identified. In first variation, the deniability is achieved by hiding the real data inside the dummy data. This is used in practical deniable storage and file system. One example is TrueCrypt [12], which is an on-the-fly disk encryption tool for Window, Mac and Linux. However, this provides weaker deniability as if the coercer knows the implementation of the system, the coercion will be continued until adversary gets the real messages and keys. The second method is the use of cryptographic primitives to provide the plausible deniability. In this case, coercible parties provide a proof for the adversary that they do not have any secret information (although they have the secrets) other than what has been revealed. Therefore, it is possible to convince the adversary that further coercion will not provide any additional information.

Most of the existing deniable schemes proposed are based on single bit encryption (bit-by-bit encryption). Thereby, we only have to focus on two plain texts values (i.e. 1 or 0) that will be used to generate any real/fake message. However, most of the attacks including statistical attacks are based on encryption of multiple bits.

Another concern with deniable encryption is the efficiency of the encryption and decryption. The efficiency can be given as lengthiness of cipher text, time taken for

encryption/decryption and processing of the schemes. In addition to that, analyzing these parameters leads to attacks like side channel attacks [13] that will compromise the security provided by the deniable encryption scheme. This is an important concern in multi-distributional deniable encryption, because by analyzing the system/device specific parameters like power consumption or processor/memory usages, adversary may be able to separate fakable algorithm from non-fakable algorithm.

When considering proposed deniable encryption schemes in current context, most of them are based on public key infrastructure. This may be due to the inherent advantages, like simplicity of key distribution, provided by PKE than symmetric key encryption schemes. However, symmetric key deniable encryption schemes are more useful in practical applications like deniable storage systems or files systems, because of the inherent efficiency of encryption and decryption.

Rikke *et al.* [14] have given lower and upper bounds for different notions of deniable encryption. There, the security of the deniable encryption was defined as the infeasibility of separating real encryption from fakable encryption. Two security levels has been proposed: polynomial security and negligible security. With polynomial security, the security is given as $1/p$ where p is the polynomial of security parameter such as key length. In contrast to polynomial security, negligible security does not depend on security parameter, where it provides stronger deniability. According to Rikke *et al.* [2], security of any non-interactive receiver deniable encryption scheme is bounded to polynomial security. Depending on that, it was deducted that any non-interactive bi-deniable encryption which is better than polynomial security, is impossible to construct. For receiver deniable encryption and bi-deniable encryption, the lower and upper bounds were defined using key length and for sender deniable encryption only upper bound was provided. If same length of secrete key and public key are considered, bi-deniable schemes have lowest upper bound and sender deniable schemes have highest upper bound.

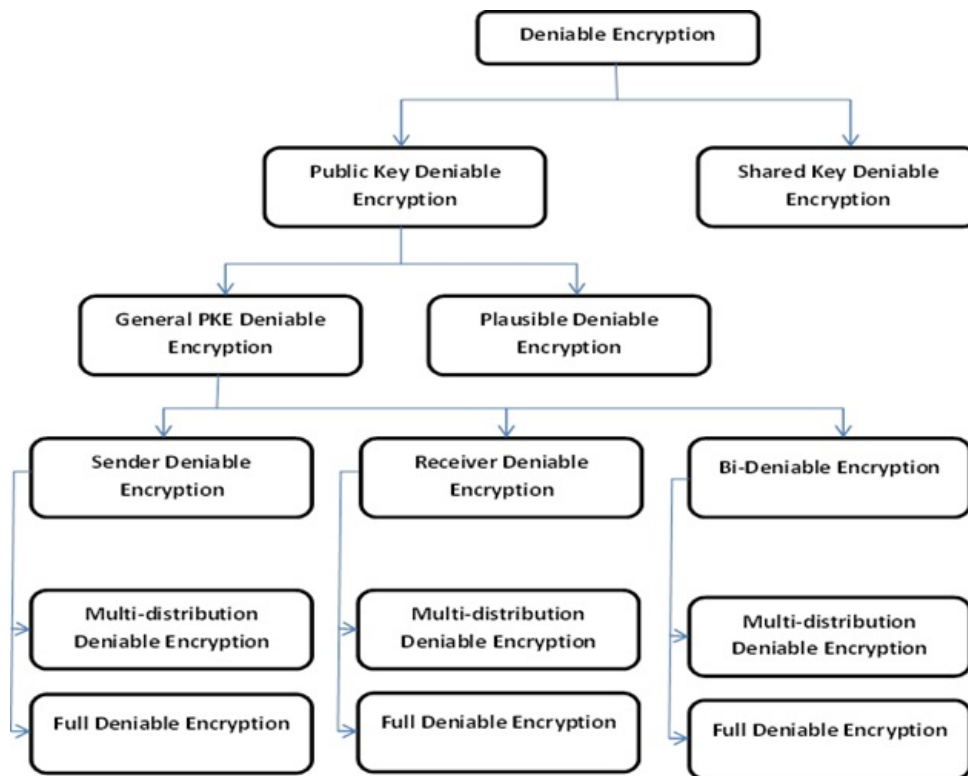


Figure 2.1: Types of deniable encryption

2.2.1 Shared Key deniable encryption

A deniable encryption scheme is a shared key $\delta(n)$ -sender-deniable encryption[1], if the scheme has below three properties with the inherent characteristics of shared key encryption. Here, n is the security parameter.

- Correctness - The message sent by the sender and the message retrieved by the receiver differ with negligible probability.
- Security - The communication (between the sender and the receiver) of two messages m_1 and m_2 computationally indistinguishable. This can be given as $com(m_1) \sim com(m_2)$.
- Deniability - The adversary's view of honest encryption/decryption and the adversary's view of fake encryption/decryption are differed with $\delta(n)$ probability.

2.2.2 Public key deniable encryption

A deniable encryption scheme is a shared key $\delta(n)$ -sender-deniable encryption[1], if the scheme has below three properties with the inherent characteristics of PKE. Here, n is the security parameter.

- Correctness - The message sent by the sender and the message retrieved by the receiver differ with negligible probability. This can be given as $D(E(m)) = m + \delta$ where δ is negligible.
- Security - The communication (between the sender and the receiver) of two messages m_1 and m_2 computationally indistinguishable. This can be given as $com(m_1) \approx com(m_2)$.
- Deniability - The adversary's view of honest encryption/decryption and the adversary's view of fake encryption/decryption should be differed with $\delta(n)$ probability.

2.2.3 Sender deniable encryption

With sender deniable encryption, the sender can provide fake encryption keys/randomness that explains the cipher text for the adversary. In particular, the adversary is capable only to coerce the sender. With public key sender deniability, only randomness can be faked at encryption where encryption key is a public parameter.

2.2.4 Receiver deniable encryption

With receiver deniable encryption, the receiver can provide fake keys/randomness that explains the transferred cipher. In contrast with public key sender deniable encryption, the public key receiver deniable encryption allows the receiver to fake the secret key in addition to randomness used.

2.2.5 Bi-deniable encryption

Bi-deniable encryption can be used in situation where both sender and the receiver are susceptible to coercion. Both the sender and the receiver will be able to fake their internal states against the coercion. In the case of simultaneous or coordinated coercion, the sender and the receiver should be capable of revealing same fake messages in a coordinated manner.

2.2.6 Multi-distribution deniable encryption

With multi-distribution deniable encryption[1], if the sender/receiver expects to fake the messages upon the coercion, he/she should use the fakable algorithm. If faking is not required, they can use non-fakable algorithm. The question raised with this scenario is why adversary believes that sender/receiver has used non-fakable algorithm. Thereby in practice, though a fakable encryption is used for the encryption/decryption, the sender/receiver always reveals non-fakable algorithm with fake/non-fake internals and the adversary should not be aware about fakable algorithm. For deniability, E_r should generate a cipher-texts such that $E_r(m) = E(m_f)$ where m is the real message and m_f is the fake message.

2.2.7 Fully-deniable encryption

In contrast to multi-distributional deniability, fully-deniable encryption[1] does not use two separate algorithms. Since revealing internals to the adversary does not provide additional knowledge, full-deniability is the ideal solution for deniability applications.

2.2.8 Plan-ahead deniable encryption

With plan-ahead deniable encryption[1], the fake messages and the relevant fake keys will be generated at the initial stage of encryption. It is considered as a weaker notion of deniable encryption. Though plan-ahead deniable encryption has this limitation, it provides sufficient security for the applications like e-voting systems where number of possible (or required) messages is limited in number.

2.2.9 Plausible deniability

In this notion of deniability, the sender or/and the receiver deny the fact of having knowledge of additional secret information. The coercible party gives a proof to the adversary that he/she does not have hidden internals which, provides additional advantage to break the security of the cryptographic system. Therefore, further coercion is not useful.

2.2.10 Publicly deniable encryption

With the general construction of deniable encryption, the sender is supposed to remember the true randomness to generate fake randomness. However with publicly deniable encryption, the sender do not have to remember the true randomness used to generate cipher text. This notion of publicly deniable encryption was introduced by Amit Sahai and Brent Waters [9]. According to the definition [9] of publicly deniable encryption, it implies another strong characteristic. With publicly deniable encryption, any party who has the cipher text and the public parameters, can generate a randomness that satisfy the encryption of selected message.



University of Moratuwa, Sri Lanka.
Electronic Theses & Dissertations
www.lib.mrt.ac.lk

2.2.11 Universal deniable encryption

Universal-deniable [9] schemes use the existing public key infrastructure without asking to re-obtain the keys that are already taken by the parties involved with the communication. The most of the deniable encryptions proposed are based on existing public key infrastructure such as RSA. Therefore, this is a strong notion that enhances the usability in the practical implementations.

2.2.12 Non-committing encryption

Non-committing encryption [15] is another related notion of deniable encryption with the common objective of providing security against adaptive adversaries. However, compared to bi-deniable encryption, it is considered to be a weaker notion of deniability [1, 2].

With traditional encryption schemes, the sender/receiver is not able to fake the encrypted data from the transmitted data over the non-secure channel. The main goal of the non-committing encryption is to remove the inherent committing characteristic of the traditional encryption schemes.

In non-committing cryptosystem, the entity called simulator is capable of generating dummy cipher text that is indistinguishable (to the adversary) from the real cipher text. The dummy cipher text can be opened as a fake message. In contrast to the deniable encryption, only the simulator can generate the dummy cipher text and it is not a must to open the dummy cipher text as a meaningful plaintext. However, a primary objective of deniable encryption scheme is to generate meaningful fake messages from the cipher text.

2.3 Applications of Deniable Encryption

One of the main applications of deniable encryption is e-voting systems [16]. In non-electronic voting, the voting booth provides the privacy by the physical arrangements and the voting does not generate receipts. However, in electronic voting schemes, there are number of tracing mechanisms inherently possible due to the nature of IT based implementations. Thereby, the possibility of making successful coercion is significantly higher in e-voting system compared to non-electronic voting system. In this case, coercion can be occurred in one or both of two steps: forcing before the voting and forcing after the voting to provide the proof of loyalty. Deniable encryption can provide strong solution against the second scenario of coercion in e-voting systems.

Another application of deniable encryption is e-auctioning [16], which is considered to be sealed-auction. In contrast to e-voting, a receipt should be generated where the receipt have to be present as the proof of the bid. Though e-auctioning and e-voting have different encryption requirement, both have same deniability requirements.

Common scenario in e-voting (or in e-auctioning) is vote selling where the voters are agreed/committed to vote for a particular entity. However, the voter may not adhere the commitment and may vote differently. After the voting, the adversary may request/acquire proof of the voting. For an example, adversary can collect encrypted ciphers by eavesdropping and asks the voters to give explanation of the cipher. On the other hand, the adversary may be able to force the voting authority directly to collect voting receipts. To prevent coercion [16] the implementation of e-voting (or e-auctioning) should provide three main notions; secure booth, un-tappable channel and security against colluded voting authority. Similar to the physical booth, the logical booth in e-voting system provides the privacy for the voter to cast their voting without the adversary's observation and the encryption schemes like receipt free encryption and anonymous encryption provide the security against the colluded voting authority. The un-tappable channel provides the security against having accesses to communication between the sender (the voter) and the receiver (the central authority who processing the votes). Achieving un-tappable channel is difficult. However, the deniable encryption can be used to achieve same security requirement. With deniable encryption, even adversary have access to the communication channel, the coercer cannot gain additional advantage for the vote verification.

Deniable encryption also provides a strong solution for preserving privacy of storage systems. In some countries, different policies are maintained for privacy of storage systems. Therefore, the level of security/privacy is based on the Key Disclosure Law(i.e - Mandatory Key Disclosure[17]) of the country. With this, any person is legally bound to reveal the keys/internals to law enforcement authority. Most importantly, the law is based on countries general law and differs from country to country. Therefore, one may have to reveal all the content in his/her hard disk at the airport when entering/leaving a country. In these situations, storing data encrypted with deniable encryption scheme may preserve the confidentiality of the data in the storage system.

Another application area of deniable encryption is cloud based storages. The cloud environments is getting increasingly popular where it provides storage, processing and bandwidth with a low cost. The encryption of transferred data is vital factor to preserve the security of the data in cloud environment. However, the encrypted data in the cloud is readily available for the cloud provider and multiple parties can access the same data. Therefore the possibility of coercion is high compared to an environment with dedicated resources. For example, instead of searching individual data storages, a legal authority can coerce the cloud provider to provide bulk data and access each individual. Therefore, deniable encryption can be used to provide strong security for the cloud based transactions and storages. To cater this requirement, Gasti *et al.* [18] have proposed a deniable encryption scheme which is optimized for the cloud storage.

The publicly deniable encryption [9] can also be a strong notion in case of losing the randomness/keys/message used for the encryption. With standard public key encryption schemes, if the sender has lost the random values used at the encryption, he/she cannot regenerate same cipher text. The sender even cannot provide the true message/randomness he/she used. With publicly deniable encryption, the sender can generate the same cipher text without remembering the real randomness/message used at the encryption.

2.4 Public Key Deniable Encryption Schemes

Considering the existing PKE deniable encryption schemes, they are based on different cryptographic primitives such as sparse set, samplable encryption and oblivious transfer. Scheme one, two and three that described below are based on sparse set. The concept of sparse set is based on trapdoor permutation. The scheme four, five, six and seven are based on samplable encryption. Dürmuth Markus and Freeman David Mandell have derived a deniable encryption scheme based on any samplable encryption scheme. One important observation of deniable encryption schemes based on samplable encryption schemes is that the correctness is less than

100%. Thereby, the probability of varying the sender's input (real plaintext) from receiver's output is negligible in single communication. The communication should be repeated multiple times to achieve the higher correctness at the receiver.

Canetti *et al.* [1] have introduced five deniable encryption schemes, three PKE deniable scheme based on trapdoor one-way permutation and two based on shared key encryption. In public key encryption, the sender and the receiver do not have shared information while in shared key encryption, two parties share a secret where adversaries do not have any prior knowledge. In addition, the paper defines simple mechanism to convert sender deniable encryption scheme to receiver deniable encryption scheme and vice versa.

2.4.1 Schemes based on sparse set

The public key encryption introduced by Canetti *et al.* [1] is based on sparse set.



There exists $S \subset \{1,0\}^t$ and trapdoor function d such that,

$|S| < 2^{t-k}$, where k is significantly large and less than t (bit length of S).

$x \in S$ can be easily generated without trapdoor information

If d is given, it is easy to decide whether $x \in S$ or not. If d is not given, it is infeasible to decide whether $x \in S$ or not.

Two sparse set constructions were proposed by Canetti *et al.* [1].

1. Construction 1

Select $x_i = \{0,1\}^s$ s.t $B(f^{-1}(x_i)) = 0$ where f is a trapdoor permutation and function $B: \{0,1\}^s \rightarrow \{0,1\}$

Construct $x = x_1 \dots x_k$.

Define $S = \{x \in \{0,1\}^t \mid \forall i = 1 \dots k, B(f^{-1}(x_i)) = 0 \text{ where } t = s \cdot k$

Here, $|S| = 2^{(s-1)k} = 2^{(t-k)}$

2. Construction 2

Select $x_0 = \{0,1\}^s$

Construct $x = x_0 b_1 \dots b_k$ s.t $b_i = B(f^i(x))$ where f is a trapdoor permutation and function $B: \{0,1\}^s \rightarrow \{0,1\}$

Define $S = \{x \in \{0,1\}^t \mid \forall i = 1 \dots k, B(f^{-i}(x_0)) = b_i \text{ where } t = s + k$

Here, $|S| = 2^{(s)} = 2^{(t-k)}$

- Scheme 1 : Basic scheme based on sparse set

To encrypt 1, x is selected as $x \in S$ and a random $x \in \{1, 0\}^t$ is used to encrypt 0. To decrypt the cipher text, the receiver detects S element using trapdoor and considers S elements as 1. Others will be detected as 0. To open honestly, true random choices used for the encryption can be revealed. To open dishonestly, if 1 was encrypted, the sender can claim $x \in S$ as $x \in \{1,0\}^t$ and can fake successfully. However, to give 0 as 1, the sender has to give a random value as a S element. But, $x \in S$ is happened only with negligible probability of 2^{-k} .

Therefore, if 0 was encrypted, lying is not feasible.

- Scheme 2: Parity based scheme based on sparse set

The vector V is defined as $V \in \{S, R\}^n$, where S and R are two sets with bit length of t (S is an element of the sparse set and R is a random where selected R is in the sparse set with negligible probability). To create the vector V , the sender selects a random $V' \in \{1,0\}^n$. If the j^{th} element of V' is 1, the sender uses a random S element and if the j^{th} element of V' is 0, then use a random element R . To encrypt 0, a random even i is selected and the sender sends a V with i number of S elements. To encrypt 1, random odd i is selected and a V with i number of S elements will be sent to the receiver. For decryption, the receiver obtains the parity of the number of S elements in the encrypted text. For honest decryption, the sender can reveal the true random choices used to generate V and the true random values used to generate S elements. For faking, the sender is able to claim the selected i as $(i - 1)$ by giving any S element as R element and is able to open the cipher text as any fake plain text. For encryption of 1, the true distribution of $r = i$ is $1,3,5,\dots,n$ and the fake $r = i-1$ distribution

is $0, 2, 4, \dots, n-1$. Therefore, opening 1 as 0 is undetectable. For an encryption of 0, the true distribution of $r = i$ is $0, 2, 4, \dots, n-1$ and the fake $r = i-1$ distribution is $-1, 1, 3, \dots, n-2$. Thereby, opening 0 as 1 is feasible for all but not with $i = 0$. In this scheme, $i = 0$ happens only with $1/n$ probability and the security of the encryption depends on n , number of S/R elements used to generate cipher text V . Therefore, the scheme is given as $(1/n)$ -deniable encryption and it only provides polynomial security.

- Scheme 3: Undetectable parity scheme based on sparse set

By defining this scheme, Canetti *et al.* [1] have introduced the notion of deniability called multi-distribution deniability. As mentioned above, multi-distribution deniable encryption schemes use a separate algorithm to generate fakable encryption. According to the scheme, four distributions are defined,

$$T_0 = \{R, R\}, \quad T_1 = C_1 = \{R, S\}, \quad C_0 = \{S, S\}$$

Here, S is an element of the sparse set and R is a random element which can be a member of the sparse set with negligible probability.

– Encryption :

- * Non-fakable encryption: Select an element from T_0 to encrypt 0 and select an element T_1 to encrypt 1

- * Fakable encryption: Select element from C_0 to encrypt 0 and select an element C_1 to encrypt 1

– Decryption:

By getting the parity of the number of elements in sparse set(S), one can find the plain text message.

– Deniability

- * Honest way: The sender can reveal the true random choices used in the sparse set.

- * Dishonest way: If the true value is from C_0 , the sender can claim that it is from T_1 (inverse of the true value) or from T_0 (same as the true value). If the true value is from C_1 , the sender can claim it is from T_0 (inverse of the true value) or from T_1 (same as the true value).

2.4.2 Schemes based on samplable encryption

Samplable encryption

With samplable encryption scheme [2], there are three algorithms defined: $\text{RandomCT}(s)$, $\text{SampleEncRand}(sk, c)$ and $\text{SampleCTRand}(pk, c)$.

RandomCT: The function takes a random value and the public key as the inputs and generates the output that will be indistinguishable from cipher text generated by encrypting any other message.

SampleEncRand: The function takes cipher text and relevant secret key as the inputs and generates output of fake randomness that will be indistinguishable from the real randomness used for the encryption.

SampleCTRand: The function takes the public key and the cipher text c as the input and generates a random element s . s will be statistically indistinguishable from input of RandomCT that will generate the same cipher text c .

Key Generation: The receiver generates $4n + 1$ number of public and secret key pairs and sends the public keys to the sender.

Encryption: To encrypt bit b , the sender has to choose $4n + 1$ indexes and group them into three sets A, B, C such that,

A = $n + 1$ number of indexes selected randomly

B = n number of indexes selected randomly

C = $2n$ number of indexes selected randomly

Using the key generation of the underlying standard public key encryption scheme (the samplable semantically secure encryption scheme used to derive the deniable

encryption scheme), $4n + 1$ key pairs of public and secret keys will be generated. Then the sender generates $n + 1$ number of encryptions of message b using public keys relevant to indexes in A and n number of encryptions of message $1 - b$ using public keys relevant to indexes in B. Using RandomCT, the sender generates random cipher text and map them to the indexes of C. Finally, all encryptions (related to A, B and C indexes) are sent to the receiver.

Decryption: Each cipher text will be decrypted using the secret keys and the receiver retrieves the majority of the decrypted message. Dürmuth Markus and Freeman David Mandell [2] have proved the correctness of the decryption is greater than $(1/2 + 1/(5\sqrt{n}))$. To achieve higher accuracy, the encryption/decryption should be repeated for single bit transfer.

Deniability: After decrypting each message, the receiver selects $n/2$ number of pairs of indexes such that each index pair is mapped with opposite messages. These pairs of indexes are sent back to the sender. The sender selects one pair of index (out of $n/2$ pairs), such that one index in A and other in C. If such a pair is not found, the sender and the receiver have to repeat the above initial steps. However, the probability of not finding such a pair is negligible. The sender sends the selected index pair to the receiver and the receiver sends corresponding shared keys to the sender.

By using cipher text of the index in A as the input to SampleCTRand, the sender generates random element s_1 . By using the cipher text of the index in C as the input to SampleEncRand, the sender generates randomness r_1 .

Upon coercion, the sender can reveal real A indexes as fake B indexes and real B indexes as fake A indexes to fake the message b . To have the additional index that should be given in fake A, the sender can use the randomness r_1 that has derived above. r_1 will generate same cipher text in real encryption. To remove the additional index in fake B (real A), the index relevant to s_1 is mapped to fake C.

While introducing the above basic scheme, two initiations [2] have also been proposed where one is based on quadratic residuosity and other is based on trapdoor permutation.

- Scheme 4: Scheme based on samplable encryption derived by quadratic residuosity [2]

Quadratic residuosity: Let an integer $N = pq$ where p and q are two odd primes and select two sets $J(N)$ and $Q(N)$ s.t,

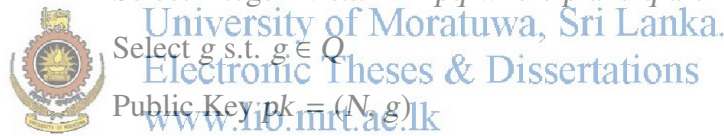
$$J(N) = \{x \in Z_N^* : (\frac{x}{N})=1\}$$

$$Q(N) = \{x = a^2 : x, a \in Z_N^*\}$$

$$P(N) = J(N)/Q(N)$$

According to Quadratic residuosity assumption, without the knowledge of p or q , $Q(N)$ and $P(N)$ are computationally indistinguishable.

– Key Generation :



Select integer N s.t. $N = pq$ where p and q are n bit primes
 Select g s.t. $g \in Q$
 Public Key $pk = (N, g)$
 Secret Key $sk = p$

– Encryption :

Select r randomly from Z_N^*

Cipher text $c = g^b r^2 \pmod{N}$

– Decryption :

If $(c/p) = 1 \rightarrow b' = 1$

Else $b' = 0$

– RandomCT

Find x s.t. $(x/N) = 1$ where x is select randomly from Z_N^*

– SampleEncRand

If $b' = 0 \rightarrow X^2 = c \pmod{N}$

If $b' = 1 \rightarrow X^2 = c/g \pmod{N}$

- SampleCTRand

Select random elements x_i randomly from Z_N^* .

Find L smallest index of i s.t. $(x_i/N) = 1$

Give output, a sequence of randomness (x_1, \dots, x_{L-1})

- Scheme 5: Scheme based on samplable encryption derived by trapdoor permutation [2]

The trapdoor consists of an algorithm Samp that extracts an element randomly and uniformly from a given domain.

- Key Generation: Function f is sampled from a family of trapdoor permutations with canonical domain sampling [19].

$f: R \rightarrow R$

Define $H: R \rightarrow \{1,0\}$

$g = f^{-1}$

Public key $pk = (f, H)$



Secret key $sk = g$

Electronic Theses & Dissertations

www.lib.mrt.ac.lk

Select r randomly from R using Samp

Cipher text $c = (f(r), H(r) \oplus b) \in R \times \{1, 0\}$

- Decryption

Find y and z s.t. $c = (y, z) \in R \times \{0, 1\}$

$b' = H(g(y))$

- RandomCT

Using Samp, Select s randomly from R

Find b randomly from $\{1,0\}$

Output (s, b)

- SampleEncRand

Find y s.t. $c = (y, z)$

Output $g(y)$

– SampleCTRand

Find y and z s.t. $c = (y, z)$

Output (y, z)

- Scheme 6: Encryption scheme based on samplable encryption derived by McEliece assumption [3]

Using the concept of deriving deniable encryption scheme from samplable encryption [2], B.M. David and A.C.A. Noscimento have proposed a new scheme that is based on McEliece assumption [3].

2.4.3 Schemes based on mediated RSA

A different approach for creating receiver deniable scheme was introduced by M. H. Ibrahim [6] and the scheme is based on mediated RSA and oblivious transfer.

Mediated RSA

This is an extension of RSA where separate entity called SEM (SEcurity Mediator) involves with the communication.



University of Moratuwa, Sri Lanka.
Electronic Theses & Dissertations
www.lib.mrt.ac.lk

Key Generation : Certificate authority generates the modulus N and a key pair e (public key) and d (private key). In contrast to general RSA, d is segmented into two components.

$$d = (s + r) \bmod(\phi(N)) \text{ where } \phi(N) \text{ is the Euler totient of } N$$

Keys will be distributed as e to sender, s to SEM and r to receiver.

Encryption :

The sender generates cipher text $c = m^e \bmod(N)$ where m is the message

Decryption :

Receiver sends the c to SEM.

SEM generates, $m_s = c^s \bmod(N)$

SEM sends m_s to receiver

Receiver retrieves the message as,

$$m_r = c_r \text{ mod}(N)$$

$$m = m_s m_r \text{ mod}(N)$$

Oblivious transfer

Oblivious transfer [8] is a primitive of cryptographic security where the sender communicates one of the message from possible set of message to the receiver obliviously. The sender does not know that which message was communicated to the receiver.

Correctness: The message communicated to the receiver should be valid data of the sender. This means, the message should be one of possible messages in the sender's domain.

Chooser's privacy: The sender or third party should not get any information about the message captured by the receiver.

Sender's privacy: The receiver should not get any information about the messages that he/she did not capture.

- Scheme 7: Receiver deniable encryption scheme based on mediated RSA

Key Generation - Key generation is same as mediated RSA scheme.

Encryption :

The sender selects a random number R s.t $R \in Z_N$ where

$R = r_0 \dots r_{n-1}$ is the binary representation of R where $n = \ln(N)$

Select random integer i such that $r^i = b$ where b is the message bit.

Generate cipher texts,

$$C_i = i^e \text{ mod}(N)$$

$$C_R = R^e \text{ mod}(N)$$

Sends C_i and C_R to the receiver.

Decryption :

Receiver calculates,

$$T_i = C_i^d \text{ mod}(N)$$

$$T_R = C_R^d \text{ mod}(N)$$

Sends T_R to SEM

Receiver derive $i = T_i S_i \text{ mod}(N)$

SEM calculates,

$$S_i = C_i^s \text{ mod}(N)$$

$$S_R = C_R^s \text{ mod}(N)$$

Sends S_i to the Receiver

SEM derive $R = T_R S_R \text{ mod}(N)$

By oblivious transfer between the receiver and the SEM, the receiver retrieves the relevant value of R related to index i .

Deniability :

If the communication between SEM and the receiver is secure and cannot be intercepted by the attacker, above scheme provides the receiver deniability. Upon coercion, the receiver can reveal his key d and any value as i . The adversary does not have a way to validate fake i value revealed.

2.4.4 Scheme based on simulatable encryption

Invertible sampling [20]:

If a function F is invertible sampling, then there is an efficient inverting algorithm I which provides indistinguishability of below two experiments.

Experiment 1:

$y \leftarrow F(x, r)$, here F 's random coins are r explicit

Return (x, y, r)

Experiment 2:

$y \leftarrow F(x, r)$, here F 's random coins are r explicit

$r' \leftarrow I(x, y)$

Return (x, y, r')

Simulatable public key encryption [20]:

With simulatable public key encryption [4], one can obviously sample a public key without having the secret key and obviously sample the encryption of a random message without having the message. There are three algorithms defined for a simulatable public key encryption. First one is the standard public key encryption that contains key-generation Gen, encryption Enc and Decryption Dec. Second is the oblivious key-generation algorithm OGen which can generate public key opk as the output. The function OGen should support invertible sampling I_{OG} where I_{OG} can generate a secret key from opk . The third algorithm considered is the oblivious encryption algorithm OEnc which takes any public key as the input and generates ciphertext oc . The function OEnc should support invertible sampling I_{OE} where I_{OE} can generate a message from oc and public key used for OEnc.

The security of simulatable encryption has following properties.

1. The encryption scheme should satisfy semantic security that is IND-CPA secure.
2. The public key generated by Gen and the public key generated obliviously by OGen should be computationally indistinguishable.
3. The distribution of the output of Enc and the relevant receiver randomness should be computationally indistinguishable from the distribution of the output of OEnc used and the relevant receiver randomness.

- Scheme 8 : Bi-deniable encryption scheme based on simulatable encryption

- Non fakable encryption

- Key generation : The receiver selects indexes of n size subset R randomly from $5n$ size domain and generates public keys for all $5n$ indexes. However, keys are generated only of the indexes of selected subset R of size n . For the reset, public keys are generated obliviously. All of $5n$ public keys is distributed.

Encryption : The sender selects n size subset S randomly from $5n$ size domain and generates n number of ciphertext of the real message (the length of the message is one bit) for the subset S . The sender also generates $4n$ number of ciphertext obliviously.

Decryption : The receiver selects ciphertext related to the indexes of R , decrypts using the security keys generated at the key generation and takes the majority to derive the decrypted message. According Ivan *et al.* [20], the decryption gives correct message with high probability due to significant overlapping of indexes in S and R .

– Fakable encryption

The Key generation : Instead of generating secret keys only for selected index of subset S , both public keys and secret keys are generated for all index of $5n$ by the receiver.



University of Moratuwa, Sri Lanka.
Electronic Theses & Dissertations
www.lib.mrt.ac.lk

Encryption: By selecting three n size subsets of indexes S_0 , S_1 and Y from $5n$ size domain, the sender generates encryption of zero for the indexes in S_0 , the encryption of one for the indexes in S_1 and the encryption of message m (one bit message) for the indexes in Y . For the rest of $2n$ indexes, the sender generates ciphertext obliviously.

Decryption: Decryption is same as the non-fakable encryption. Because Y makes the majority of decryption as message m , it is possible to achieve the accuracy of decryption with high probability.

– Deniability: The proposed method is based on multi-distributional deniable encryption. Therefore, the adversary's knowledge should be limited to non-faking encryption given above and should not aware on the

fakable encryption/decryption. When adversary coerces the sender to reveal the internals, the senders can simply give S_0/S_1 as S in non-fakable encryption. The reset can be given as obviously generated ciphertext.

When coercing the receiver, the receiver can select n number of indexes that generates the fake message according the majority setting. The receiver can give those selected indexes as R in non-fakable encryption and the relevant secret keys of the selected indexes to the adversary. The public keys relevant to rest of the indexes can be given as obviously generated.

This scheme uses coordinated faking where the faking message should be known by both parties based on prior agreement. The sender can generate fake S from S_0 , S_1 and Y subset indexes without depending on the receiver. But the receiver has to depend on the senders choice. The paper



proposed to do this communication "in-band" using another instance of the same simulatable cryptosystem.

University of Moratuwa, Sri Lanka
Electronic Theses & Dissertations
www.lib.mrt.ac.lk

2.4.5 Scheme based on indistinguishability obfuscation

The formal study of program obfuscation was started by Barak *et al.* [21, 22] in 2001. They focused on virtual black-box obfuscation where the obfuscated program is equivalent to black-box that does not give information of its internals. They have shown that the notion of virtual black-box obfuscation can provide significant result in cryptography including converting shared key encryption into public key encryption. But they have also shown that it is not possible to achieve a general purpose virtual black-box obfuscation.

They also introduced a second notion called indistinguishability obfuscation [22]. In contrast to virtual black-box, indistinguishability obfuscation has possible implementations of general programs. The obfuscations of two distinct

programs with identical functionalities are indistinguishability obfuscated, if two programs are computationally indistinguishable from each other. The first construction of indistinguishability obfuscation for general programs was given by Barak *et al.* [23] and the authors proposed a mechanism to apply indistinguishability obfuscation to achieve functional encryption for general circuits.

Definition: If obfuscation of $F1$ (i.e. $O(F1)$) and obfuscation of $F2$ (i.e. $O(F2)$) are indistinguishability obfuscated, function $F1$ and $F2$ should satisfy below.

$$O(F1) \approx O(F2) \text{ where } F1(x) = F2(x) \text{ for any input } x$$

Punctured programs

Based on indistinguishability obfuscation, new technique called punctured program was introduced by Amit Sahai and Brent Waters [9]. With punctured programs, one can remove the key elements of a program without changing the functionality of the program and generate different functions that are indistinguishability obfuscated. However, the punctured location should not be functionally accessible by the program. Amit Sahai and Brent Waters have proposed applications of punctured programs [9] including deriving public key encryption scheme from shared key encryption scheme and achieving deniable encryption.

- Scheme 9 : Deniable encryption based on punctured programs

The Sender of the proposed deniable encryption scheme has two obfuscated programs: the function **Encrypt** which encrypt the message and the function **Explain** that provides the deniability by giving explanation to the adversary.

The obfuscated program **Encrypt** takes a message m and a random u as the input and produces output cipher c . The encryption uses a standard public key encryption scheme for the general encryption. However, before the encryption, it checks whether u has special arrangement called *hidden sparse triggers*. If u

is an encoding (i.e. $u = \text{Enc}(c, m)$) of a cipher text c and the message m , it gives c as the output cipher. If u is not an encoding of a cipher text c and m , it encrypts m using standard PKE and r . The random u consists of two parts u_1 and u_2 that are equivalent to α and β of the output of Explain program.

The *hidden sparse triggers* should have special set of properties that includes sparseness, possibility of having oblivious-sampling, indistinguishability under malleability attacks and possibility of having publicly-generated-triggers. To achieve the correctness, *hidden sparse set* should be in a sparse subset of a large set where sparseness is achieved. With oblivious sampling, one can derive a sample from full set obliviously. For a third property, it is hard to distinguish a real random value from a hidden sparse trigger value. Having the last property, it is possible to generate a *hidden sparse trigger* by anyone.

If $u_1 = \text{PRF}_2(K_2, (m, c, r))$ and $\text{PRF}_3((K_3, u_1) \oplus u_2) = (m, c, r)$

Output Cipher = c



University of Moratuwa, Sri Lanka.
Electronic Theses & Dissertations
www.lib.mrt.ac.lk

Output Cipher = $\text{Encrypt}_{\text{PKE}}(PK, m; x)$

The program Explain which is also obfuscated, encodes given message m_f using cipher text c and some randomness r . It simply outputs the encoding of m_f and c without considering context of c and provides fake randomness to explain any fake message m_f . The output encoding e of Explain is generated as below. PRF_2 is an injective-puncturable PRF and PRF_3 is a puncturable PRF.

Set $\alpha = \text{PRF}_2(K_2, (m, c, \text{PRG}(r)))$

Set $\beta = \text{PRF}_3(K_3, \alpha) \oplus (m, c, \text{PRG}(r))$

Output encoding $e = (\alpha, \beta)$

The sender encrypts the message using Encrypt and sends the cipher to the receiver. Because possibility of having *hidden sparse triggers* is negligible, the sender is using standard PKE to encrypt the message and the receiver can

correctly decrypt the message using standard PKE with high probability. In case of coercion the sender, the sender can generate encoding e as the fake randomness using `Explain` program. Since the `Encrypt` is obfuscated, the adversary cannot learn the internals. To validate the given randomness, the adversary only can give fake message m_f and e (as the randomness) to `Explain` program as the inputs and compare the output cipher c with stored cipher of real encryption. However, when processing fake message m_f and fake randomness e , the `Encrypt` will detect *hidden sparse triggers* and give c as the output instead of standard PKE encryption. Therefore, the adversary cannot detect the faking.

The construction of deniable encryption by Amit Sahai and Brent Waters is given as publicly deniable encryption [9] where the sender does not require to remember the true randomness to generate the fake randomness. The sender can generate fake randomness using fake message m_f , cipher c and some randomness r . In addition, anyone that have access to the cipher text c can do the same.



University of Moratuwa, Sri Lanka
Electronic Theses & Dissertations
www.lib.mrt.ac.lk

The security of the publicly deniable encryption consists of two separate areas. First is Indistinguishability under Chosen Plaintext Attack (IND-CPA) which is same as the security requirement of standard public key encryption. The second security requirement is the Indistinguishability of explanation, which provides the deniability. This implies that the randomness used to encrypt the real message and the random given by the `Explain` program should be indistinguishable from each other.

The core concept of this deniable encryption is based on achieving the Indistinguishability Obfuscation by puncturing. The puncturing provides computational indistinguishability of what is publicly available for encryption and what is really implemented.

Amit Sahai and Brent Waters have also introduced an extension [9] of above implementation that supports universal deniable encryption. With universal deniable encryption, one can use existing public key infrastructure without re-keying for the deniable encryption. For an example, in above mentioned scenario, there will be same PKC implementation and changes are not required to support deniability. With overwhelming probability (when not selecting *hidden sparse set element* as a random element), the sender is also using standard PKC implementation for encryption and public key encryption system can be defined as a parameter of the obfuscated programs.

2.4.6 Transforming sender deniable encryption to receiver deniable encryption or vice versa

According to the given scheme [1, 6], a sender-deniable-encryption can be transformed to receiver-deniable-encryption with additional number of transactions. With this scheme, the receiver initiates the transaction. The receiver selects random number r and sends it to the sender using the sender-deniable-encryption scheme. The sender retrieves the r value and derives $(m \oplus r)$, where m is the message. The generated cipher $m \oplus r$ is sent to the receiver. Upon coercion, the receiver is able to preserve the confidentiality using the faking capabilities provided by original sender deniable encryption[1] which was used to communicate the random r .

A sender deniable scheme can be derived from a receiver deniable scheme via the inverse of above methodology.

2.4.7 Transforming a sender/receiver deniable encryption to bi-deniable encryption

By having number of intermediaries, a sender/receiver deniable encryption scheme can be transformed into a bi-deniable encryption scheme [1]. To send bit b , n number of bits are generated such that $\oplus b_i = b$ and each b_i is sent to an intermediary (n number of intermediaries are involved) using a sender deniable encryption scheme. Each intermediary sends b_i to the receiver using a receiver deniable encryption scheme. The receiver deniable encryption can be derived by the sender deniable

encryption and vice versa. The receiver can derive the b using $\oplus b_i = b$.

If least one of the intermediaries is not coercible or corruptible, the scheme still resilient against the coercion. When the coercer is able to do the simultaneous coercion, additional coordination is required between coercible parties to preserve the privacy of the communication.

2.5 Shared Key Deniable Encryption Schemes

- Scheme 10 : Scheme based on one-time-pad

In this scheme [1], the message m is encrypted with key k and is produced the cipher text $c = m \oplus k$. To give the fake message m_1 , k_1 is generated such that $k_1 = m_1 \oplus c$. However, the keys will be equal to the length of the message. Thereby, the scheme may not be used in practical applications.

- Scheme 11 : Scheme based on pseudo random generator [1]

Encryption:  University of Moratuwa, Sri Lanka.

In addition to real message (m) and the real key (k_1), the sender selects $(t-1)$ number of fake messages ($m_2 \dots m_t$) and keys ($k_2 \dots k_t$). Each message is segmented into blocks with n bits.

$$m_1 = m_1^1, m_1^2, \dots$$

$$m_2 = m_2^1, m_2^2, \dots$$

...

$$m_t = m_t^1, m_t^2, \dots$$

Using pseudorandom generator, a and b pairs are generated as in figure 2.2. The sender finds the Q_j polynomial equations s.t Q_j satisfies condition $m_i^j = Q_j(a_i^j) - b_i^j$ and derives C_j (coefficient of Q_j). The figure 2.3 elaborates the Q_j and C_j generation.

Decryption:

The receiver can generate same pseudorandom output using given k value and generates the message as $m_i^j = Q_j(a_i^j) - b_i^j$. To retrieve the real message m_1, k_1

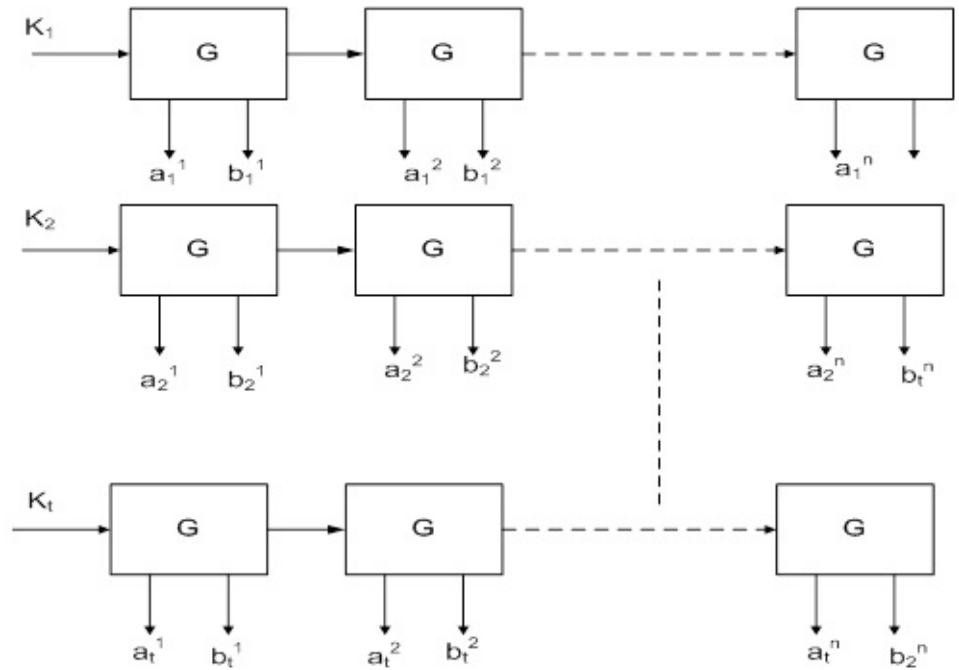


Figure 2.2: Pseudorandom number generation


 University of Moratuwa, Sri Lanka.
 Electronic Theses & Dissertations
 www.lib.mrt.ac.lk
 can be used and the rest of the keys decrypt the cipher text into different fake messages.

Deniability:

Upon coercion, the sender/receiver can select any of the key values and relevant fake messages. However, the scheme provides plan-ahead deniability where the cipher text can be only opened as $(t - 1)$ number of fake messages that are decided at initial stage.

Table 2.1: Summary of the existing deniable encryption schemes

Scheme	PKE/ Shared Key	Type of Deniability	Implementation
Scheme 1	PKE	Sender/Full Deniable	Using Sparse Set

Scheme 2	PKE	Sender/Full Deniable	Using Sparse Set
Scheme 3	PKE	Sender/Multi-Distributional Deniable	Using Sparse Set
Scheme 4	PKE	Sender/Full Deniable/Interactive Construction	Based on Samplable Encryption
Scheme 5	PKE	Sender/Full Deniable/Interactive Construction	Based on Samplable Encryption
Scheme 6	PKE	Sender/Full Deniable/Interactive Construction	Based on Samplable Encryption
Scheme 7	PKE	Receiver/Multi-Distributional Deniable	Based on MediatedRSA and Oblivious Transfer
Scheme 8	PKE	Bi/Multi-Distributional Deniable	Based on Simulatable Encryption
Scheme 9	PKE	Sender/Full Deniable	Based on Simulatable Encryption
Scheme 10	Shared Key	Full Deniable	Using One time padding
Scheme 11	Shared Key	Full Deniable	Using pseudo random generators

2.6 Cryptanalysis

2.6.1 Ciphertext indistinguishability

Indistinguishability is a main security notion of encryption which can be further described based on below types. In each of below definitions, the challenger generates keys and sends the public key to the adversary. The adversary can experiment with any number of encryptions before receiving challenge message m_b where $b \in \{0,1\}$.

- IND-CPA (Indistinguishability under chosen-plaintext attack): If an adversary is given message m_0, m_1 and the cipher text $C = E(m_b)$, the probability of finding correct m_b from m_0 and m_1 should be $\frac{1}{2} + \epsilon$ where $b \in \{0,1\}$ and ϵ is negligible.

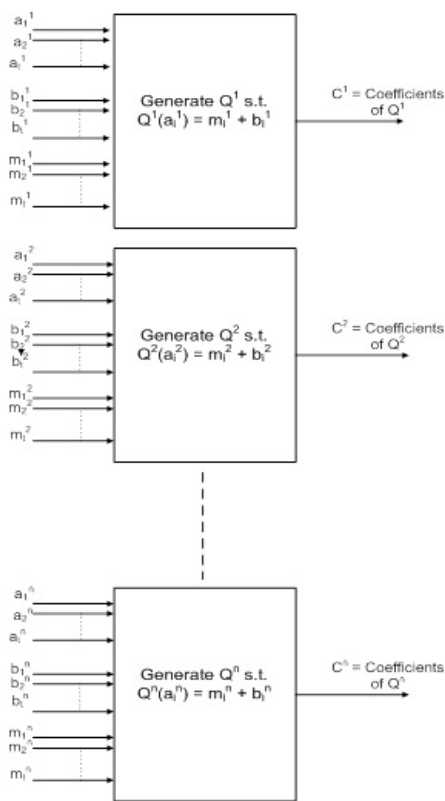


Figure 2.3: Encryption of shared key deniable encryption using pseudorandom generation

- IND-CCA1 (Indistinguishability under chosen-cipher text attack): If an adversary is given the cipher text $C = E(m_b)$ of plaintext messages m_0 or m_1 , the probability of finding correct m_b from m_0 and m_1 should be $1/2 + \epsilon$ where ϵ is negligible. Same as IND-CPA, the adversary can do any number of polynomial bound encryptions. In addition to that, the adversary can request any number of decryption (from decryption oracle) of arbitrary cipher texts before deriving the message m_b .
- IND-CCA2 (Indistinguishability under adaptive chosen-cipher text attack): If an adversary is given the cipher text $C = E(m_b)$ of plaintext messages m_0 or m_1 , the probability of finding correct m_b from m_0 and m_1 should be $1/2 + \epsilon$ where ϵ is negligible. Same as IND-CCA1 the adversary can use any number of polynomial bound encryptions and can request any number of decryption (from decryption oracle) for arbitrary cipher texts before receiving cipher text C . In contrast to IND-CCA1, the adversary can access decryption oracle even after the receiving the challenge cipher text C . However, the adversary is not allowed to pass cipher text C to decryption oracle.

- Indistinguishability from random noise: This is the Indistinguishability of cipher-text against the true random value generated with same length. This is not a critical requirement of encryption in semantic security. However, this is critical factor for steganographic schemes and the notion of plausible deniability where adversary should not be able to detect the existence of message in the given cipher text.

2.6.2 Deterministic encryption vs probabilistic encryption

Deterministic encryptions always generate same cipher text for a given plaintext in any repetition with same key and cipher text does not depend on any random value. With Deterministic encryption, if an adversary able to recover any message with existing cipher, he/she will have additional advantage for detecting the plaintext of future encryptions. For example, the adversary can encrypt known possible messages using the public key and can do an statistical check on cipher text to find occurrences of know messages.



University of Moratuwa, Sri Lanka.
 Electronic Theses & Dissertations
www.lib.mrt.ac.lk

In contrast to deterministic encryption, probabilistic encryption uses additional randomness r to generated $e \leftarrow (m, k, r)$. Therefore, the cipher text for given message and a public key is mapped to number of possibilities based on the randomness r . Therefore, it is infeasible to launch a statistical attack using past history of cipher to message mapping.

2.6.3 Malleability of encryption

With malleable encryption, one can generate a ciphertext c_2 by transforming a ciphertext c_1 which is the encryption of know message m_2 (knowledge of m_2 can be partial). If the encryption is malleable, the adversary can generate ciphers that will decrypt to plaintext with known characteristics.

2.6.4 Methods of cryptanalysis

- Statistical cryptanalysis

- Frequency analysis : The elements such as letters of the plain text alphabet do not have same probabilistic distribution of occurrence in specific usage. The variation of the probability depends on the language characteristics. For example, the letter E in English language has high probability of occurrence compare to letter Z. Same as letters, combinations of letter also have different probability distribution. In substitution ciphers, this probabilistic distribution can be used to identify the occurrences of particular element of the alphabet. Since the frequency of each element in ciphertext reflects the plain text element that was substituted, one can break the cipher by observing the frequency distribution of elements such as letters and combinations.
- Linear cryptanalysis : Linear cryptanalysis is one of the commonly used cryptanalysis method that is based on finding linear approximations to relate the ciphertext, the plaintext and the key. In the first step of linear cryptanalysis, one has to deduct linear equation to relate the elements of key to plain-texts and the relevant cipher-texts. To derive the linear equation, he/she can use the cipher text of known plain texts and known keys. Then the equation can be used to find the real key and break the encryption of specific communication.
- Differential cryptanalysis : Differential cryptanalysis is a statistical analysis method that is based on the possibility of deriving high probability function that relates input plaintext differences to the relevant output differences of the cryptosystem. This was first introduced by Eli Biham and Adi Shamir in 1990. Though differential cryptanalysis is primarily applicable for block ciphers using chosen plaintext attack, it can be used to break stream cipher as well. In basic differential cryptanalysis, the attacker selects pair of plaintext with constant differences and finds the relevant differences in the outputs. The attacker analyses the result distribution to find any statistical patterns and uses them to recover the key. With the basic key recovery, large number of plaintext pairs may be



needed. But by evaluating the stages of encryption algorithm, the plaintext pairs can be selected for successful attack with lesser number of plaintext pairs.

- Rubber hose cryptanalysis : Rubber hose cryptanalysis is based on coercion/torturing to reveal the secret internals of the cryptosystem instead of using cryptographic methods. The retrieved data can be used for further cryptanalysis. Though the name indicates physical torturing, any coercion using physiological/legal factors is also considered as rubber hose cryptanalysis.



University of Moratuwa, Sri Lanka.
Electronic Theses & Dissertations
www.lib.mrt.ac.lk

3 IMPLEMENTATION OF FULL SENDER DENIABLE ENCRYPTION

3.1 Implementaion

In this research, a full-sender-deniable encryption has been implemented using the parity based scheme introduced by Canetti *et al.* [1]. The implementation consists of four main modules : Keygen, Sender, Receiver and Adversary.

3.1.1 Keygen

Keygen is to generate PKC keys (for an example, RSA) for sparse set generation. For RSA as PKC, it provides public key (e with N) and private key (d with N) as the outputs. The user has to provide the bit length as the security parameter input.

3.1.2 Sender

Sender module has four distinct functions. Sparse set generation, deniable encryption, providing non-fake randomness used for encryption and providing fake randomness that produce the given fake message. Sparse set can be generated based on three algorithms. Two of them are based on the schemes proposed by Canetti *et al.* [1] and one is a new construction proposed in this research.

1. Sparse set generation.

In each implementation, $H1$ is an array of random numbers which is supposed to be given to the adversary at the coercion.

- a) Implementation 1: This is based on the schemes proposed by Canetti *et al.* [1] and the sparse set values are generated as below. The function Parity (x) returns the parity of x . Sparse set is generated using RSA as the trapdoor.

Function SparseSetGen

Do $i = 1$ to k

Select $x = \text{Random}(\text{bit length} = s)$

```

Do while Parity(x) <> 1
    Select x = Random(bit length = s)
End do
S1 = EncryptRSA(x)
Store H1[i] = x
S = Concatenate(S , S1)
End do

```

- b) Implementation 2: This is also based on the schemes proposed by Canetti *et al.* [1]. The sparse set is generated using RSA as the trapdoor. The function Parity(x) simply returns the parity of x and S values are generated as below.

```

Function SparseSetGen
Select x = Random(bit length = s)
Store H1 = x
e = x
Do i = 1 to k
    e = EncryptRSA(e)
    S1 = Parity(e)
S = Concatenate(S, S1)
End do

```



University of Moratuwa, Sri Lanka.
 Electronic Theses & Dissertations
 www.lib.mrt.ac.lk

- c) Implementation 3: This is a new construction proposed in this research. The construction does not directly satisfy the first point of sparse set definition as we are not using k. However, the first point of the definition is to satisfy the accuracy at decryption and deniability for the sender. Below construction satisfies both requirements and the detailed analysis is given in section 3.2. The sparse set is generated using RSA as the trapdoor. K is a constant value.

```

Function SparseSetGen
Select x = Random(bit length = s)

```

```

Store  $H1[next] = x$ 
 $x_2 = (x + K) \bmod(2^s)$ 
 $S1 = \text{EncryptRSA}(x)$ 
 $S2 = \text{EncryptRSA}(x_2)$ 
 $S = \text{Concatenate}(S1, S2)$ 

```

2. Encryption:

$H2$ is an array of random numbers which is supposed to be given to the adversary at the coercion.

Function DeniableEncryption

$B[] = \text{BitRepresentation}(m)$

For $i = 1$ to $B.length$

Select $x = \text{Random}(\text{bit length} = p)$

Do while $\text{Parity}(x[]) = B[i]$

Select $x[] = \text{Random}(\text{bit length} = p)$

End do

Store $H2[i] = x[]$



University of Moratuwa, Sri Lanka.
 Electronic Theses & Dissertations
www.lib.mrt.ac.lk

For $j = 1$ to p

If $x[j] = 1$

Get $e = \text{SparseSetGen}()$

Else

If implementation = 1: $e = \text{Random}(\text{length} = s*k)$

If implementation = 2: $e = \text{Random}(\text{length} = s + k)$

If implementation = 3: $e = \text{Random}(\text{length} = 2s)$

$c = \text{concatenate}(c, e)$

End For

End For

3. Providing non-fake randomness:

Function GiveNonFakeRnd

Give $H1$ and $H2$

4. Provide fake randomness:

Function GiveFakeRnd

Get *fakeMessage* and *realMessage*

$B[] = \text{BitRepresentation}(\text{realMessage})$

$B2[] = \text{BitRepresentation}(\text{fakeMessage})$

For $i = 0$ to $B.\text{length}$

$F1 = H1$

If $B[i] = B2[i]$

$F2[i] = H2[i]$

Else

Select $r = \text{Random}(1 \text{ to } p)$

Do while $H2[i].\text{BitAt}(r) \neq 1$

Select $r = \text{Random}(1 \text{ to } p)$

End do

$F2[i] = H2[i]$

$F2[i].\text{BitAt}(r) = 0$

$F1[H2.\text{numberOfOnesUpto}(\text{position} = i * p + r)] = \text{null}$

End if

End For

$F1.\text{removeNullElements}$

Give $F1$ and $F2$



University of Moratuwa, Sri Lanka.
Electronic Theses & Dissertations
www.lib.mrt.ac.lk

3.1.3 Receiver

Receiver has one main function that can be divided into two main sub-functions based on the level of decryption. They are (1) decryption of deniable encryption and (2) sparse set elements detection using a common PKC (here we are using RSA).

1. Detecting sparse set elements

a) Implementation 1

Function DetectingSparseSetElement

$C[] = \text{segment}(c, \text{length} = s)$

```

For  $i = 0$  to  $k$ 
     $D = \text{DecryptRSA}(C[i])$ 
    If  $(\text{Parity}(D) = 0)$ 
        Break and Return False
    End If
End For
Return True

```

b) Implementation 2

Function DetectingSparseSetElement

```

 $X = c.\text{firstBits}(\text{length} = s)$ 
 $e[] = c.\text{lastBits}(\text{length} = k)$ 
For  $i = k$  to  $0$ 
     $X = \text{DecryptRSA}(X)$ 
    If  $(\text{Parity}(X) \neq e[k])$ 

```

Break and Return False

End If

End For

Return True



University of Moratuwa, Sri Lanka.
 Electronic Theses & Dissertations
www.lib.mrt.ac.lk

c) Implementation 3

Function DetectingSparseSetElement

```

 $X1 = c.\text{firstBits}(\text{length} = s)$ 
 $S1 = \text{DecryptRSA}(X1)$ 
 $X2 = c.\text{lastBits}(\text{length} = S)$ 
 $S2 = \text{DecryptRSA}(X2)$ 
If  $(X2 - X1 = K)$ 

```

Return True

Else

Return False

2. Decryption of deniable encryption

Function DecryptionDeniable

```

S[] = C.segment(length = s)
For i = 0 To S[].length
    If DetectingSparseSetElement(S[i]) = True
        B[i] = 1
    Else
        B[i] = 0
    End For
For i = 0 To B[].length
    M[i] = Parity(B[i*p] to B[i*(p + 1) - 1])
End For
m = M[].ConvertToString()
return m

```

3.1.4 Adversary

The implementation of an adversary consists of below functions

1. Validation - Here the adversary validates the given fake/non-fake random values against the cipher text captured. For faking, the sender passes $F1[]$ and $F2[]$ as $T1[]$ and $T2[]$. For non faking, the sender passes $H1[]$ and $H2[]$ as $T1[]$ and $T2[]$.

a) Implementation 1

```

Function GetSparseSet(j)
    Do i = 0 to k
        Select x = T1[j*k + i]
        S1 = EncryptRSA(x)
        S = Concatenate(S , S1)
    End do
    Return S

```

b) Implementation 2

```

Function GetSparseSet(j)
    Select x = T1[j]

```

```

e = x
Do i = 1 to k
    e = EncryptRSA(e)
    S1 = Parity(e)
    S = Concatenate(S , S1)
End do
S = Concatenate(e, S)
Return S

```

c) Implementation 3

Function GetSparseSet(*i*)

```

Select x = F1[i]
x2 = (x + K) mod( 2s)
S1 = EncryptRSA(x)
S2 = EncryptRSA(x2)
S = Concatenate(S1, S2)

```

Return S



d) Validation by the adversary

Function Validate

```

C1[] = C.segment(length = s*k)
For i = 0 to T2.length
    If T2[i] = 1
        S[i] = GetSparseSet(i)
        If S[i] <> C1[i]
            Return "Faking Detected"
        End if
    Else
        R[i] = C1[i]
    End if
End for
Return "No Faking Detected"

```


2. Analysis 1 - With this analysis, adversary simply checks whether the given $F2[]$ has all ones (i.e. $\{1\}^l$) random blocks. If any value detected, the adversary can fully trust on the sender on those values. Based on those value, one can reduce the possible message space of the real message.
3. Analysis 2 - Assuming that the standard public key encryption used for the trapdoor permutation is plaintext aware, the adversary uses given S values and compares it with given R values to find any possible faking. For this purpose, he compares $S[]$ and $R[]$ arrays generated at validation to find common occurrences. However, the detection of duplication does not directly imply the faking as S is a subset of R . To detect the faking, the frequency of duplication should be compared against the accuracy of each implementation (i.e. $1/2^k$, $1/2^k$ and $1/2^s$ for each implementation respectively).
4. Analysis 3 - This analysis provides the number of standard public key encryptions required to generate the cipher text from a given message. This can be used as a tool for a side channel attack. Since only S elements are given as R elements, the estimated number of required encryptions will be always less than the number of standard public key encryptions in real encryption.
5. Analysis 4 - The adversary derives the inverse of the bit of the given plain text(fake/non-fake) by the sender. However, if the derived inverse is directly converted to plaintext, it will be outside the considered alphabet (for an example, here we consider English letters/numbers). Therefore, before converting to plain text, the inverse should be rearranged to support the alphabet in use.
6. Attack using Analysis 4 - The adversary uses the message generated in analysis 4 and asks the sender to generate fake randomness $F2[]$. Then the adversary can calculate the bitwise xor of $T2[]$ of the sender given as true message and $F2[]$ for the generated fake message. The result will be the real value of the communication.

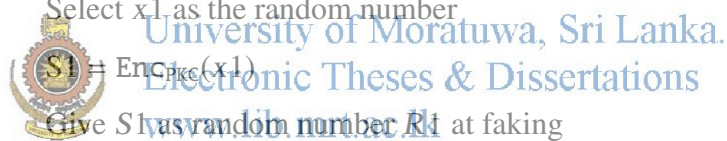
3.2 Sparse Set Generation using Probabilistic Encryption

The sparse set generation explained by Canetti *et al.* [1] is based on deterministic encryption. Therefore, the detection (by adversary) of S elements generated by repeating the underlying public key encryption of the same random number is a main consideration. This is explained in below S element generation with deterministic encryption.

Occurrence 1

Select x_1 as the random number
Generate S_1 using $\text{Enc}_{\text{PKC}}(x_1)$
Give S_1 as *Sparse Set Element*
Adversary store x_1

Occurrence 2

Select x_1 as the random number
 $S_1 = \text{Enc}_{\text{PKC}}(x_1)$
Give S_1 as random number R_1 at faking

In occurrence two, the adversary with previous knowledge can detect faking. Therefore, when using deterministic encryption as the trapdoor, the value k and t should satisfy two conditions.

- For the accuracy, $|S| < 2^{t-k}$ where k should be significantly large (but $k < t$).
With implementation 3, probability of selecting S element as R element can be given as $1/2^s$. Therefore, even with small k , by increasing the length s , higher accuracy can be achieved.
- To prevent above statistical attack, the bit length of the random number t/k also should be large. However, with probabilistic encryption, the adversary can not detect x_1 from S_1 . Therefore, above statistical analysis is not possible. The S element generation can be as simple as $S = \text{Enc}_{\text{PKC}}(X)$ where X is a constant

public value.

Since implementation 3 satisfies the above two condition, it provide the same security as implementation 1 and implementation 2.

3.3 Performance Comparison of the Implementations

In addition to the security of a crypto-system, the performance parameters are important in practical applications. The performance of a crypto-system at encryption and decryption can be evaluated with three main parameters: correctness, throughput and execution speed. The correctness of a crypto-system yields the accuracy of retrieving same message at the decryption by the receiver. The throughput is given as ratio of the cipher-text to plain-text and this effects the bandwidth/storage requirement of the particular implementation. The execution speed is given as the time taken for encryption/decryption in practice. Because all three implementations use the same algorithm except for the way of S element generation/detection, the execution speed can be directly mapped into the number of standard PKC encryptions/decryptions required.



University of Moratuwa, Sri Lanka
Electronic Theses & Dissertations
www.lib.mrt.ac.lk

3.3.1 Encryption

The performance of the throughput at the encryption is evaluated based on the cipher-text to plain text ratio. The execution speed of the encryption is evaluated based on the number of PKC encryptions. Since the length of the random V is the main security parameter of the implementation, it is considered as the x-axis for all three comparisons. The bit length is kept as constant ($k = 16$). The performance comparison is given in Fig 3.1 and Fig 3.2.

3.3.2 Decryption

The performance of the decryption is given based on the number of PKC decryptions required. The performance comparison is given in Fig 3.3.

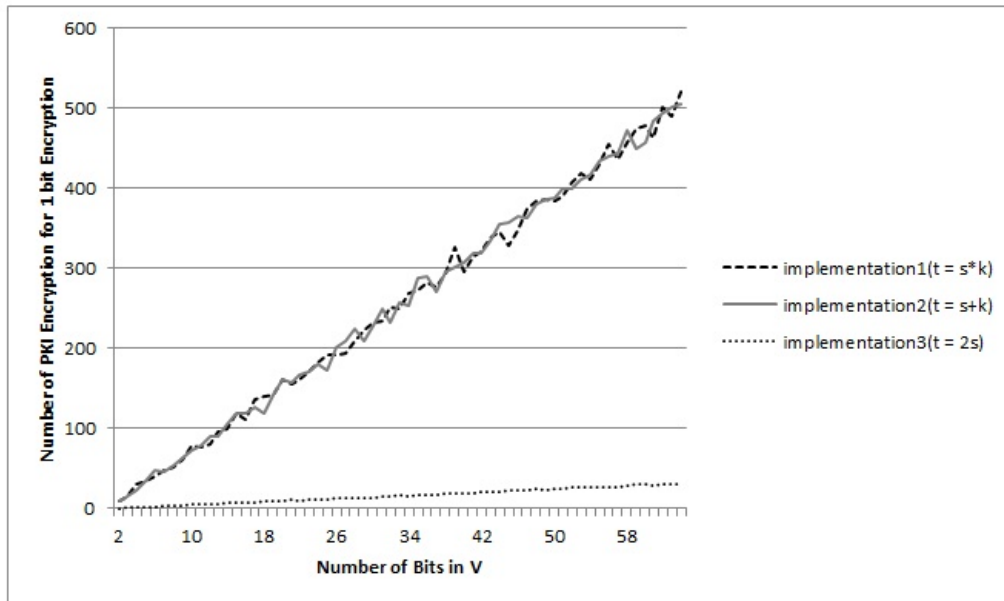


Figure 3.1: Number of PKC encryption for 1 bit of deniable encryption vs bit length of random V



University of Moratuwa, Sri Lanka.
 Electronic Theses & Dissertations
www.lib.mrt.ac.lk

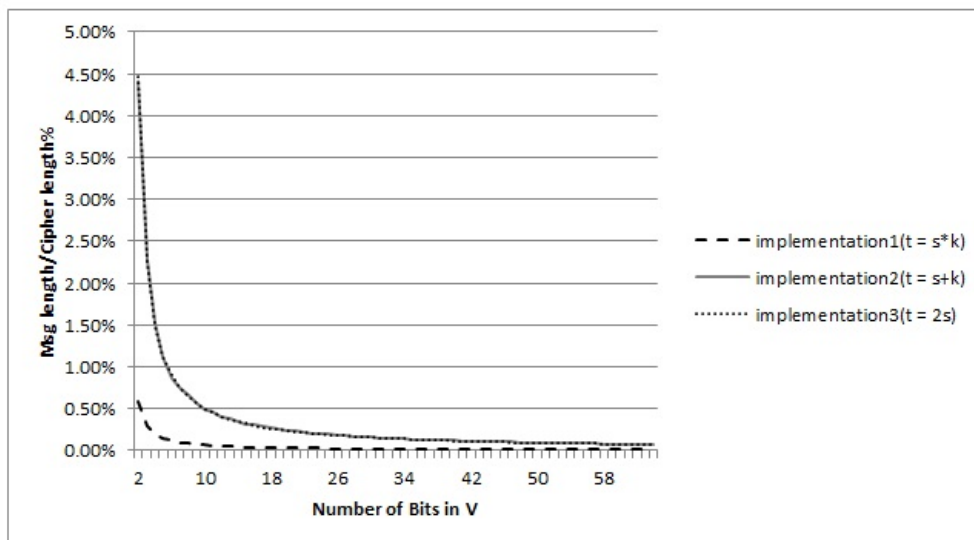


Figure 3.2: Message length /Cipher-text ratio vs bit length of V

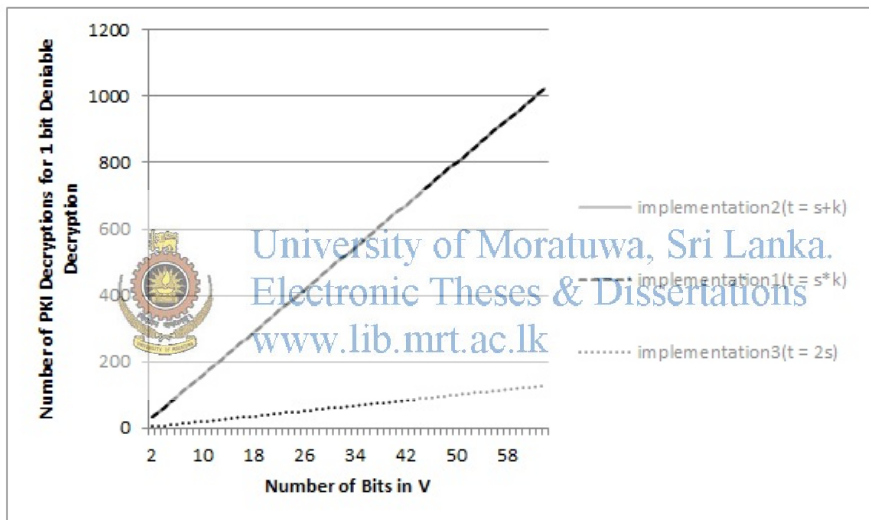


Figure 3.3: Number of PKC decryption for 1 bit of deniable encryption vs bit length of random V

4 CRYPTANALYSIS

With the objective of crypt-analyzing full-sender deniable encryption, this research has defined a common model based on two full-sender deniable encryption schemes: the parity based scheme [1] and the scheme based on samplable encryption [2].

4.1 Common Model for Full-Sender Deniable Encryption

1. Three main entities involve with the communication: the sender encrypts the message, the receiver decrypts the message and the adversary who is able to coerce the sender to reveal internals. The standard PKC key generation is considered as an integral part to the receiver.
2. Threat Model:
 - a) The adversary does not have access to the internal execution of the encryption where partial calculation is not possible to break the encryption. Therefore, by monitoring the encryption, the adversary cannot gain an additional advantage to validate the information revealed by the sender.
 - b) The adversary does not have access to the internal execution of the decryption where partial calculation is not possible to break the decryption. Therefore, by monitoring the decryption, the adversary cannot gain an additional advantage to validate the information revealed by the receiver.
 - c) The communication channel between the sender and the receiver is not secure. Hence, the adversary has access to the cipher text which was transmitted.
 - d) The coercible party does not erase the plaintext messages or the coercer does not have any confirmation on the such a deletion.

3. Encryption: The Encryption can be explained in four steps. The third and fourth steps provide the deniability while the steps one and two are to provide encoding/encryption requirements using one-to-one/one-to-many substitutions without collisions.

- a) Step 1: All the given encryptions are bitwise encryptions. Thereby, the first step of the encryption is to convert the plain text message into bits.
- b) Step 2: Substituting each bit with a random value based on a given criteria. With the parity based scheme, this is the step equivalent to selecting a random V while with the scheme based on Samplable encryption, this step is equivalent to selecting random indexes for A , B and C .
- c) Step 3: Substituting each bit with a cipher value generated using a standard public key encryption. In parity scheme, each 1 is replaced with S element generated using PKC as the trapdoor and 0 is replaced with a random element. In the scheme based on Samplable encryption, each bit is encrypted by selected public key value.
- d) Step 4: This step consists of the communication with the receiver. The parity scheme uses single one way communication. The scheme based on samplable encryption uses interactive method that uses multiple communications between the sender and the receiver.

4. Two main views are considered as given in Figure 4.1.

- a) The sender's view of encryption: Use real message m , randomness R_s , public key K_{pk} and proceed above 4 steps.
- b) The adversary's view of encryption (Validation): Use message m' , randomness R'_s , public key K_{pk} and proceed above 4 steps.

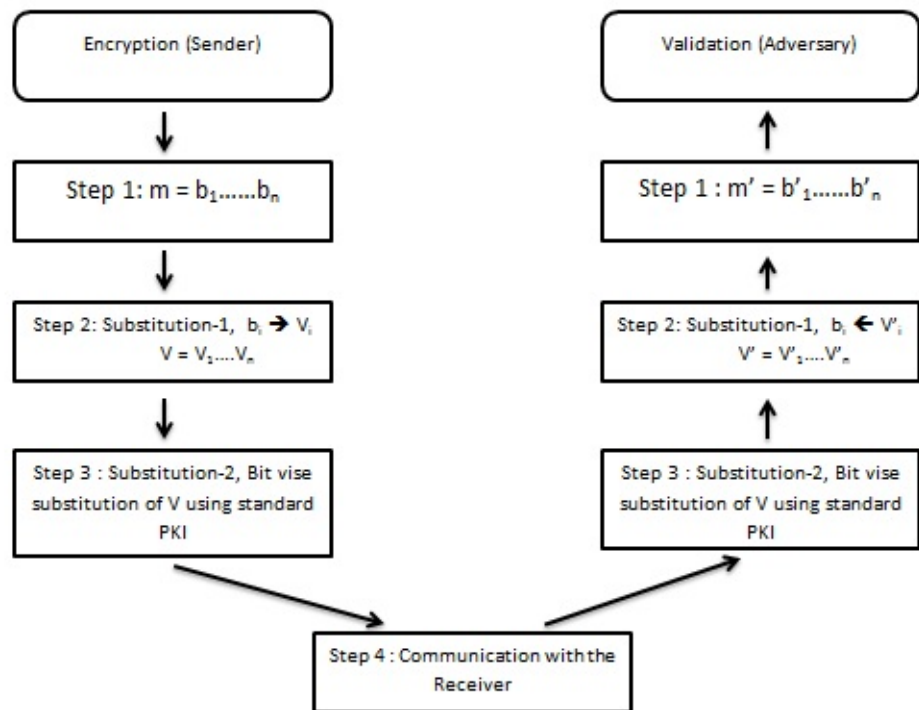


Figure 4.1: Sender's view vs adversary's view of the encryption

5. Possible analysis/attacking points

University of Moratuwa, Sri Lanka.
Electronic Theses & Dissertations
www.lib.mrt.ac.lk

- a) Analysis at Step 1 and 2: Faking of full sender deniable public key encryption can be elaborated as a shared key encryption with one time pad (i.e. the secret key) as given in Figure 4.2. Therefore, one can try with the shared key based analysis at this steps.
- b) Analysis at Step 3: As given above the standard PKC is used in this step. Therefore, any attack based on vulnerabilities/limitations of standard PKC can be considered. Also the notion of collusion and correctness should be considered at this step.
- c) Analysis at Step 4: Analysis based on interactions between the sender and the receiver is considered at Step 4. The possible scenarios including collecting public parameters, the coercion of the sender and any access to the receiver (without coercion) can be considered.

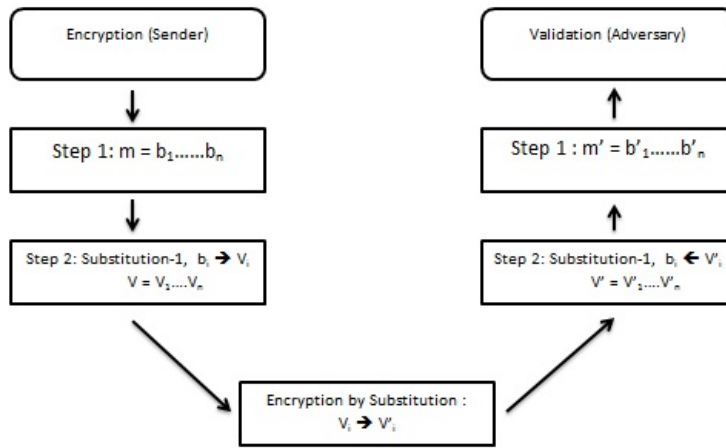


Figure 4.2: Sender faking as shared key encryption

For the detail cryptanalysis, this research has selected the parity based scheme [1] proposed by Canetti *et al.* In first part of crypt-analysis, the analysis based on the limitation given by Canetti *et al.* [1] has been reevaluated. In second part, this research has given possible attacks using statistical methods and finally, possible attacks proposed based on the faking algorithm.

4.2 Cryptanalysis Based on $1/n$ -deniability

The main limitation of the parity scheme is that any bit can be faked only with $1/2^n$ probability where n is the bit length of a random value V . When encrypting 0, if the sender selects (randomly) $V = 0$ as the random, it is not possible to generate a fake randomness (by flipping one 1 coin as 0 coin) as S element are not included in the original random V . In this case, one can suggest not to use $V = 0$ as the parity value at the real encryption. However, because this is full deniable encryption, *not using* $V = 0$ is also know factor for the adversary. Therefore, when giving $V = 0$ at faking, the adversary can detect the faking. Moreover, the probability of un-deniability will increase to $n/2^n$, because the probability of having $V = 1$ in encryption is $n/2^n$.

On the other hand, if the adversary is given all one as the random value V , there couldn't be any faking possibility and the adversary can fully trust the sender on those values. If the message length is m , then expected number of guaranteed true bits is

$(m/2^n)$. This means, if small n is used to encrypt significantly long message, the adversary can find significant number of guaranteed bits as true value. The adversary can detect those all one occurrences and gains the advantage of having narrowed message space compared to the original possible message space of 2^m . The expected sized of the possible message space will be $2^{(m(1-1/2^n))}$

4.3 Cryptanalysis Based on Statistics

As given above, the parity based encryption can be considered as an encryption with two substitutions. If the true message is $m = B_1...B_n$ where B is $\{0, 1\}$, the two substitutions can be given as below,

1. Each B_i is substituted with a random number V_i where $B_i = \text{Parity}(V_i)$
2. Each one in V_i is substituted with S element and each zero in P_i is substituted with R element

The adversary's view of encryption also can be explained as two substitutions. If the fake message given is $m' = B'_1...B'_n$, the two substitutions can be given as below,

1. Each B'_i is substituted with random number P'_i where $B'_i = \text{Parity}(V'_i)$
2. Each one in V'_i is substituted with S element and each zero in V'_i is substituted with R element

Therefore, we can consider faking as a symmetric encryption that encrypt true random values($V_1...V_n$) to fake random values($V'_1...V'_n$) using single step of substitution. The key of the encryption can be considered as one time pad and it is generated based on the sender's choice (i.e. faking or not faking of each bit).

With this new idea, one may suppose to continue the cryptanalysis of the deniable encryption based on known methods used to break substitution cipher. However, the alphabet of the plaintext/ciphertext of encryption considered does not have characteristics of natural language. The characteristics of the alphabet mainly depend on the random number generator.

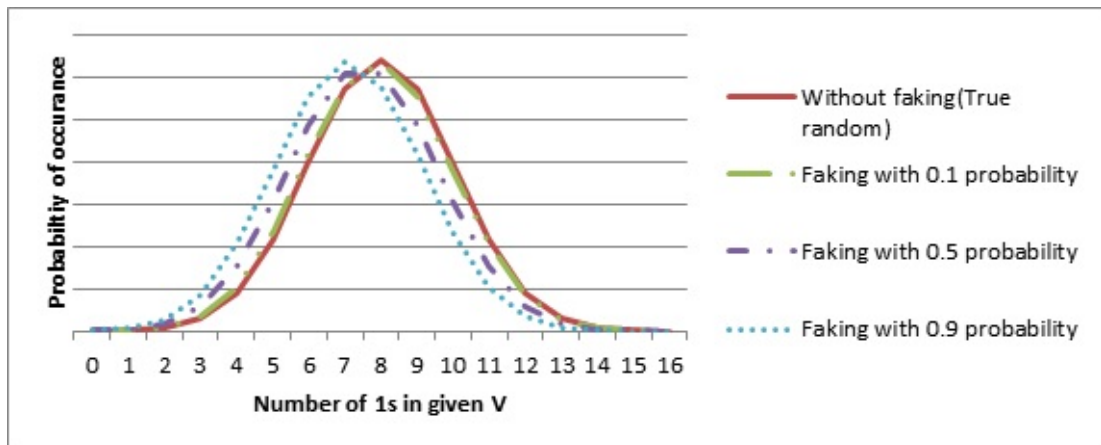


Figure 4.3: Faking against random number generator

First, we assume of having a perfect random number generator. In faking, the sender is giving one S element as R in V . Therefore, the probability distribution of fake V is expected to deviate from the probability distribution of true V . However, each random V has uniform probability distribution and for fake detection, the sample size has to be significant. With given parity based deniable encryption, probability of having i number of 1s in V is ${}^n C_i / 2^n$ where maximum probability is at $i = n/2$. If the faking happens, the probability distribution will be deviated from probability distribution of true randomness same as above. But, this is detectable with a smaller sample size compared to detecting deviation in the distribution V .

If the faking is done with P_f , then the expected probability distribution of 1s in V can be given as $[(1 - P_f) * {}^n C_i + P_f * {}^n C_{i+1}] / 2^n$. According to the equation, faking distributions should be moved to the left from the real random distribution graph. The detection is significant when the n is small. The figure 4.3 shows the distribution with $n = 16$.

Moreover, if a given deniable encryption uses a non-perfect random number

generator, the adversary may have additional advantage based on the known characteristics of the generator.

The probability of S element given as R element is $1/2^k$ which is considered to be negligible value. But when the sender is giving multiple fake messages, the adversary can collect S values and checks them against the given R value. Because at faking, it is always gives S values as R values and not in other way, the adversary may be able to detect probability of S elements given as R element is more than $1/2^k$ and detect the sender's faking.

However, the above statistical attack is only possible, if the underlying public key encryption(for an example, RSA) is deterministic where the sender generates same cipher for a plain text message every time. But if the underlying public key encryption is probabilistic (CPA secure), it is infeasible for the adversary to collect the data on cipher text space by comparison.

4.4 Cryptanalysis Based on Faking Algorithm



University of Moratuwa, Sri Lanka.
Electronic Theses & Dissertations
www.lib.mrt.ac.lk

According to the faking algorithm, the sender gives S elements as R elements only, but not R element as S elements. Therefore, the adversary can trust on all the values given as S elements and need to work only on random R elements.

4.4.1 Coercing faking algorithm

If the sender is generating number of fake message on same ciphertext, the adversary can collect all those random S values and can generate the true message by considering all S values in different fake messages cumulatively. But one can argue if the sender is generating fake messages for same cipher, the adversary can already detect the faking. Therefore, the sender should have to generate only one fake message in particular instants of coercion.

But above argument is not true with full deniable encryption. The internal

implementation of the sender's encryption is considered to be public parameter and the adversary already knows the possibility of faking by the sender. Thereby, the adversary can ask/coerce the sender to generate fake messages for the encryption (fake or non-fake) and collect fake messages. In this case, the sender has to generate fake messages without resistance.

4.4.2 Detecting faking

The adversary can ask to generate fake randomness for any message other than the given for the adversary at initial faking. The sender's faking is based on the original message and the sender provides randomness as $V'_f = \{S_2, R\}^n$ where S_2 is a subset of S . The randomness given by the sender at initial faking is $V' = \{S_1, R\}^n$ where S_1 is a subset of S . If the true randomness is given, S_2 should be a subset of S_1 ($S_2 \subset S_1$). Any s element in S_2 that is not in S_1 implies the faking and the faking can be detectable.

This can be further explaining by considering the above idea of *faking as a shared key encryption*. In non-faking, the plain text (the real random V used for the encryption) and the cipher text (the V' given to the adversary) should be equal where all the bits of one-time pad k should be 0.

$$V' = V \oplus k$$

$$V'_f = V \oplus k_f$$

$$V' \oplus V'_f = k \oplus k_f$$

$$k = V' \oplus V'_f \oplus k_f$$

The adversary can derive k using known values of V_f , V'_f and k_f . if any k does not equal to all zero indicates the faking. The time complexity of detection is $O(n)$ for all three implementations where n is the bit length of the plaintext message.

4.4.3 Deriving true randomness

Assume the sender gives randomness V' for the adversary. The adversary calculates the inverse of given bit values $b...b_n$ by flipping each bit and generates the message m_a . The message m_a is given to the sender and asks to generate fake randomness V_f that explains the same cipher text and m_a . The adversary can derive the true message

by finding the union of the S values given in V' and the S values in V_f . The explanation of the derivation is given below.

Consider b , b_g and b_{rg} are original message bit of the encryption, message bit given to the adversary at initial coercion and the message bit generated by adversary to request the fake randomness respectively. The true random value, the value given at initial coercion and the value generated by sender for the adversary's faking request are V , V' and V_f respectively.

If the sender has done faking,

$$b_g = (1 - b)$$

$V' \rightarrow$ One of the S element in V is given as R element

$$b_{rg} = b$$

$V_f \rightarrow$ same as V

If the sender has not done faking,

$$b_g = b$$

$V' \rightarrow$ same as V

$$b_{rg} = (1 - b)$$

$V_f \rightarrow$ One of the S element in V is given as R

In this case, adversary does not know whether sender has faked or not faked. But the adversary can derive original r value by getting union of S value given for V' and V_f . Then the true message bit b can be derived using r .

One possible solution to prevent above detection is, generating the further faking messages based on initial faking randomness given to the coercer. The faking algorithm has to store the fake randomness V' at the initial coercion. When the adversary requests randomness by further faking, only S elements in the V' should be given as R . Then S_2 will be always a subset of S_1 and faking will not be detected. However, considering the encryption is full deniable, the adversary may also know the internal implementation at faking. Therefore, the adversary may ask to flush the

memory by coercion or the adversary can check the status of the memory using same detection explained above. Therefore, storing initial fake message is only useful when faking with multi-distributional characteristics.

4.5 Side Channel Attacks

Above cryptanalysis is based on the given common model where the adversary does not have access to the internal execution of encryption and decryption. But in this section we assumed that the adversary has access to monitor the execution of encryption and decryption. Consider a faking includes x number of flipping of fake bits, where x number of S elements are given as R elements and the total resource utilization is U . The resource utilization relates to generate S element is denoted by P_s and the resource utilization relates to generate R element is denote by P_r . The resource can be time, processor or memory usage. The resource utilization for fake message encryption will be $U_f = U - x * (P_s - P_r)$. In general, because of the PKC encryption, P_s and P_r will have significant differences. Therefore, faking can be detectable and detectability increases with x .



5 Conclusion

Deniable encryption introduced by Canetti *et al.* [1] is a strong cryptographic notion that provides a way to remove the committing nature of legitimate parties to the internals of encryption/decryption. In deniable encryption, the sender and/or the receiver can generate fake internals to satisfy the security verification against the original ciphertext.

This research has introduced a common model for existing full sender deniable encryption scheme. By introducing the common model, this research has defined a threat model for the common model of a full sender deniable encryption scheme. The encryption was explained in four steps and the possible analysis points were defined based on those four steps. For the common model, two main views were defined as the sender's view of encryption and the adversary's view of encryption. Based on the views defined, an elaboration was done where the full sender deniable encryption scheme is considered as a shared key encryption that encrypt true plain text message of sender's view into fake plain text message of adversary's view. The fake plain text is generated as the ciphertext of the shared key encryption. This research has showed that breaking deniability of the original encryption is equivalent to breaking a shared key encryption which has a key length equals to message length.

Initial cryptanalysis was based on the common model and further analysis was carried out on a real implementation. The implementation is based on the parity based full sender deniable encryption scheme proposed by Canetti *et al.* [1] and RSA has been as the trapdoor to generate the sparse set. Three types of sparse set generation were implemented. Two of them were based on the deniable encryption schemes introduced by Canetti *et al.* [1] and the third is a new construction. The performance of these three different sparse set generation methods is evaluated with respect to message length to cipher text ratio and the number of standard PKC decryption to retrieve a single bit of plain text. Similar to the result give by Canetti *et al.* [1], this

research has found better performance in the implementation of the second method compared to first method. The third method proposed in this research has better performance compared to first and second methods proposed by Canetti *et al.* [1].

Based on the third implementation, this research has demonstrated that with the use of probabilistic encryption like enhanced RSA instead of using deterministic encryption, which was used for the initial sparse set definition [1], the required number of PKC encryptions can be reduced from k to 1. Since the plaintext is not directly mapped to the cipher text value in probabilistic encryption, the adversary cannot detect the S -values based on previous knowledge. Therefore, the S element can be generated using only one standard PKC encryption. However, it requires future research work to derive the security requirement of probabilistic encryption to prevent statistical attack in deniable encryption.

In the first part of the cryptanalysis work, this research has showed that the adversary can reduce the possible true message space by analyzing the S -values and use that to validate the fake/non-fake message given by the sender. The limitation of faking based on $1/n$ -deniability given by Canetti *et al.* [1] has been also discussed.

In statistical cryptanalysis, this research has showed that it is possible to detect the faking by detecting the deviation in distribution of V values from expected distribution of V in non-faking. The detectability of faking increases when reducing the bit length of V .

Finally, this research has showed that it is possible to break the implemented full sender deniable scheme [1] by coercing the sender to generate more fake messages. The faking of the given scheme can be detected by analyzing any other fake message generated based on original real message and true message can be derived by collectively analyzing the fake messages generated for same real message. This research has also elaborated a straightforward method to find the real message by requesting/forcing the sender to generate a fake randomness of given fake message

derived based on initial fake message and randomness given to the adversary.



University of Moratuwa, Sri Lanka.
Electronic Theses & Dissertations
www.lib.mrt.ac.lk

References

- [1] R. Canetti, C. Dwork, M. Naor, and R. Ostrovsky, “Deniable encryption,” in *In Crypto 97*, Springer-Verlag, 1996, pp. 90–104.
- [2] D. Markus and F. D. Mandell, “Deniable encryption with negligible detection probability: an interactive construction,” in *Proceedings of the 30th Annual International Conference on Theory and Applications of Cryptographic Techniques: Advances in Cryptology*, ser. EUROCRYPT’11, Berlin, Heidelberg: Springer-Verlag, 2011, pp. 610–626.
- [3] B. M. David and A. C. A. Nascimento, “Deniable encryption from the mceliece assumptions,” *IACR Cryptology ePrint Archive*, 2011, <http://eprint.iacr.org/2011/144>.
- [4] O. Adam, P. Chris, and W. Brent, “Bi-deniable public-key encryption,” in *Proceedings of the 31st Annual Conference on Advances in Cryptology*, ser. CRYPTO’11, Berlin, Heidelberg: Springer-Verlag, 2011, pp. 525–542.
- [5] M. H. Ibrahim, “A method for obtaining deniable public-key encryption,” in *International Journal of Network Security (IJNS)*, 2009, pp. 159–164.
- [6] M. H. Ibrahim, “Receiver-deniable public-key encryption,” in *International Journal of Network Security (IJNS)*, 2009, p. 165.
- [7] B. Dan, D. Xuhua, T. Gene, and W. C. Ming, “A method for fast revocation of public key certificates and security capabilities,” in *Proceedings of the 10th Conference on USENIX Security Symposium*, ser. SSYM’01, vol. 10, Berkeley, CA, USA: USENIX Association, 2001, pp. 297–308.
- [8] H. M. Sun, Y. Chen, and J. S. Chou, *An efficient secure oblivious transfer*, Cryptology ePrint Archive, <http://eprint.iacr.org/2009/521>, 2009.
- [9] A. Sahai and B. Waters, “How to use indistinguishability obfuscation: deniable encryption, and more,” 2013, <http://eprint.iacr.org/>.

- [10] A. Menezes, P. V. Oorschot, and S. Vanstone, *Handbook of Applied Cryptography*. CRC Press, 1996.
- [11] K. P. Klonowski Marek and K. Mirosław, “Practical deniable encryption,” in *SOFSEM 2008: Theory and Practice of Computer Science*, vol. 4910, ser. Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2008, pp. 599–609.
- [12] A. Czeskis, D. J. S. Hilaire, K. Koscher, S. D. Gribble, T. Kohno, and B. Schneier, “Defeating encrypted and deniable file systems: truecrypt v5.1a and the case of the tattling os and applications,” in *Proceedings of the 3rd Conference on Hot Topics in Security*, ser. HOTSEC’08, Berkeley, CA, USA: USENIX Association, 2008, 7:1–7:7.
- [13] K. John, S. Bruce, W. David, and H. Chris, “Side channel cryptanalysis of product ciphers,” *J. Comput. Secur.*, vol. 8, no. 2,3, pp. 141–158, Aug. 2000.
- [14] B. Rikke, N. J. Buus, N. P. Sebastian, and O. Claudio, “Lower and upper bounds for deniable public key encryption,” in *Proceedings of the 17th International Conference on The Theory and Application of Cryptology and Information Security*, ser. ASIACRYPT’11, Berlin, Heidelberg: Springer-Verlag, 2011, pp. 125–142.
- [15] R. Canetti, U. Feige, O. Goldreich, and M. Naor, “Adaptively secure multi-party computation,” in *Proceedings of the Twenty-eighth Annual ACM Symposium on Theory of Computing*, ser. STOC ’96, New York, NY, USA: ACM, 1996, pp. 639–648.
- [16] H. Jaydeep, N. Vivek, and M. A. K. Basu Saikat, “Uncoercibility in e-voting and auctioning mechanisms using deniable encryption,” vol. 3, p. 97, 2011.
- [17] A. Skillen and M. Mannan, “Deniable storage encryption for mobile devices,” *Dependable and Secure Computing, IEEE Transactions on*, vol. 11, no. 3, pp. 224–237, 2014.

- [18] P. Gasti, G. Ateniese, and M. Blanton, “Deniable cloud storage: sharing files via public-key deniability,” in *Proceedings of the 9th Annual ACM Workshop on Privacy in the Electronic Society*, ser. WPES '10, New York, NY, USA: ACM, 2010, pp. 31–42.
- [19] O. Goldreich, L. A. Levin, and N. Nisan, “On constructing 1-1 one-way functions,” in *Studies in Complexity and Cryptography*, 2011, pp. 13–25.
- [20] I. B. Demgård and J. B. Nielsen, “Improved non-committing encryption schemes based on a general complexity assumption,” in *Advances in Cryptology CRYPTO 2000*, M. Bellare, Ed., vol. 1880, ser. Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2000, pp. 432–450.
- [21] B. Barak, O. Goldreich, R. Impagliazzo, S. Rudich, A. Sahai, S. Vadhan, and K. Yang, “On the (im)possibility of obfuscating programs,” in *Lecture Notes in Computer Science*, Springer-Verlag, 2001, pp. 1–18.
- [22] B. Barak, O. Goldreich, R. Impagliazzo, S. Rudich, A. Sahai, K. Yang, and S. Vadhan, “On the (im)possibility of obfuscating programs,” vol. 59, New York, NY, USA: ACM, May 2002.
- [23] G. Sanjam, G. Craig, H. Shai, R. Mariana, S. Amit, and W. Brent, “Candidate indistinguishability obfuscation and functional encryption for all circuits,” in *Proceedings of the 2013 IEEE 54th Annual Symposium on Foundations of Computer Science*, ser. FOCS '13, Washington, DC, USA: IEEE Computer Society, 2013, pp. 40–49.
- [24] S. Dreyfus, *The idiot savant's guide of rubberhose*, <http://web.archive.org>, 2012.
- [25] M. Bo and W. JiangQing, “An efficient receiver deniable encryption scheme and its applications,” *JNW*, vol. 5, no. 6, pp. 683–690, 2010.
- [26] D. Dolev, C. Dwork, and M. Naor, “Non-malleable cryptography,” in *SIAM J. on Computing*, 2000, pp. 542–552.
- [27] A. Irwin and R. Hunt, *Forensic methods for detection of deniable encryption in mobile networks in Communications*. 2009, pp. 169–174.

- [28] G. Juan, W. Daniel, and Z. H. Sheng, “Somewhat non-committing encryption and efficient adaptively secure oblivious transfer,” in *Proceedings of the 29th Annual International Cryptology Conference on Advances in Cryptology*, ser. CRYPTO '09, Berlin, Heidelberg: Springer-Verlag, 2009, pp. 505–523.
- [29] S. Choi, D. Dachman-Soled, T. Malkin, and H. Wee, “Improved non-committing encryption with applications to adaptively secure protocols,” in *Advances in Cryptology - ASIACRYPT 2009*, M. Matsui, Ed., vol. 5912, ser. Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2009, pp. 287–302.
- [30] K. Sako and J. Kilian, “Receipt-free mix-type voting scheme: a practical solution to the implementation of a voting booth,” in *Proceedings of the 14th Annual International Conference on Theory and Application of Cryptographic Techniques*, ser. EUROCRYPT'95, Berlin, Heidelberg: Springer-Verlag, 1995, pp. 393–403.
- [31] R. J. McEliece, “A public-key cryptosystem based on algebraic coding theory,” *DSN progress report*, vol. 42, no. 44, pp. D145416, 1978.
- [32] N. Ryo, I. Hideki, K. Kazukuni, and M. Kirill, “Semantic security for the mceliece cryptosystem without random oracles,” *Des. Codes Cryptography*, vol. 49, no. 1-3, pp. 289–305, 2008.
- [33] P. Christof and P. Jan, *Understanding Cryptography: A Textbook for Students and Practitioners*, 1st. Springer Publishing Company, Incorporated, 2009.
- [34] B. Schneier, *Applied Cryptography : Protocols, Algorithms, and Source Code in C*, 2nd. John Wiley & Sons, Inc., 1996.

msc.bib