# RSA-TBOS Signcryption with Proxy Re-encryption.

Varad Kirtane* , C. Pandu Rangan*
(varad.kirtane@gmail.com),(prangan55@gmail.com)

**Abstract**

The recent attack on Apple iTunes Digital Rights Management [17] has brought to light the usefulness of proxy re-encryption schemes for Digital Rights Management. It is known that the use of proxy re-encryption would have prevented the attack in [17]. With this utility in mind and with the added requirement of non-repudiation, we propose the first ever signcryption scheme with proxy re-encryption that does not involve bilinear maps. Our scheme is called RSA-TBOS-PRE and is based on the RSA-TBOS signcryption scheme of Mao and Malone-Lee [7]. We adapt various models available in the literature concerning authenticity, unforgeability and non-repudiation and propose a signature non-repudiation model suitable for signcryption schemes with proxy re-encryption. We show the non-repudiability of our scheme in this model. We also introduce and define a new security notion of Weak-IND-CCA2, a slightly weakened adaptation of the IND-CCA2 security model for signcryption schemes and prove that RSA-TBOS-PRE is secure in this model. Our scheme is Weak-IND-CCA2 secure, unidirectional, extensible to multi-use and does not use bilinear maps. This represents significant progress towards solving the open problem of designing an IND-CCA2 secure, unidirectional, multi-use scheme not using bilinear maps proposed in [15][12].

## 1 Introduction

Proxy Re-encryption is a new and interesting area of cryptography, pioneered by Blaze et al.[2]. In proxy re-encryption, a ciphertext meant for a user (say Bob) may be converted to a ciphertext meant for another user (say Charlie), with the help of a *semi-trusted third party* called *proxy*. The chief feature of this system is that this delegation is achieved without the proxy learning anything about the plaintext. Proxy Re-encryption thus models the concept of secure delegation through a semi-trusted party.

The proxy that actually carries out the transformation is semi-trusted in the sense that it is curious and thus it may attempt to discover the plaintext or secret keys using the information that is legitimately provided to it, but

*Both authors from Indian Institute of Technology Madras.

will follow the laid out protocol perfectly. This is a reasonable assumption to make in many practical scenarios, as the proxy is typically a part of the institution using the delegation mechanism. Thus, any deviation from the protocol can be easily detected and deemed malicious. This ease of detection forces the proxy to resort to passive attacks and rules out active attacks.

The primitive of proxy re-encryption has several interesting applications, as there are number of scenarios in which we wish to convert the ciphertext of one user to another, with the help of a party that cannot be fully trusted. A well-known example is that of the DRM of Apple's iTunes [17]. In March 2005, Apple's iTunes DRM was cracked by programmers who managed to steal the plaintext (song) made available during a translation from a ciphertext encrypted under a global key into a ciphertext encrypted under a key unique to each iPod. This attack was possible as the re-encryption was carried out by first decrypting the ciphertext encrypted under a global key and then encrypting it under the key for a particular iPod. If Apple had used a proxy re-encryption scheme, then the plaintext of the song would not have been available to steal. *Secure email forwarding* [3] and *distributed secure storage systems* [4] are two other examples.

All the proxy re-encryption schemes proposed to date provide the important primitive of confidentiality. However, many practical applications also demand the property of non-repudiation. This property is one of the chief characteristics of digital signature schemes. In 1997, Zheng[6] proposed the concept of signcryption, which provides both confidentiality and non-repudiation in single primitive, by performing both encryption and signing simultaneously. This primitive is more efficient than signing and encrypting separately, making it an ideal candidate for use in systems that require both confidentiality and non-repudiation.

A typical scenario that would require signcryption with proxy re-encryption would be when confidential data requiring authentication/non-repudiation must be transferred from one device to another (e.g. transfer from the iTunes server to an iPod). Proxy re-encryption would be indispensible due to the attack on iTunes described above. Signcryption would be required as authenticity/non-repudiability of the message is necessary.

## 1.1 Structure of the paper

The rest of the paper is organised as follows. We first describe the original RSA-TBOS scheme along with the proof of correctness, followed by the additional calculations that are required for proxy re-encryption. We then discuss various properties of our scheme, followed by a discussion of various non-repudiation models and a proof of non-repudiability. Then follows the discussion and definition of suitable models for security for systems like ours. We then show our scheme can be made into a multi-use one using the technique given in [13]. Finally, we conclude the paper with a recap of the important results shown and suggest directions for future research. A rigourous formal proof of Weak-IND-CCA2 security is provided in the appendix.

# 2 RSA-TBOS with Proxy Re-encryption

We describe our scheme, RSA-TBOS-PRE, a signcryption with proxy re-encryption in two parts. The first part (signcryption) is the RSA-TBOS scheme itself, which we reproduce here for convenience. The second part consists of the additional calculations required to perform proxy re-encryption. This is followed by a discussion of various properties that the scheme possesses.

## 2.1 RSA-TBOS Signcryption scheme

**Key Parameters:**

- $k$ : Even positive integer.
- Sender Alice's RSA Public & Private Key: $(N_A,e_A)$ and $(N_A,d_A)$ respectively.
  Receiver Bob's RSA Public & Private Key: $(N_B,e_B)$ and $(N_B,d_B)$ respectively.
  Note: We must have $|N_A| = |N_B| = k$.
- Two hash functions $H$ and $G$,
  where $H$: $\{0,1\}^{n+k_0} \mapsto \{0,1\}^{k_1}$ and
  $G$:$\{0,1\}^{k_1} \mapsto \{0,1\}^{n+k_0}$
  and $k = n + k_1 + k_0$, with $2^{-k_0}$ and $2^{-k_1}$ being negligible.
  Note that the output size of $H$ is greater than the input size, but is deemed a hash function as it satisfies all other properties of hash functions (such collision resistance etc.).

**Signcryption:**
When Alice signcrypts a message $M \in \{0,1\}^n$ for Bob, she performs:

1. $r \leftarrow_U \{0,1\}^{k_0}$.
2. $\omega \leftarrow H(M||r)$.
3. $s \leftarrow G(\omega) \oplus (M||r)$.
4. If $s||w > N_A$ goto 1.
5. $c' \leftarrow (s||w)^{d_A}(modN_A)$.
6. If $c' > N_B$, $c' \leftarrow c' - 2^{k-1}$.
7. $c \leftarrow c'^{e_B}(modN_B)$.

8. Send $c$ to Bob.

**Unsigncryption:**
When Bob unsigncrypts a cryptogram received from Alice, he performs:

1. $c' \leftarrow c^{d_B}(modN_B)$.
2. If $c' > N_A$, reject.
3. $\mu \leftarrow c'^{e_A}(modN_A)$.
4. Parse $\mu$ as $s||\omega$.
5. $M||r \leftarrow G(\omega) \oplus s$.
6. If $H(M||r) = \omega$, return $M$.
7. $c' \leftarrow c' + 2^{k-1}$.
8. If $c' > N_A$, reject.
9. $\mu \leftarrow c'^{e_A}(modN_A)$.
10. Parse $\mu$ as $s||\omega$.
11. $M||r \leftarrow G(\omega) \oplus s$.
12. If $H(M||r) \neq \omega$, reject.
13. Return $M$.

## 2.2 Correctness of Unsigncryption

We now show that given a valid signcrypted text, the unsigncryption algorithm returns the original plaintext. The first step of the unsigncryption merely gets back the ciphertext $c'$ from $c$, as $c'^{(d_B * e_B)} \equiv c' \ (mod\ N_B)$. At the second step, we assume that the $c' < N_B$ branch was taken during the signcryption. In this case, clearly $c' < N_A$ is true, because we did calculations modulo $N_A$ during the signcryption. The other case, namely $c' > N_B$ will handled from step 7 onwards. We then make consistency checks in steps 5 and 6. Clearly, because we had $s \leftarrow G(\omega) \oplus (M||r)$ during the signcryption, we will have $M||r = G(\omega) \oplus s$ during the unsigncryption if the ciphertext is appropriately formed. Also, as in the signcryption, we will have $\omega = H(M||r)$. Thus, if $c' < N_B$, the ciphertext unsigncrypts correctly. Now, if indeed $c' < N_B$, the algorithm will end at step 6 returning the message $M$. Else, we must add $2^{k-1}$ to the ciphertext to undo the subtraction during the signcryption. Steps 8 to 12 are identical to steps 2 to 6, and are correct by the same argument. The last step merely returns the message. Thus, even in the case $c' > N_B$ the unsigncryption works correctly.

### 2.3 Additional calculations for Proxy Re-encryption

Let us assume that Bob (the delegator) wants to delegate his work to Dave (the delegatee) with the help of the proxy. Let us assume that Dave uses an IND-CCA2 secure cryptosystem whose encryption algorithm we denote by $Enc_D$ and decryption algorithm we denote by $Dec_D$. The ReKey generation algorithm (carried out by Bob, for generation of ReKey from Bob to Dave) is as follows:
**ReKey Generation:**

1. Generate a pair $(e', d')$ such that
   - $e' \in_R Z^*_{N_B}$ ($\in_R$ denotes a uniformly random choice.)
   - $d'e' \equiv 1 \ (mod\ \phi(N_B))$
2. Calculate $d_B * e'$.
3. Calculate $Enc_D(d')$.
4. Send $ReKey_{B \to D} = (d_B * e', Enc_D(d'))$ to the proxy.

It is important to note here that the same ReKey is used everytime for re-encryption from Bob to Dave. Also, it is obvious that in case of multiple delegatees, the same value of $e'$ should not be used. Moreover, we must keep in mind that since $d$ and $e$ have similar properties, the $e'$ should not be equal to any of the $d'$ values as well. Thus, all the $(e, d)$ pairs, including $(e_B, d_B)$, must be distinct. It is easily seen that due to the abundance of values in $Z^*_{N_B}$, such a restriction can be easily met. We now show how the re-encryption and unsigncryption works.

Let $c$ be the cryptogram sent from Alice to Bob, which has to be delegated to Dave. The re-encryption algorithm (carried out by the proxy) is as follows:
**Re-encryption:**

1. Calculate $c_1 \leftarrow c^{d_B * e'} (mod\ N_B)$.
2. Send $(c_1, Enc_D(d'))$ to Dave.

Upon receiving the pair $(u, v)$ from the proxy, Dave uses the unsigncryption algorithm as follows:

1. Obtain $d' \leftarrow Dec_D(v)$.
2. Run the unsigncryption algorithm exactly as Bob would, using $u$ in place of $c$ and $d'$ in place of $d_B$.

To see that this leads to the appropriate answer, we just need to observe that $Dec_D(v) = Dec_D(Enc_D(d')) = d'$ and $c_1 = c'^{e'} (mod\ N_B)$ ; the proof of correctness of the unsigncryption algorithm is simply as explained for the basic signcryption scheme above, with $u$ in place of $c$ and $d'$ in place of $d_B$.

### 2.4 Proxy Re-Encryption Properties

RSA-TBOS-PRE is *unidirectional, non-interactive, non-transitive* and *single-use* proxy re-encryption scheme.

1. In *unidirectional* schemes, the proxy cannot compute the ReKey from $B$ to $A$, given the ReKey from $A$ to $B$. However, in a *bi-directional* scheme it can. It is clear that RSA-TBOS-PRE is unidirectional, as ReKey from $A$ to $B$ does not even involve the secret key of $B$, whereas ReKey from $B$ to $A$ requires $B$'s secret key.

2. In *non-interactive* schemes, the delegatee is not involved in the computation of ReKey, while in *interactive* schemes it is. Our scheme is obviously non-interactive, as $A$ can compute the ReKey from $A$ to $B$ by himself.

3. In *non-transitive* schemes, the ReKey from A to C cannot be computed, given the ReKeys from A to B and B to C, whereas in *transitive* schemes they can be. Our scheme is non-transitive because given $d_A.e', Enc_B(d')$ and $d_B.e'', Enc_C(d'')$, it is not possible to compute $d_A.e''', Enc_C(d''')$, for any $e'''$.

4. In *single-use* schemes a re-encrypted ciphertext cannot be re-encrypted again, whereas in a *multi-use* scheme it can be. As it stands RSA-TBOS-PRE is single-use, because once a signcryption is re-encrypted, it becomes independent of any secret keys; hence if the ReKey is used to re-encrypt again, the second component does not provide the information to decrypt it. However, we later provide an extension to our scheme that converts it to a multi-use scheme.

Additionally, we would like to mention that our scheme maintains the confidentiality of messages even if the secret key of the sender is leaked. This is an interesting property that is carried over in our system from the original RSA-TBOS scheme. We note in accordance with [5] that not all signcryptions have this property, e.g. [14][6].

## 3 Algorithms Involved

We will discuss various models of security for signcryption schemes as well as signcryption schemes with proxy re-encryption in this paper. Hence, we first define the various algorithms that will be used in these models.

- Given a security parameter $k$, the Setup algorithm returns public parameters for the sender and the receiver, along with their secret keys.

- Given the public parameters, the secret key of the delegator and a delegatee, the Extract algorithm returns the unsigncryption key corresponding to the delegatee.

- Given the public parameters, secret key of the sender, a plaintext $m$ and random input $r$ (if required by the algorithm), the SIGNCRYPT algorithm returns the ciphertext $C$, which is the signcryption of $m$ (with the random value $r$, if present).

- Given the public parameters, a delegatee and the decryption key corresponding of the delegator, the RKGEN algorithm returns a re-encryption key from the delegator to the delegatee.

- Given a re-encryption key $ReKey_{B \to D}$ and a ciphertext $C_B$ which is signcrypted for $B$, the REENCRYPT algorithm returns the ciphertext $C_D$ which is signcrypted for $D$.

- Given the unsigncryption key and a ciphertext $C$, the UNSIGNCRYPT algorithm returns the plaintext or $\perp$. Note that this is the same algorithm for the delegator as well as the delegatee, but with different keys.

- Given the signature (underlying the signcryption) the VERIFY algorithm returns $\top$ if the signature is valid or $\perp$ if it is not.

## 4 Non-Repudiation and Unforgeability

The RSA-TBOS-PRE scheme is a signcryption and therefore it must provide the features that are provided by a digital signature. Two essential qualities of a digital signature are unforgeability and non-repudiation. In an ordinary digital signature scheme, the two notions are equivalent, because if a signature is unforgeable, only the signer can produce it and hence cannot deny that he did so and if a signature is non-repudiable, then it must be unforgeable as otherwise the signer could claim that the signature under consideration was forged. However, a signcryption being an encryption as well, non-repudiation is not straightforward, as only the receiver might be able to verify the authenticity of the signature and proving the veracity of the signature to a third party may be a non-trivial task. In our scheme, however, as in the original RSA-TBOS, non-repudiation is easily achieved by the receiver decrypting the ciphertext up to step 2 and handing it over to the third party for verification.

Several security models have been proposed that deal with non-repudiation and unforgeability of signcryption schemes. We focus on the models proposed in [5]. The original paper on RSA-TBOS [7] also contains a proof for unforgeability. Let us examine the relationship between these models. Informally, the ciphertext authentication model from [5] says that if the adversary is not the sender or the receiver, it is highly unlikely for him to produce a signcryption having an valid underlying signature of $(m, \sigma)$, without having expressly queried for some signcryption of $m$ from the same sender to the same receiver. The signature non-repudiation model in [5] states that if the adversary is not the sender, then it is highly unlikely for him to produce a signcryption that has a valid underlying signature of $(m, \sigma)$ without having expressly queried for a signcryption from the same sender containing a valid underlying signature of $(m, \sigma)$. The original RSA-TBOS paper [7] shows that if the adversary is not the sender, then it is highly unlikely for him to produce a signcryption with a valid underlying signature $(m, \sigma)$ without having queried for the signcryption of $m$. Now, it is quite clear that neither of the two models from [5] is stronger than the other. Further, it is also easily seen that the signature non-repudiation model from [5] is stronger than the unforgeability model in [7] as we are allowed to signcryption of any user in the signature non-repudiation model rather than just the receiver as in the model in [7]. RSA-TBOS can be shown to be secure in the signature non-repudiation model just as in shown in [7], with the only difference being that the signcryption simulator $S_{sim}$ will now take as argument $(N, e)$ pair of any user rather than just the receiver. We refer the reader to [5] and [7] for a formal description of the models.

# 5 Non-Repudiability of RSA-TBOS-PRE

Having seen how the signature non-repudiability of the basic RSA-TBOS scheme can be proven, we now show the non-repudiability of RSA-TBOS-PRE. In order to prove non-repudiation for the RSA-TBOS-PRE scheme, we require that the sender (say $S$) use different RSA moduli for signature and encryption. With this minor cost, we will be able to guarantee the non-repudiability of our scheme.

For the signature non-repudiability of a signcryption scheme with proxy re-encryption, we propose the following attack game. Let us assume that $S$ is under attack.

**Signature Non-Repudiation Attack Game (for schemes with proxy re-encryption)**

- **Setup:** The challenger chooses a security parameter $k$ and runs $\textsc{Setup}(k)$ to obtain the public parameters and various secret keys. The public parameters are revealed to the adversary.
- **Probe Stage**: In this stage we allow the adversary the following queries:
  - $\textsc{Signcrypt}(m, r, X)$ for any plaintext $m$ and random input $r$ for any user $X$, including $S$ as the sender or the receiver.
  - $\textsc{Unsigncrypt}(X, C)$ for any user (including $S$).
  - $\textsc{Extract}(X)$ for any user excluding $S$.

- RKEXTRACT($S$,$X$) for any delegatee $X$ of $S$. The challenger responds by running RKGEN($S$,$X$) and returning the output.
- REENCRYPT($S, X, C$) for any ciphertext $C$ and any delegatee $X$ of $S$.

The following restriction may also be imposed:

- The adversary may not query both the RKEXTRACT($S$,$X$) and EXTRACT($X$) for any user $X$.

- **Forge Stage:** The adversary returns a ciphertext $C$. Let $(m, \sigma)$ be the underlying signature (obtained by partial unsigncryption). The adversary wins the game if VERIFY$(m, \sigma) = \top$, subject to the condition that no signcryption query answered contained $(m, \sigma)$ as the underlying signature.

The adversary $A$ is said to be a $(\epsilon, t)$ adversary if it outputs the signature $(m, \sigma)$ in time $t$ with advantage $\mathbf{Adv}(A) = Pr[A \text{ wins}] = \epsilon$.

**Definition (Non-Repudiation for schemes with proxy re-encryption).** *A signcryption scheme with proxy re-encryption with security parameter k is said to be non-repudiable if after the attack game described above being played with any polynomially bounded attacker A, the advantage* $\boldsymbol{Adv}(A) = Pr[A \textit{ wins}] = \epsilon$ *is a negligible function of k.*

To see why RSA-TBOS-PRE is non-repudiable, we make the simple observation that the extra RKEXTRACT, REENCRYPT and UNSIGNCRYPT do not provide any information to the adversary as the values returned pertain to a different RSA modulus than the one used for signing as per the discussion at the beginning of this section. Hence, the proof non-repudiability of the RSA-TBOS holds for RSA-TBOS-PRE as well, with exactly the same values of probability. Thus, we see that the RSA-TBOS-PRE retains the feature essential of a signcryption, i.e. non-repudiation of the sender.

# 6 Models of Security

RSA-TBOS-PRE is one of the first signcryption schemes with proxy re-encryption and thus there is no explicit mention of any security model suitable for such schemes in the literature. Hence, we must first provide such security models suitable for signcryption schemes with proxy re-encryption. The purpose of this section is to explore, discuss and eventually formally define such models.

A natural starting point for formulation of these models is to study the notions of security for signcryption schemes [11][8][5][7] and proxy re-encryption schemes[15][12] as well as the basic ideas behind these notions of security as formulated originally for encryption schemes [9]. In the following sections, we discuss and then formally define increasingly stronger security models.

## 6.1 Indistinguishable Chosen Plaintext Security (IND-CPA)

A signcryption scheme is supposed to have the properties of an encryption scheme. It is therefore fitting that we define a IND-CPA for such schemes, based on the security definitions for ordinary encryption schemes [10] as well as encryption schemes with proxy re-encryption [13] . Such a definition also makes sense for a scheme with proxy re-encryption, since it is not desirable that the confidentiality of the scheme is broken after the adversary has seen the encryptions of a few selected messages. In the IND-CPA model for encryption schemes, decryption queries are not allowed, and we disallow them in this model too. Further, since we have proxy re-encryption in this model, there is also the possibility of the adversary asking for the secret key of one or more of the delegatees, modelling the scenario that one of the delegatees is dishonest. This, however, is prohibited in this model in accordance with the IND-CPA model for proxy re-encryption schemes proposed in [13]. This restriction is sensible, because in an IND-CPA attack, the adversary is allowed to choose only plaintexts, and is not supposed to have decryption capabilities. Providing access to the secret key would violate this, and is therefore disallowed. However, we allow in this model ReKey queries to any number of delegatees in the system, because this models the proxy (who has access to ReKeys) trying to break the system. The proxy is only semi-trusted, and assumed to be curious and hence we must safeguard against the proxy learning any secret information. Note that re-encryption queries are not required since the adversary can re-encrypt himself once he has queried for the ReKey.

Formally, the following is IND-CPA the attack game. Here B is considered to be the user under attack.

**IND-CPA Attack Game**

- **Setup:** The challenger chooses a security parameter $k$ and runs $\textsc{Setup}(k)$ to obtain the public parameters and various secret keys. The public parameters are revealed to the adversary.
- **Find Stage**:
  - $\textsc{Signcrypt}(m, r)$ for any plaintext $m$ and random input $r$.
  - $\text{RK}\textsc{Extract}(B,X)$ for any delegatee $X$ of $B$. The challenger responds by running $\text{RK}\textsc{Gen}(B,X)$ and returning the output.
- **Challenge Stage**: The adversary outputs a pair $(m_0, m_1)$. The challenger then generates a random bit $b$ and returns $C^* = \textsc{Signcrypt}(m_b,r)$. Note that in this case the adversary has no control over $r$. See [7] for a detailed explanation of this.
- **Output Stage**: The adversary outputs a bit $b'$, and wins the game if $b = b'$.

The adversary $A$ is said to be a $(\epsilon, t)$ adversary if it outputs the bit $b'$ in time $t$ with advantage $\mathbf{Adv}^{IND-CPA}(A) = 2|Pr[b = b'] - 1| = \epsilon$.

**Definition (IND-CPA Security).** *A signcryption scheme with proxy re-encryption with security parameter k is said to be secure against an indistinguishable chosen plaintext attack (IND-CPA secure) if after the attack game described above being played with any polynomially bounded attacker A, the advantage* $\boldsymbol{Adv}^{IND-CPA}(A) = 2|Pr[b = b'] - 1| = \epsilon$ *is a negligible function of k.*

## 6.2 Indistinguishable Adaptive Chosen Ciphertext Security, Weak Version (Weak-IND-CCA2)

Security against IND-CPA attack game described above is necessary, but not sufficient in a practical scenario. In IND-CPA, we expressly disallow decryption queries and queries for the secret key of the delegatees. However, in a practical scenario, the attacker may indeed be one the delegatees (trying to learn the delegators secret key, decrypt messages not delegated to him etc.) which would allow him access to secret key. Multiple delegatees may also collude to attack the delegator or another delegatee. Further, these attackers may be able to temporarily get access to a decryption system of the attacked entity, similar to the scenario in IND-CCA2 attack game of traditional cryptosystems. To model these scenarios, we must allow re-encryption as well as secret key queries (with certain restrictions that ensure sensibility of the attack game, like not querying the secret key of the attacked entity etc.). The entire gamut of queries allowed and the restrictions placed on them are formally defined in the attack game that follows. Once again, $B$ is considered to be the entity under attack.

### Weak-IND-CCA2 Attack Game

- **Setup:** The challenger chooses a security parameter $k$ and runs $\textsc{Setup}(k)$ to obtain the public parameters and various secret keys. The public parameters are revealed to the adversary.

- **Find Stage**:
  - $\textsc{Signcrypt}(m, r)$ for any plaintext $m$ and random input $r$.
  - RK$\textsc{Extract}(B,X)$ for any delegatee $X$ of $B$. The challenger responds by running RK$\textsc{Gen}(B,X)$ and returning the output.
  - $\textsc{Reencrypt}(B, X, C)$ for any ciphertext $C$ and any delegatee $X$ of $B$.
  - $\textsc{Unsigncrypt}(X, C)$ for any user (including $B$).
  - $\textsc{Extract}(X)$ for any delegatee $X$ of $B$.

There are, however, the following restrictions on the queries:

- $R_1$: Both RKExtract($B$,$X$) and Extract($X$) may not be queried for any delegatee $X$.
- $R_2$: Both Reencrypt($B, X, C$) and Extract($X$) may not be queried for any delegatee $X$ any ciphertext $C$.

- **Challenge Stage**: The adversary outputs a pair $(m_0, m_1)$. The challenger then generates a random bit $b$ and returns the $C^*$=Signcrypt($m_b$,$r$). Note that in this case the adversary has no control over $r$.

- **Guess Stage**: Same queries as in the Find stage, with restrictions $R_1$ and $R_2$. There are additional restrictions as follows:

  - $R_3$: Unsigncrypt($B, C^*$) cannot be queried.
  - $R_4$: Reencrypt($B, X, C^*$) and Unsigncrypt($X, C^*$) cannot both be queried.
  - $R_5$: Unsigncrypt($X, C^*$) cannot be queried if RKExtract($B$,$X$) has been queried at any stage.
  - $R_6$: Reencrypt($B, X, C^*$) cannot be queried if Extract($X$) has been queried at any stage.

- **Output Stage**: The adversary outputs a bit $b'$, and wins the game if $b = b'$.

The adversary $A$ is said to be a $(\epsilon, t)$ adversary if it outputs the bit $b'$ in time $t$ with advantage $\mathbf{Adv}^{Weak-IND-CCA2}(A) = 2|Pr[b = b'] - 1| = \epsilon$.

**Definition (Weak-IND-CCA2 Security).** *A signcryption scheme with proxy re-encryption with security parameter $k$ is said to be secure against an weak indistinguishable adaptive chosen ciphertext attack (Weak-IND-CCA2 secure) if after the attack game described above being played with any polynomially bounded attacker $A$, the advantage $\boldsymbol{Adv}^{Weak-IND-CCA2}(A) = 2|Pr[b = b'] - 1| = \epsilon$ is a negligible function of $k$.*

### 6.3 Indistinguishable Adaptive Chosen Ciphertext Security, Strong Version (IND-CCA2)

It is possible to come up with a model of security that is slightly stronger than the Weak-IND-CCA2 proposed above. Such a model has all the security requirements one would expect of a signcryption scheme with proxy re-encryption. This model is exactly the same as the Weak-IND-CCA2 model, except that one of the restrictions is weakened. The restriction:

- $R_2$: Both Reencrypt($B, X, C$) and Extract($X$) may not be queried for any delegatee $X$ any ciphertext $C$.

is weakened to

- $R_2^*$: Both $\textsc{Reencrypt}(B, X, C^*)$ and $\textsc{Extract}(X)$ may not be queried for any delegatee $X$. ($C^*$ is the challenge ciphertext).

In other words, we allow for the adversary to query re-encryption of any ciphertext apart from the challenge ciphertext to any delegatee, even a corrupted one for which the adversary knows the secret key. We would like to point out that there is currently no signcryption scheme with proxy re-encryption that is IND-CCA2 secure. In fact, the scheme we propose is one of the first to be even Weak-IND-CCA2 secure. However, we stress that it is not impossible to achieve this higher level of security, although it is certainly non-trivial. The chief reason is that once re-encryption queries are allowed to corrupted users, junk re-encryptions can be easily detected by the adversary, and hence the simulator in the IND-CCA2 proof of security must find a way to generate legitimate re-encryptions, even if the adversary queries junk ciphertext. Preventing this querying of junk ciphertext is tantamount to public verifiability of the ciphertext, and there is currently no obvious way of achieving this for known signcryption schemes. Note that use of one-time signatures is not an option, as that would defeat the very purpose of signcryption.

# 7 Weak-CCA2 Security Proof of RSA-TBOS-PRE

Having discussed and defined various security notions in the previous section, we now formally prove the security of the RSA-TBOS-PRE scheme in the Weak-IND-CCA2 security model. The proof of security of the scheme is proved in two parts, as follows:

1. In part one, we show that the original RSA-TBOS is IND-CCA2 secure even against a stronger adversary $A_{str}$ who can make all queries allowed in standard IND-CCA2 plus two extra kinds of queries.

2. In part two, we show that if there exists an polynomially bound adversary $A'$ that can break the RSA-TBOS-PRE with non-negligible advantage, then $A'$ may be used to construct a polynomially bound strong adversary $A_{str}$ that can break the original RSA-TBOS system with non-negligible advantage.

The proof has been shifted to the appendix for brevity.

### Remark about the Proof

In our proof in the appendix, we have assumed that the secret key of the sender (say Alice) is not known to the adversary. This is in accordance with [7]. However, as we mentioned previously, even if the secret key of Alice were known to the adversary, it would be of no consequence as far as IND-CCA2 sceurity is concerned. In part one of the proof, we would simply allow the adversary $A_{str}$ to query Alice's secret key, and since the simulator generates it itself, it could easily answer this query. Notice that Alice's secret key is unrelated to the receiver's (Bob's) RSA parameters and hence it would provide no information and have no bearing on the probabilities calculated. The only advantage available after knowing Alice's secret key is that $A_{str}$ would be able to generate signcryptions for himself. However, we already provide this facility to $A_{str}$. In part two of the proof, if the adversary $A'$ queried for Alice's secret key, the constructed adversary $A_{str}$ would merely ask the RSA-TBOS Oracle for it and return the answer to $A'$. Note that the discussion on $A'$ probabilities of generating valid re-encrypted ciphertexts would still be valid, as $A'$ must still query for the hash function values.

# 8 Multi-Use RSA-TBOS-PRE

The RSA-TBOS-PRE scheme as it stands is a single use scheme, because ciphertext that is re-encrypted once cannot be used for further re-encryptions, as explained previously. However, [13] mention a technique through which many single-use schemes may be converted to multi-use schemes. Although the idea is expressed in the context of identity based schemes, our scheme is pliable enough to apply the technique in [13].

The central idea behind the conversion to a multi-use scheme is that of proxy re-encryption of the ReKey. Recall that our scheme merely requires an IND-CCA2 secure encryption scheme for encryption of the second component of the ReKey. Suppose that instead we use an IND-CCA2 secure encryption scheme with proxy re-encryption. In that case, the second component of the ReKey can now be re-encrypted further. The key to decrypt this second component must now be enclosed in a third component. This process can be continued indefinitely, each time adding one more component. We thus have a multi-use scheme. However, the chief drawback of this scheme is that it leads to ciphertext whose size expands linearly with the number of re-encryptions. However, we would like to stress that this is the state-of-the-art method as far as unidirectional schemes are concerned, and creating a multi-use scheme without linear expansion of ciphertext for unidirectional schemes is an open problem.

We mention here that the properties of signature non-repudiation and Weak-IND-CCA2 security we have proved for RSA-TBOS-PRE single-use scheme, also hold for this extension. This is simply because the extension is independent of the base system, and thus there is no new useful information available to adversary over that of the base system, and hence there are no new useful queries that the adversary can make. Thus, the adversary is restricted to the attack models in which we have already proven security, causing Multi-Use RSA-TBOS-PRE to have the properties of signature non-repudiation and Weak-IND-CCA2 security.

## 9 Conclusion and Directions for Future Work

We have extended the concept of proxy re-encryption to signcryption schemes, and have proposed the first signcryption scheme with proxy re-encryption. As proxy re-encryption has proved to be an important tool in the deployment of DRM, we think that this result is of significance to various DRM applications. The simplicity of the scheme and reliance on standard assumptions provide an added advantage. The scheme is based on the RSA-TBOS scheme of Mao et al.[7]. This scheme is more efficient than simply performing a digital signature and adding a layer of proxy re-encryption capable cryptosystem thus retaining the chief advantage of signcryption. Previous work done in this area [15] has concluded that a fundamental problem in this area is to find an IND-CCA2 secure, unidirectional, multi-use scheme. Such schemes have been proposed in [15][12]. However, these make use of bilinear maps. It has been suggested in [15] that an IND-CCA2 secure, unidirectional, multi-use scheme not involving bilinear maps is highly desirable. Our scheme is unidirectional, Weak-IND-CCA2 secure, extensible to a multi-use scheme and does not involve bilinear maps. We believe that this represents an important step in the full-fledged solution of the fundamental problem mentioned above.

This work leads to several interesting avenues of research. Arguably the most interesting and important extension would be to create a full-fledged IND-CCA2 secure scheme from our scheme, or to prove the IND-CCA2 security of our scheme if possible. Another interesting direction for research would be to find a signcryption (or even plain encryption) scheme that is secure in the standard model instead of the random oracle model [10] in which we prove security and which does not involve bilinear maps. The last, but by no means the least, important direction of investigation would be to design an extension that converts our single-use scheme to a multi-use scheme that has constant re-encrypted ciphertext size instead of the currently proposed extension that has ciphertext size that linearly expands with the number of re-encryptions.

We note here that [16] have independently proposed the idea of splitting the RSA decryption key into two parts. However, this has been done for an encryption scheme and not a signcryption scheme. [16] have proved the scheme CPA-secure, whereas our proof of security is in the Weak-IND-CCA2 model, which is strictly stronger than CPA.

## 10 Acknowledgements

The authors would like to thank an anonymous referee for pointing out the work done in [16].

## References

[1] R. Rivest, A. Shamir and L. Adleman. *A Method for Obtaining Digital Signatures and Public Key Cryptosystems.* Communications of the ACM, 21(2): pages 120-126. Feb. 1978.

[2] M. Blaze, G. Bleumer and M. Strauss, *Divertible protocols and atomic proxy cryptography.* In Advances in Cryptology - Proceedings of EUROCRYPT 1998, volume 1403 of Lecture Notes in Computer Science, pages 127-144, 1998.

[3] H. Khurana, J. Heo and M. Pant, *From Proxy Encryption Primitives to a Deployable Secure-Mailing-List Solution*, Eighth International Conference on Information and Communications Security (ICICS '06), volume 4307 of Lecture Notes in Computer Science, pages 260-281, December 2006.

[4] G. Ateniese, K. Fu, M. Green, and S. Hohenberger, *Improved Proxy Reencryption Schemes with Applications to Secure Distributed Storage.* ACM Transactions on Information and System Security (TISSEC), 9(1): pages 1-30, Feb 2006.

[5] L. Chen, J. Malone-Lee, *Improved Identity-Based Signcryption*, Public Key Cryptography 2005 (PKC 2005), volume 3386 of Lecture Notes in Computer Science, pages 362-379, Springer-Verlag, 2005.

[6] Y. Zheng, *Digital Signcryption or How to Achieve Cost (Signature & Encryption) << Cost(Signature) + Cost(Encryption).* In Advances in Cryptology - Proceedings of CRYPTO 1997, volume 1294 of Lecture Notes in Computer Science, pages 165-179. Springer-Verlag,1997.

[7] J. Malone-Lee and W. Mao, *Two Birds One Stone: Signcryption Using RSA*, In Topics in Cryptology - CT-RSA 2003, volume 2612 of Lecture Notes in Computer Science, pages 211-225, 2003.

[8] J. Baek, R. Steinfeld and Y. Zheng, *Formal Proofs for the Security of Signcryption*, Public Key Cryptography 2002 (PKC 2002), volume 2274 of Lecture Notes in Computer Science, pages 80-98. Springer-Verlag, 2002.

[9] M. Bellare, A. Desai, D. Pointcheval and P. Rogaway, *Relations Among Notions of Security for Public-Key Encryption Schemes*. In Advances in Cryptology - CRYPTO 1998, volume 1462 of Lecture Notes in Computer Science, pages 26-45. Springer-Verlag, 1998.

[10] M. Bellare and P. Rogaway, *Random Oracles are Practical: A Paradigm for Designing Efficient Protocols*, In Proceedings of the First ACM Conference on Computer and Communications Security (CCS '93), pages 62-73. 1993.

[11] J.H. An, Y. Dodis and T. Rabin, *On the Security of Joint Signature and Encryption*, In Advances in Cryptology - EUROCRYPT 2002, volume 2332 of Lecture Notes in Computer Science, pages 83-107. Springer-Verlag, 2002.

[12] J. Shao, D. Xing and Z. Cao, *Identity-Based Proxy Re-encryption Schemes with Multiuse, Unidirection, and CCA Security* . Unpublished report available at Cryptology ePrint Archive: http://eprint.iacr.org/2008/103.pdf.

[13] M. Green and G. Ateniese. *Identity-Based Proxy Re-encryption.* In Applied Cryptography and Network Security 2007 (ACNS'07). Also available at Cryptology ePrint Archive: http://eprint.iacr.org/2007/473.pdf.

[14] F. Bao and R.H. Deng, *A signcryption scheme with signature directly Verfiable by Public Key.* In Proceedings of Public Key Cryptography 1998 (PKC '98), volume 1431 of Lecture Notes in Computer Science, pages 55-59, Springer Verlag, 1998.

[15] R. Canetti and S. Hohenberger, *Chosen Ciphertext Secure Proxy Re-encryption.* In Proceedings of the 14th ACM conference on Computer and Communications Security (CCS 2007), pages 185-194, 2007. Also available at Cryptology ePrint Archive: http://eprint.iacr.org/2007/171.pdf.

[16] A. Ivan and Y. Dodis, *Proxy Cryptography Revisited.* In Proceedings of the Network and Distributed System Security Symposium (NDSS 2003), San Diego, California, USA.

[17] T. Smith, *DVD Jon: buy DRM-less Tracks from Apple iTunes*, March 18, 2005. Available at http://www.theregister.co.uk/2005/03/18/itunes_pymusique.

# Appendix

## Weak-CCA2 Security Proof of RSA-TBOS-PRE

Having discussed and defined various security notions in the previous chapter, we now formally prove the security of the RSA-TBOS-PRE scheme in the Weak-IND-CCA2 security model. The proof of security of the scheme is proved in two parts, as follows:

1. In part one, we show that the original RSA-TBOS is IND-CCA2 secure even against a stronger adversary $A_{str}$ who can make all queries allowed in standard IND-CCA2 plus two extra kinds of queries.

2. In part two, we show that if there exists an polynomially bound adversary $A'$ that can break the RSA-TBOS-PRE with non-negligible advantage, then $A'$ may be used to construct a polynomially bound strong adversary $A_{str}$ that can break the original RSA-TBOS system with non-negligible advantage.

### Part One

The modified IND-CCA2 attack game for the stronger adversary $A_{str}$ is as follows:

- **Setup**: Using the security parameter $k$, public/private keys $(e_A, d_A, N_A)$ and $(e_B, d_B, N_B)$ are generated for a target sender/receiver respectively. Also, for some pre-determined constant $\eta$ (polynomial in the security parameter $k$), $\eta$ pairs of integers $(e_i, d_i)$ satisfying the following constraints are generated.

  - $e_i * d_i \equiv 1 \ (mod\ \phi(N_B))$.
  - $e_i$'s are randomly chosen with the restriction that all the $(e_i, d_i)$ pairs are distinct from each other and from $(e_B, d_B)$.

- **Find Stage**: The adversary is provided the public parameters. He is allowed to query the following:

  - $\textsc{Signcrypt}(m||r)$ for any message $m$ and any random input $r$.
  - $\textsc{Unsigncrypt}(c)$ for any ciphertext $c$.
  - $RK_i$, for any $i, 1 \leq i \leq \eta$, answered by returning $d_B * e_i$.
  - $SK_i$, for any $i, 1 \leq i \leq \eta$, answered by returning $d_i$.

  However, there is a restriction on the $RK_i$ and $SK_i$ queries:

  - Restriction $R$: Both $RK_i$ and $SK_i$ may not be queried for any $i, 1 \leq i \leq \eta$ throughout the entire simulation (both Find and Guess stages).

- **Challenge:** The adversary $A_{str}$ returns a message pair $(m_0, m_1)$ with $|m_0| = |m_1|$. A bit $b$ is chosen uniformly at random. $C^* = \textsc{Signcrypt}(m_b, r)$ is given to the adversary. Note that in this stage the adversary has no control over $r$. This is exactly in line with the IND-CCA2 security model for encryption, where the adversary can choose only the message pair, but not any random input that may be used by the encryption algorithm. This is also in line with the IND-CCA2 security model for signcryption schemes proposed in [7].

- **Guess Stage**: The adversary is allowed the same queries as in the Find Stage, with two restrictions:

  - Restriction $R$: Both $RK_i$ and $SK_i$ may not be queried for any $i, 1 \leq i \leq \eta$ throughout the entire simulation (both Find and Guess stages).
  - $\textsc{Unsigncrypt}(C^*)$ cannot be queried.

- **Output Stage**: The adversary $A_{str}$ outputs a bit $b'$, and is said to win the game if $b = b'$.

$A_{str}$ is said to be a $(\epsilon, t)$ adversary if it outputs the bit $b'$ in time $t$ with advantage $\mathbf{Adv}_{str}^{IND-CCA2}(A_{str}) = |2.Pr[b = b'] - 1| = \epsilon$.

**Definition** (**IND-CCA2 Security**). *A signcryption scheme with security parameter $k$ is said to be secure against an indistinguishable adaptive chosen ciphertext attack (IND-CCA2 secure) against a strong attacker if after the attack game described above being played with any strong polynomially bounded attacker $A_{str}$, the advantage $\boldsymbol{Adv}_{str}^{IND-CCA2}(A) = |2.Pr[b = b'] - 1| = \epsilon$ is a negligible function of $k$.*

To prove the IND-CCA2 security against a strong adversary $A_{str}$, we will show that any such $A_{str}$ that can break the security of the system may be used to break the *partial-domain one-wayness* of the RSA permutation [1]. Thus, the IND-CCA2 security of the RSA-TBOS system will be reduced to the partial-domain one-wayness of the RSA permutation. We now formally define the notions of *one-wayness* and *partial-domain one-wayness.*

**Definition** (**One-Wayness**). *The function $f$ is $(t, \epsilon)$ one-way if the success probability of any adversary $A$ wishing to recover the preimage of $f(s||\omega)$ in time less than $t$ is upper bounded by $\epsilon$. This we state as:*

$$\boldsymbol{Adv}_f^{ow}\ (A) \leq Pr_{s,\omega}[A(f(s||\omega)) = s||\omega] < \epsilon$$

*For any $f$ we denote the maximum value of $\boldsymbol{Adv}_f^{ow}(A)$ over all adversaries running for time $t$ as $\boldsymbol{Adv}_f^{ow}(t)$.*

**Definition** (**Partial Domain One-Wayness**). *The function $f$ is $(t, \epsilon)$-partial domain one-way if the success probability of any adversary $A$ wishing to recover the partial preimage of $f(s||\omega)$ in time less than $t$ is upper bounded by $\epsilon$. This we state as:*

$$\boldsymbol{Adv}_f^{pd-ow}(A) \leq Pr_{s,\omega}[A(f(s||\omega)) = \omega] < \epsilon$$

*For any $f$ we denote the maximum value of $\boldsymbol{Adv}_f^{pd-ow}(A)$ over all adversaries running for time $t$ as $\boldsymbol{Adv}_f^{pd-ow}(t)$.*

In our case the function $f$ is the RSA permutation[1].

Let us consider a simulator $Sim'$, based on the simulator $Sim$ in the original RSA-TBOS proof, that plays the IND-CCA2 attack game defined above with the adversary $A_{str}$ to break the partial-domain one-wayness of RSA permutation. Let us also assume that $A_{str}$ is able to break the IND-CCA2 security of RSA-TBOS with advantage $\epsilon$ in time $t$. Let us assume that the $Sim'$ is given a random element $C^*$ from the ciphertext space which is to be partially inverted. Note that the randomness of $C^*$ is paramount as it ensures that we can partially invert any ciphertext, and not just a special class of ciphertexts. $Sim'$ obtains the partial inversion of $C^*$ by playing the IND-CCA2 attack game with $A_{str}$. We mention that the simulation that follows is largely similar to the original simulation by [7]. For $Sim'$ to play the attack game, it is necessary to show how to respond to $A_{str}$'s queries to the random oracles $G$ and $H$, the signcryption/unsigncryption oracles and $RK_i$ and $SK_i$. We denote the algorithms to do this as $G_{sim}$, $H_{sim}$, $S_{sim}$, $U_{sim}$ and $RKSK_{sim}$ respectively and we describe them below. To make our simulations sound we keep five lists, $L_G$, $L_H$ and $L_{RKSK}$ that are initially empty. Each list will consist of query/response pairs to that oracle. The list $L_H$ will also store some extra information as described in $H_{sim}$ below. At the end of the simulation, we hope to find the partial preimage of $C^*$ among the queries in $L_G$. For the remainder of the proof we assume that $A_{str}$ makes at most $q_g, q_h, q_s, q_u$ and $q_{rksk}$ queries to the oracles $G_{sim}$, $H_{sim}$, $S_{sim}$, $U_{sim}$ and $RKSK_{sim}$ respectively.

| $U_{sim}(c)$ | $S_{sim}(m\|\|r)$ |
|---|---|
| If $(m\|\|r, \omega, c) \in L_H$ for some $m$: | Run $H_{sim}(m\|\|r)$ |
| Return $m$ | Search $L_H$ for entry $(m\|\|r, \omega, c)$ |
| Else Reject | Return $c$ |

$G_{sim}(\omega)$
If $(\omega, x) \in L_G$ for some $x$:
Return $x$
Else:
$x \leftarrow^R \{0,1\}^{n+k_0}$
Add $(\omega, x)$ to $L_G$
Return $x$

$H_{sim}(m||r)$
If $(m||r, \omega, c) \in L_H$ for some $\omega$:
Return $\omega$
Else:
1. $\omega \leftarrow^R \{0,1\}^{k_0}$
2. $x \leftarrow G_{sim}(\omega)$
3. $s \leftarrow x \oplus (m||r)$
4. If $(s||\omega > N_A)$ , goto 1
5. $c' \leftarrow (s||\omega)^{d_A}\ (mod\ N_A)$
6. If $c' > N_B$, $c' \leftarrow c' - 2^{k-1}$
7. $c \leftarrow c'^{e_B}\ (mod\ N_B)$
8. Add $(m||r, \omega, c)$ to $L_H$
9. Return $\omega$

$RKSK_{Sim}(arg, i)$
If $arg = RK$ and $i \in L_{RK}$ for some $(i, RK_i)$:
Return $RK_i$.
Else If $arg = SK$ and $i \in L_{SK}$ for some $(i, SK_i)$:
Return $SK_i$.
Else If $arg = RK$
Generate $RK_i \leftarrow^R Z^*_{N_B}$, such that $RK_i \notin L_{RK}$ and $RK_i \notin L_{SK}$
Add $(i, RK_i)$ to $L_{RK}$.
Return $RK_i$.
Else If $arg = SK$
Generate $SK_i \leftarrow^R Z^*_{N_B}$, such that $SK_i \notin L_{RK}$ and $SK_i \notin L_{SK}$
Add $(i, SK_i)$ to $L_{SK}$.
Return $SK_i$.

Note that in $H_{sim}$ above we assume that each query has form $m||r$. This just means that each query has length $n + k_0$ bits and so may be parsed as $m||r$ where $m$ has $n$ bits and $r$ has $k_0$ bits. This assumption is justified because, in the random oracle model, it would not help $A_{str}$ to make queries of length different from $n + k_0$, as the answer provided is merely a randomly generated number having no correlation at all with any other query. We also allow $A_{str}$ to make queries of the form $m||r$ to $S_{sim}$ i.e. we allow $A_{str}$ to provide its own random input. This is consistent with a CCA2 attack on an encryption scheme such as RSA-PSS where an adversary can encrypt messages itself using its own random input. This is also consistent with the IND-CCA2 attack game in [7]. At the beginning of the challenge stage, $A_{str}$ outputs $m_0$ and $m_1$ . We choose a bit $b$ uniformly at random and supply the adversary with $C^*$ as the signcryption of $m_b$. Suppose $C^* = f(s^*||\omega^*)$, where $f$ is the RSA-TBOS transformation. This places the following constraints on the random oracles $G$ and $H$:

$$H(m_b||r^*) = \omega^*\ and\ G(\omega^*) = s^* \oplus (m_b||r^*) \qquad (1)$$

We denote by $\mathsf{AskG}$ the event that during $A_{str}$'s attack $\omega^*$ has ended up in $L_G$. We denote by $\mathsf{AskH}$ the event the query $m||r^*$ has ended up in $L_H$ for some $m$. If $\omega^* \notin L_G$ , then $G(\omega^*)$ is undefined and so $r^*$ is a uniformly distributed random variable. Therefore the probability that there exists an $m$ such that $m||r^* \in L_H$ is at most $2^{-k_0}.(q_h + q_s)$. This value comes from the fact that $r$ has $k_0$ bits and $L_H$ gets updated during $H_{sim}$ queries as well as $S_{sim}$ queries. The above argument tells us that:

$$Pr[\mathsf{AskH}|\neg\mathsf{AskG}] \leq 2^{-k_0}.(q_h + q_s) \tag{2}$$

Our simulation $U_{sim}$ can only fail if it outputs reject when it is presented with a valid ciphertext. We denote this event $\mathsf{UBad}$. Suppose that $U_{sim}$ is queried with $c = f(s||\omega)$ (where $f$ is the RSA-TBOS transformation) and let

$$m||r = G(\omega) \oplus s \tag{3}$$

We may mistakenly reject a valid ciphertext if $H(m||r) = \omega$, while $m||r$ is not in $L_H$. Suppose that this query occurs before $C^*$ is given to $A_{str}$ then, since $m||r$ is not in $L_H$ , $H(m||r)$ will take its value at random. If this query is made after $C^*$ is given to $A_{str}$ then the restriction $C \neq C^*$ laid out in the attack game means that $(m, r) \neq (m_b, r^*)$ and so (1) above is irrelevant. In either case $H(m||r)$ may take its value at random which means that

$$Pr[\mathsf{UBad}] \leq 2^{-(k_1-1)}.q_u \tag{4}$$

Note that the extra factor of 2 ($-k_1$ becomes $-k_1 + 1$) comes because there are two possibilities for the ciphertext to be valid, corresponding to $c' < N_B$ and $c' > N_B$.

Our simulation $RKSK_{sim}$ may fail if $A_{str}$ is able to detect that the values returned are different from the actual values. In the Indistinguishability Lemma at the end of part one, we show that if all the $(e_i, d_i)$ pairs are chosen randomly and distinct from each other and from $(e_B, d_B)$, then the adversary will be unable to detect any discrepancy. In $RKSK_{sim}$, we return random and distinct values that correspond to $d_B * e_i$ and $d_i$. Hence, we are automatically and uniquely defining $(e_i, d_i)$ pairs. However, since $\phi(N_B)$ is unknown to us, we cannot verify this and hence there is a possibility that some of them happen to be the same. Let us call this event $\mathsf{RSBad}$. Hence $\neg\mathsf{RSBad}$ is the event that $\mathsf{RSBad}$ does not occur. Clearly,

$$Pr[\mathsf{RSBad}] = 1 - Pr[\neg\mathsf{RSBad}] \tag{5}$$

We note that $A_{str}$ can make at most $\eta$ distinct $RK_i/SK_i$ queries, each asking for either $d_B * e_i$ or $d_i$. We want that the $(e, d)$ pair defined by these value be distinct from each of the previous ones. Hence, we have the following:

$$\begin{aligned}
Pr[\neg\mathsf{RSBad}] &= \frac{(\phi(N_B)-2)}{\phi(N_B)} * \frac{(\phi(N_B)-4)}{\phi(N_B)} * \cdots * \frac{(\phi(N_B)-2\eta)}{\phi(N_B)} \\
&\geq \frac{\phi(N_B)-2\eta^{\eta}}{\phi(N_B)} \\
&= (1-\frac{2\eta}{\phi(N_B)})^{\eta} \\
&\geq 1-\frac{2\eta*\eta}{\phi(N_B)} \ldots [(1-x)^{\eta} \geq 1-\eta x,\ when\ |x|<1]
\end{aligned}$$

Therefore, we finally have

$$Pr[\mathsf{RSBad}] \leq \frac{2\eta^2}{\phi(N_B)} \tag{6}$$

Notice that in the above equation, the numerator is $O(poly(\eta))$ whereas the denominator is $O(2^{\eta})$, where $\eta$ itself is polynomial in the security parameter $k$.

Let us define the event $\mathsf{Bad}$ as

$$\mathsf{Bad} = \mathsf{AskG} \cup \mathsf{AskH} \cup \mathsf{UBad} \cup \mathsf{RSBad} \tag{7}$$

Let us denote the event that the adversary wins, i.e. it outputs $b'$ such that $b'=b$, by $\mathsf{S}$. In the event $\neg\mathsf{Bad}$ the bit b is independent of our simulations, and therefore independent of the adversary's view. We infer from this that:

$$Pr[\mathsf{S}|\neg\mathsf{Bad}] = \frac{1}{2} \tag{8}$$

Also, in the event $\neg\mathsf{Bad}$, the adversary interacts with a perfect simulation of all oracles. This gives

$$Pr[\mathsf{S}\cap\neg\mathsf{Bad}] \geq \frac{1}{2}+\frac{\epsilon}{2}-Pr[\mathsf{Bad}] \tag{9}$$

Equation (8) gives us

$$Pr[\mathsf{S}\cap\neg\mathsf{Bad}] = Pr[\mathsf{S}|\neg\mathsf{Bad}].Pr[\neg\mathsf{Bad}] = \frac{1}{2}.(1-Pr[\mathsf{Bad}]). \tag{10}$$

From (9) and (10) we get

$$Pr[\mathsf{Bad}] \geq \epsilon \tag{11}$$

From (7) we have

$$\begin{aligned}
Pr[\mathsf{Bad}] &\leq Pr[\mathsf{AskG}\cup\mathsf{AskH}] + Pr[\mathsf{UBad}] + Pr[\mathsf{RSBad}] \\
&= Pr[\mathsf{AskG}] + Pr[\mathsf{AskH}\cap\neg\mathsf{AskG}] + Pr[\mathsf{UBad}] + Pr[\mathsf{RSBad}] \\
&\leq Pr[\mathsf{AskG}] + Pr[\mathsf{AskH}|\neg\mathsf{AskG}] + Pr[\mathsf{UBad}] + Pr[\mathsf{RSBad}]
\end{aligned} \tag{12}$$

Together, (4), (6) and (12) give

$$Pr[\mathsf{AskG}] \geq \epsilon - 2^{-k_0}.(q_h + q_s) - 2^{-k_1}.q_u - 2\eta^2.(\phi(N_B))^{-1} \tag{13}$$

Note that in our simulation we use $C^*$, a random element as the ciphertext. This gives us an extra consideration in our simulation. We say that the simulation is Good if

(i) $C^{*d_B}\ (mod\ N_B) < N_A$

(ii) $gcd(C^{*d_B}\ (mod\ N_B), N_A) = 1$

Over random choices of $(N, e_B)$, $(N_B, d_B)$, $C^*$ and $N_A$, we have

$$Pr[(i)] = \frac{1}{2}$$
$$Pr[(ii)|(i)] \geq 1 - 2^{-(k/2)+(3/2)}$$

This gives us

$$Pr[\mathsf{Good}] \geq (2^{-1} - 2^{-\frac{k}{2}+\frac{1}{2}}) \tag{14}$$

Hence from equation (13) we now get

$$Pr[\mathsf{AskG}|\mathsf{Good}] \geq \epsilon - 2^{-k_0}.(q_h + q_s) - 2^{-k_1}.q_u - 2k^2.(\phi(N_B))^{-1} \tag{15}$$

We are interested in the event $\mathsf{AskG} \cap \mathsf{Good}$. We have

$$Pr[\mathsf{AskG} \cap \mathsf{Good}] = Pr[\mathsf{AskG}|\mathsf{Good}].Pr[\mathsf{Good}] \tag{16}$$

Thus, (13), (15) and (16) give us

$$\begin{aligned} Pr[\mathsf{AskG} \cap \mathsf{Good}] \geq (2^{-1} - 2^{-\frac{k}{2}+\frac{1}{2}}).(\epsilon - 2^{-k_0}.(q_h + q_s) \\ - 2^{-k_1}.q_u - 2k^2.(\phi(N_B))^{-1}) = \delta \end{aligned} \tag{17}$$

Now, in the event $\mathsf{AskG} \cap \mathsf{Good}$ we recover a set $L_G$ of size

$$q = q_g + q_s + q_h \tag{18}$$

that contains the $k_1$ least significant bits of $z_0^*$ where

$$(z_0^{*d_A}\ (mod\ N_A))^{e_B}\ mod\ N_B = C^*.$$

Recovering the partial pre-image of $C^*$ from these bits is done exactly as in the original proof in [7].

Hence, proceeding exactly as in [7] we get the following Lemma. Note that $\nu$ represents the number of simulation runs required for recovery of the bits.

**Lemma** (**RSA-TBOS Security**). *Let $A_{str}$ be an adversary that uses a IND-CCA2 attack with RK and SK queries to attempt to break RSA-TBOS with security parameter $k$. Suppose that $A_{str}$ succeeds with probability $\epsilon$ in time $t$ after making at most $q_g, q_h, q_s, q_u$ and $q_{rksk}$ queries to $G$, $H$, the signcryption oracle, the unsigncryption oracle and RKSK oracle respectively. In the random oracle model for $G$ and $H$ we may use $A_{str}$ to partially invert RSA with probability $\epsilon'$ in time $t'$ where*

$$\epsilon' \geq \delta^{\nu} - 2^{-k/8}$$
$$t' \leq \nu.t + (q_g + q_h + q_s)^{\nu}.poly(k) + 2.\nu.(q_h + q_s).T$$

$\lceil (5k)/(4k_1) \rceil$, *and $T$ is the time taken for modular exponentiation.*

**Lemma** (**Indistinguishability of random values**). *Even if the adversary $A_{str}$ queries either $RK_i$ or $SK_i$ for all $i, 1 \leq i \leq \eta$, they are indistinguishable from random distinct values from $Z^*_{N_B}$ as along as the $(e_i, d_i)$ pairs defined by them are distinct from each other and from $(e_B, d_B)$.*

*Proof.* Without loss of generality, let us assume that $A_{str}$ queries $RK_i$ for $1 \leq i \leq \eta'$ and $SK_i$ for $\eta' < i \leq \eta$. The entire knowledge of the adversary can be summed up in the following three types of equations.

**Type I**

$$e_B * d_B + \mu_0.\phi(N_B) = 1$$

**Type II**

$$\begin{aligned} d_B * e_1 + \mu_1.\phi(N_B) &= a_1 \\ d_B * e_2 + \mu_2.\phi(N_B) &= a_2 \\ &\vdots \\ d_B * e_{\eta'} + \mu_{\eta'}.\phi(N_B) &= a_{\eta'} \end{aligned}$$

**Type III**

$$\begin{aligned} e_{(\eta'+1)} * d_{(\eta'+1)} + \mu_{(\eta'+1)}.\phi(N_B) &= 1 \\ e_{(\eta'+2)} * d_{(\eta'+2)} + \mu_{(\eta'+2)}.\phi(N_B) &= 1 \\ &\vdots \\ e_{\eta} * d_{\eta} + \mu_{\eta}.\phi(N_B) &= 1 \end{aligned}$$

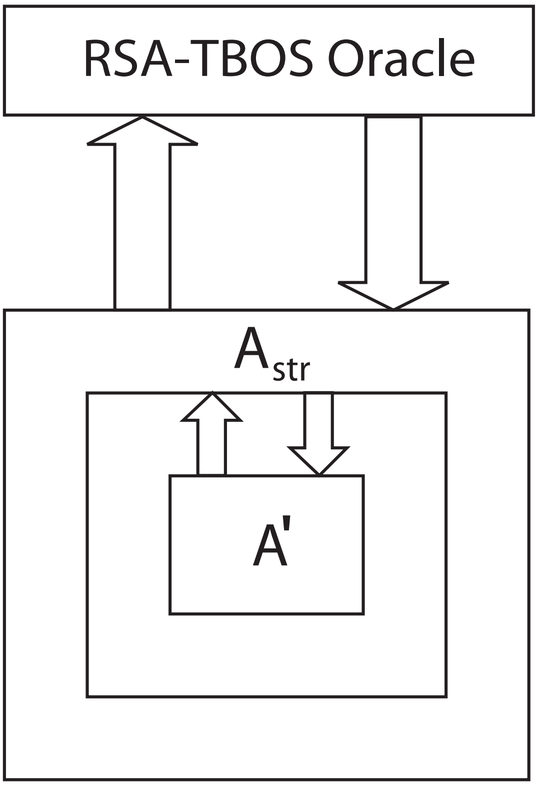
If we now are able to show that any subset of these equations has atleast one more unknown than the number of equations, it would mean that our queries do not enable the adversary to compute any secret information. Note that this is a necessary but not sufficient condition. Consider the following kinds of subsets:

- *Type I only*: There is only one equation and $d_B$ and $\mu_0.\phi(N_B)$ are two unknowns.
- *Type II only*: In the worst case scenario, all the $\mu$'s are zero. Still, if we take $i$ of these equations, $e_1 \ldots e_i$ and $d_B$ are $i+1$ unknowns.
- *Type III only*: Here, the worst case scenario is that all the $\mu$'s are equal. Yet, if we take $i$ of these equations, $d_B * e_{(\eta'+1)} \ldots d_B * e_{(\eta'+i)}$ and $\mu * \phi(N_B)$ are $i+1$ unknowns.
- *Type II and III only*: The worst case here is that all the $\mu$'s are equal. It is easily seen that the number of $d_B * e$'s will be equal to the number of equations and $\mu * \phi(N_B)$ will be the extra unknown. Another worst case arises when the $\mu$'s in Type II equations are zero and those in Type III are equal. We see that the number of $e$'s will be equal to the number of equations and $\mu * \phi(N_B)$ will be an extra unknown.
- *Type I, II and III*: This case is exactly the same as the previous one, with $d_B$ playing the part of $e$ for the Type I equation.

In order to see that any random value from $Z^*_{N_B}$ can serve as $RK_i$ or $SK_i$, consider the following three tuple : $(e_i, d_i, d_B * e_i)$. Notice that no matter what value from $Z^*_{N_B}$ we call as $d_i$ it will automatically determine the value of $e_i$ and $d_B * e_i$. Thus, as long as the adversary does not know $e_i$ and $d_B * e_i$, there is no way he can detect a discrepancy. Similarly, no matter what value from $Z^*_{N_B}$ we call as $d_B * e_i$ it will automatically determine the value of $e_i$ and $d_i$. Again, as long as the adversary does not know $e_i$ and $d_i$, there is no way he can detect a discrepancy. □

### Part Two

In this part we reduce the IND-CCA2 security of RSA-TBOS-PRE to the IND-CCA2 security of the RSA-TBOS scheme with a strong adversary $A_{str}$. For this, let us assume that there exists an adversary $A'$ that can break the IND-CCA2 security of RSA-TBOS-PRE with $\mathbf{Adv}^{IND-CCA2}(A') = |2.Pr[b = b'] - 1| = \epsilon$ in time $t$. To create $A_{str}$ using $A'$ as a routine, we let $A_{str}$ take over all the communications from $A'$ to the external world. $A_{str}$ itself communicates with a RSA-TBOS oracle. This situation is sketched in the diagram below.




We have in section X, given the Weak-IND-CCA2 attack game. We now show how to answer $A'$'s queries through queries made to the RSA-TBOS oracle. **$A_{str}$'s responses**

- **Setup:** The RSA-TBOS oracle chooses a security parameter $k$ and runs $\textsc{Setup}(k)$ to obtain the public parameters and various secret keys. The public parameters are revealed to the $A_{str}$, who passes them on to $A'$.

- **Find Stage**:

  – $A'$ asks $\textsc{Signcrypt}(m, r)$ for any plaintext $m$ and random input $r$ : $A_{str}$ responds by querying the same to the RSA-TBOS oracle and returning the answer to $A'$.

  – $A'$ asks $\text{RK}\textsc{Extract}(B, X)$ for any delegatee $X$ of $B$: $A_{str}$ responds by querying $RK_i$ (corresponding to the ReKey for $X$) to the RSA-TBOS oracle and returning ($RK_i$, $R$), where $R$ is a random ciphertext from the range of $Enc_X$. Note that as $Enc_X$ is IND-CCA2 secure, it is possible to distinguish a random value

  - $A'$ asks REENCRYPT$(B, X, C)$ for any ciphertext $C$ and any delegatee $X$ of $B$: $A_{str}$ responds by querying $RK_i$ (corresponding to the ReKey for $X$) to the RSA-TBOS oracle and returning $C^{RK_i}$ $(mod N_B)$.
  - $A'$ asks UNSIGNCRYPT$(X, C)$ for any user (including $B$): If the user is $B$, $A_{str}$ responds by querying the same to the RSA-TBOS oracle. Else, $A_{str}$ responds by querying $SK_i$ (corresponding to the secret key for $X$), running the UNSIGNCRYPT algorithm himself and returning the unsigncrypted value to $A'$.
  - $A'$ asks EXTRACT$(X)$ for any delegatee $X$ of $B$: $A_{str}$ responds by querying $SK_i$ (corresponding to the secret key for $X$).

- **Challenge Stage**: $A'$ outputs a pair $(m_0, m_1)$. $A_{str}$ simply passes this on to the RSA-TBOS oracle, and returns the RSA-TBOS oracle's response to $A'$.

- **Guess Stage**: $A_{str}$ responds as in the Find stage.

- **Output Stage**: $A'$ outputs a bit $b'$, and $A_{str}$ simply passes this to the RSA-TBOS oracle as his response. We have taken over all the communication of $A'$, and hence $A_{str}$ must also provide $A'$ with access to public hash functions $G$ and $H$. $A_{str}$ does this by asking the RSA-TBOS oracle these queries and returning (as well as recording) the answers.

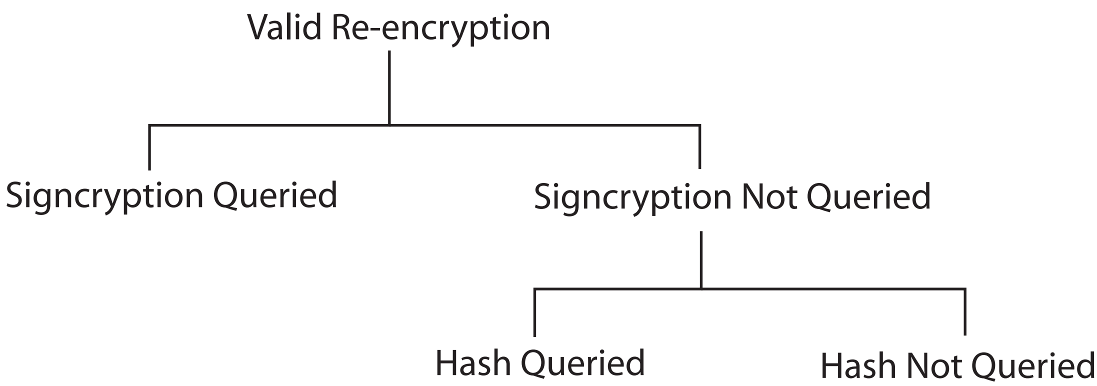
Note that $A_{str}$ has the restriction that it may not query the RSA-TBOS oracle for both $RK_i$ and $SK_i$ for any $1 \leq i \leq \eta$. We must also show that Weak-IND-CCA2 attack game imposes such restrictions on $A'$ that $A_{str}$ will never need to query $RK_i$ and $SK_i$ for any $1 \leq i \leq \eta$ and yet answer $A'$ queries correctly with high probability. From the way $A_{str}$ answers, we see that there are potentially three scenarios in which $A_{str}$ might have to query the RSA-TBOS oracle for both $RK_i$ and $SK_i$ for some $1 \leq i \leq \eta$:

1. $A'$ queries for the ReKey as well as the secret key of a particular user.

2. $A'$ queries for the secret key of a particular user and asks for a re-encryption to him.

3. $A'$ queries for the ReKey of a particular user and then asks for an unsigncryption to him.

We note that the first two scenarios are impossible, as both these are expressly forbidden in the Weak-IND-CCA2 security model. To answer an unsigncryption query after the adversary has already queried the ReKey, $A_{str}$ uses the signcryption, ReKey and hash function queries $A'$ has made. $A'$ can create a valid re-encrypted ciphertext in the following ways (the leaves of the tree diagram):




Let us consider each of the leaves as a separate case.

- Ask for a signcryption and re-encrypt it using the ReKey: In this case, by applying the each of the queried ReKeys to each of the queried signcryptions, $A_{str}$ can figure out which message was queried and return this to $A'$.

- Create the re-encrypted ciphertext from hash function queries to $H$: In this case, the list of queries for $H$ will contain $m||r$, where $m$ is the unsigncrypted message and $A_{str}$ can use the ReKey queries and $H$ queries to figure out which of the $H$ queries actually has the message embedded and return this message as the unsigncryption.

- Create the re-encrypted ciphertext without hash function queries to $H$ and without signcryption queries: In this case, the $H(m||r)$ implicitly defined by the signcryption can only be correct with a probability $2^{-(k_1-1)}$. Let $\mathsf{Bad}$ be the event that this happens and we are thus unable to answer $A'$'s query appropriately. If $A'$ makes $q_u$ unsigncryption queries, we have:
$$Pr[\mathsf{Bad}] \leq 2^{-(k_1-1)}.q_u \tag{19}$$

Let $\mathsf{S}$ be the event that $A_{str}$ wins the game with the RSA-TBOS oracle. We know that in case $\mathsf{Bad}$ does not occur, $A_{str}$ will have the same advantage as $A'$. Hence,
$$Pr[\mathsf{S}|\neg\mathsf{Bad}] \geq \frac{1}{2} + \frac{\epsilon}{2} \tag{20}$$

Consider now the following:

$$\begin{aligned}
Pr[\mathsf{S}] &= Pr[\mathsf{S} \cap \mathsf{Bad}] + Pr[\mathsf{S} \cap \neg\mathsf{Bad}] \\
&\geq Pr[\mathsf{S} \cap \mathsf{Bad}] \\
&= Pr[\neg\mathsf{Bad}].Pr[\mathsf{S}|\neg\mathsf{Bad}] \\
&\geq (1 - 2^{-(k_1-1)}.q_u).(\frac{1}{2} + \frac{\epsilon}{2}) \\
&= \frac{1}{2} + \frac{\epsilon}{2} - 2^{-k_1}.q_u - 2^{-(k_1-1)}.q_u.\frac{\epsilon}{2} \\
&= \frac{1}{2} + \frac{1}{2}.(\epsilon - 2^{-(k_1-1)}.q_u[1+\epsilon]) \\
&\geq \frac{1}{2} + \frac{1}{2}.(\epsilon - 2^{-(k_1-2)}.q_u)
\end{aligned}$$

Thus, from an $(\epsilon, t)$ adversary $A'$ that can break the IND-CCA2 security RSA-TBOS-PRE, we can create an $(\epsilon', t')$ adversary $A_{str}$ that can break IND-CCA2 security of RSA-TBOS, where:

$$\begin{aligned}
\epsilon' &\geq \epsilon - 2^{-(k_1-2)}.q_u \\
t' &\leq t + poly(k)
\end{aligned}$$

Note that the second inequality holds as all of $A_{str}$'s extra operations can be done in polynomial time.