SETUP in Secret Sharing Schemes

Ruxandra F. Olimid

E-mail: ruxandra.olimid@fmi.unibuc.ro

Abstract. Secret sharing schemes split a secret into multiple shares that are usually distributed to distinct participants with the goal that only authorized subsets of participants can recover it. We show that SETUP (Secretly Embedded Trapdoor with Universal Protection) attack can be embedded in schemes that employ enough randomness to give the attacker an overwhelming advantage to access the secret. In case of ideal schemes, a coalition of a few participants (within at least one is the attacker) can succeed the attack, while in case of non-ideal schemes the attacker knowledge can be enough to reveal the secret. We exemplify the proposed attack against Shamir's threshold scheme, as being the most well-known and used secret sharing scheme. Finally, we consider some prevention techniques against the attack.

Key words: Secret Sharing, SETUP, Black-Box Cryptography.

1 Introduction

SETUP (Secretly Embedded Trapdoor with Universal Protection) mechanism was introduced by Young and Yung [17]. It represents a malicious technique performed by the manufacturer of a cryptosystem that consists in implementing a subliminal channel that leaks encrypted secret information. The encryption is performed using the attacker's public key and therefore he is the only one that gains access to the leaked information.

From its introduction, SETUP attack was applied to different encryption systems, signatures schemes, key generation algorithms, e-voting schemes and network protocols [1, 4, 6, 10–13, 17–20, 22, 23]. Defense techniques against this kind of attack were also analyzed [7, 8].

In this paper, we consider the embedding of the SETUP mechanism in a secret sharing device. A secret sharing cryptographic device permits the splitting of a secret into multiple shares that are securely distributed to participants. The secret can be recovered if and only if an authorized subset of participants agrees with its reconstruction.

We show that under certain conditions, the manufacturer can maliciously modify the cryptographic sharing device so that it permits a specific participant (the attacker) to gain an overwhelming advantage to access the secret. More precisely, the attacker becomes able to determine the secret by himself or with a help of a few other participants, without the requirement to form an authorized group.

The attack is possible if the sharing mechanism is performed in a black-box model that receives as input the secret and distributes the output shares to the participants. In such a model, the malicious behavior remains hidden for the owner of the secret sharing device, who considers the device trustable. He inputs a secret into the cryptographic device, which is in charge of the generation and the distribution of shares without being aware that a participant may gain access to the secret information. The owner of the device is unable to determine the malicious behavior because of the indistinguishability of the outputs towards the genuine device.

We propose a general method of performing the SETUP attack against secret sharing schemes that employ enough randomness. In order to exemplify the applicability of the proposed mechanism, we present the embedding of SETUP in the most popular secret sharing scheme, Shamir's [15].

The paper is organized as follows. Section 2 gives the preliminary notions. Section 3 introduces the SETUP attack against secret sharing schemes that use enough random values. Section 4 analyses the security of the proposed attack. We exemplify the usability of the proposed mechanism in Section 5. Section 6 considers the prevention techniques against the proposed attack. Finally, we conclude in Section 7.

2 Background

2.1 SETUP

When a cryptosystem is implemented as a black-box, a user can only access its inputs and outputs. The internal design, the implementation of the algorithm and the internal memory are not externally accessible.

The black-box model permits the attacker to change the internal design in order to obtain a unique advantage. While the modification does not apparently affect the inputs or outputs of the cryptosystem, the user cannot suspect any malicious behavior. This is accomplished by the SETUP mechanism, defined by Young and Yung as follows [18]:

Definition 1. *Assume that C is a black-box cryptosystem with a publicly known specification. A (regular) SETUP mechanism is an algorithmic modification made to C to get C' such that:*

1. *The input of C' agrees with the public specifications of the input of C;*
2. *C' computes efficiently using the attacker's public encryption function e (and possibly other functions as well), contained within C';*
3. *The attacker's private decryption function d is not contained within C' and is known only by the attacker;*
4. *The output of C' agrees with the public specifications of the output of C;*
5. *The output of C and C' are polynomially indistinguishable (as in [5]) to everyone except the attacker;*

6. *After the discovery of the specifics of the SETUP algorithm and after discovering its presence in the implementation (e.g. reverse-engineering of hardware tamper-proof device), users (except the attacker) cannot determine past (or ideally, future) keys.*

A modified cryptosystem that implements SETUP is called *contaminated* [17].

2.2 Secret Sharing Schemes

A secret sharing scheme is a method of splitting a secret S into n ($n > 1$) shares, which are then securely distributed to the participants. The secret can be recovered only when an authorized subset of participants combines their shares together. The set of all authorized subsets is called the *access structure*.

Definition 2. *The access structure of a (k, n) threshold secret sharing scheme consists of all sets whose cardinality is at least k.*

Therefore, in case of a (k, n) threshold secret sharing scheme at least k out of n shares are required for a successful reconstruction.

Definition 3. *A secret sharing scheme is called ideal if the space of all possible secrets equals the space of all possible shares.*

We emphasize that in an ideal secret sharing scheme the size of a share equals the size of the secret.

The rest of the paper focuses on centralized secret sharing schemes that use random values. Such schemes involve the presence of a trusted party (the dealer) who splits the secret into shares, which are computed based on some random values. The majority of those involve the following two phases:

1. Selection of the random values;
2. Computing the shares $C_1, \ldots, C_n$ (share C_j is distributed to participant P_j, $1 \leq j \leq n$) based on the random values previously selected.

In Section 3 we show that if the previous two steps are performed in the black-box model and the number of the randomly chosen values is sufficiently large, then a SETUP attack can be mounted on the secret sharing device.

An example of ideal secret scheme that performs the required two steps is Shamir's [15]:

```
Input: $n$ the number of participants, $q \geq n + 1$ a prime number and $S \in \mathbb{Z}_q$ the secret.
Output: $C_1, C_2, \ldots, C_n$ the shares corresponding to the secret $S$.
1: $n$ distinct and public elements $x_1, x_2, \ldots, x_n \in \mathbb{Z}_q$ are chosen for the participants ($x_j$ for $P_j$, $1 \leq j \leq n$).
```

2: A $k-1$ degree random polynomial

$$f(x) = a_0 + a_1 x + \cdots + a_{k-1} x^{k-1} \pmod{q}$$

is picked, where $a_0 = S$ and $a_j \in \mathbb{Z}_q$, $(1 \leq j \leq k-1)$.

3: Each participant P_j $(1 \leq j \leq n)$ receives his share:

$$C_j = f(x_j)$$

The reconstruction is based on polynomial interpolation: given at least k points (x_i, C_i) with distinct x_i's, the polynomial $f(x)$ satisfying $C_i = f(x_i)$, $1 \leq i \leq k$, is unique and can be found by interpolation:

$$f(x) = \textstyle\sum_{i=1}^{k} C_i \prod_{1 \leq j \leq k, i \neq j} \dfrac{x - x_i}{x_i - x_j}$$

The secret S is evaluated as $f(0)$. Since any k or more participants can recover the secret, Shamir's scheme is a (k, n) threshold secret sharing scheme.

We will mount a SETUP attack on Shamir's scheme in Section 5.

2.3 Public Key Encryption from Trapdoor Permutations

Diffie and Hellman introduced the notions of one-way functions and trapdoor functions in 1976 [3]. The particular classes of one-way permutation and trapdoor permutation require that the domain and range of the functions are equal. The most well-known example of a trapdoor permutation is RSA [14].

This section introduces the general method to construct public key cryptosystems from trapdoor permutations [2], [16]. We will use this construction in Section 5 to exemplify the proposed SETUP attack in Shamir's ideal secret sharing scheme.

Definition 4. *Let X be a set. A function $f : X \to X$ is a one-way permutation if it is:*

- *easy to compute: there exists a polynomial time algorithm that on input $x \in X$ outputs $y = f(x)$;*
- *hard to invert: for all probabilistic polynomial time algorithms, the probability that on input $y \in X$ outputs $x \in X$ such that $f(x) = y$ is negligible;*

Definition 5. *A pair $(\widetilde{Gen}, f)$ is a family of secure trapdoor permutation if the following holds:*

- *$\widetilde{Gen}$ is the parameter generation algorithm, which on input 1^λ, outputs a pair (pk, sk) and a function $f_{pk} : X \to X$;*
- *without the knowledge the trapdoor sk, f_{pk} is a one-way function;*
- *with the knowledge of the trapdoor sk, the inverse $f_{pk}^{-1}(y)$ of f_{pk} is easily computable, $\forall y \in X$.*

Definition 6. *Let $(\widetilde{Gen}, f)$ be a family of secure trapdoor permutations. A deterministic probabilistic polynomial time algorithm hc, which on inputs pk and $x \in X$, outputs a single bit $hc_{pk}(x)$ is called hard-core predicate if for all probabilistic polynomial time algorithms $\mathcal{A}$:*

$$Pr[\mathcal{A}(pk, f_{pk}(x)) = hc_{pk}(x)] \leq 1/2 + negl(\lambda)$$

Let $(\widetilde{Gen}, f)$ be a family of trapdoor permutations and hc an associated hard-core predicate as described before. A public key encryption scheme (Gen, Enc, Dec) can be constructed as:

– Gen is the key generation algorithm, which on input 1^λ runs $\widetilde{Gen}$ in order to obtain the public key pk and the private key sk;
– Enc is the randomized encryption algorithm, which on inputs a public key pk and a message $M = (M_1, \ldots, M_l) \in \{0,1\}^l$, randomly chooses $x_1 \in X$, computes $x_i = f_{pk}(x_{i-1})$, for all $i > 1$ and outputs the ciphertext $C = (c_0, c_1, \ldots, c_l) = (x_{l+1}, hc_{pk}(x_1) \oplus M_1, \ldots, hc_{pk}(x_l) \oplus M_l)$;
– Dec is the deterministic decryption algorithm, which on inputs a private key sk and a ciphertext C outputs $M = (hc_{pk}(x_1) \oplus c_1, \ldots, hc_{pk}(x_l) \oplus c_l)$, where $x_i = f_{pk}^{-1}(x_{i+1})$, $1 \leq i \leq l$.

Theorem 1. *If $(\widetilde{Gen}, f)$ is a family of secure trapdoor permutations and hc is a corresponding hard-core predicate, then the public key cryptosystem (Gen, Enc, Dec) is semantically secure.*

The proof is beyond the purpose of this paper. The reader may refer to the bibliography for more details [2], [9], [16].

3 The SETUP Attack

The main goal of SETUP is to offer the attacker an overwhelming advantage to reconstruct the secret.

A trivial attack could be considered at first: in the distribution phase, the attacker receives the (encryption of the) secret instead of a valid share. The honest participants will not be able to determine the dishonest behavior if the reconstruction is not performed. However, in case of reconstruction, if the attacker is not be able to provide a valid share, then the attack will be revealed. We highlight that his is not the case when the encryption of the secret represents a valid share.

We propose such a technique for which, in the worst-case scenario, the attacker can reveal the secret with the knowledge of a few other shares. Any participant other than the attacker (or his allies) needs all the shares of participants forming an authorized subset in order to reconstruct the secret.

The proposed SETUP attack becomes possible under the following assumptions:

1. The sharing mechanism is implemented as a black-box that can store information across multiple invocations of the sharing algorithm in a non-volatile memory;
2. The sharing mechanism selects a set of random values and uses them to compute the shares (phases 1 and 2 mentioned in Subsection 2.2 are performed within the black-box);
3. The number of random values in each share is greater or equal to the number of the components of the shared secret (or the share itself can be considered a random value);
4. The shares are distributed through secure channels;
5. The attacker is always one of the participants;
6. Several secrets are shared.

Assumptions 1 and 4 are general assumptions for SETUP, respectively secret sharing schemes.

Assumptions 2 and 3 specify the properties that a particular secret sharing should meet to be vulnerable to the proposed attack.

Assumption 5 states that the attacker must be one of the participants. For the rest of the paper, we consider (without loss of generality) that the attacker is the first participant P_1. Moreover, he must always receive a specific share that is computed in a special way within the contaminated device. In case of an ideal secret sharing scheme, the attacker needs access to at least one more share of a specific participant, which may be an ally.

The attack fails if only one secret is shared. However, assumption 6 does not restrict the applicability of the SETUP attack, because usually the number of shared secrets is large in practice. It is uncommon to believe that a sharing device is used only once or it is restored to factory defaults after each run.

We consider the following notations: $\mathcal{S}$ the set of all possible secrets and $\otimes$ the group operation in $\mathcal{S}$; $\mathcal{C}$ the set of all possible shares; $H : \{0,1\}^* \to \mathcal{S}$ a cryptographically strong hash function; ID a random and secret bit string of considerable length that uniquely identifies the sharing device.

Let (Gen, Enc, Dec) be a semantically secure public key encryption scheme, where:

– Gen is the key generation algorithm, which on input 1^λ outputs a key pair (pk, sk), where pk is the public key and sk is the secret key;
– Enc is the randomized encryption algorithm that on input a public key pk and a message $M \in \mathcal{S}$, outputs a ciphertext $C \in \mathcal{C}^t$, $t \in \mathbb{Z}^*$ fixed.
– Dec is the deterministic decryption algorithm that on input a private key sk and a ciphertext $C \in \mathcal{C}^t$, outputs the plaintext $M \in \mathcal{S}$.

We highlight that the security against passive adversaries is enough, due to the general assumption that shares are distributed through secure channels (assumption 4).

As an extra requirement, the cryptosystem must accept any value in $\mathcal{C}^t$ as a valid ciphertext: $\forall C \in \mathcal{C}^t$ and pk public key, $\exists M \in \mathcal{S}$ such that $Enc_{pk}(M) = C$.

The election of the public key encryption system is performed so that $t \in \mathbb{Z}^*$ is minimum, while the system remains semantically secure. The high speed of the encryption represents an advantage.

The contaminated sharing mechanism uses a public key encryption system as described above. Let (pk, sk) be the pair of public and secret keys of the attacker. When a secret is given as input, the contaminated device encrypts it using the public key of the attacker that is stored in the non-volatile memory of the black-box and outputs it as one or more shares. The secret key sk is not stored within the device and cannot be recovered from pk as the encryption system is secure. Therefore, the attacker is the only one that can benefit of the leaked information.

Besides the public key pk of the attacker, the random string ID is also stored in the non-volatile memory of the contaminated device.

As we have already mentioned, we consider that multiple secrets are shared. Each secret is shared during one round. A round consists of the following steps: the secret is given as input, the internal algorithm runs, outputs the shares and distributes them to the participants. Let S^i be the secret to be shared in round i $(i \geq 1)$.

Given the previous assumptions and notations, we define the SETUP attack as follows:

Input: n the number of participants and $S^1, S^2, \ldots$ the secrets to be shared.
Output: for each round $i \geq 1$, $C_1^i, C_2^i, \ldots, C_n^i$ the shares that correspond to the secret S^i.

```
1: if i==1 then
2:     $\alpha \in \mathcal{C}$ is randomly chosen.
3:     $C_1^1 = \alpha$.
4:     $H(ID||C_1^1)$ is stored in memory for further usage.
5:     $C_2^1, C_3^1, \ldots, C_n^1$ are computed accordingly to the genuine secret sharing
       scheme, taking into consideration the share $C_1^1$.
6: else
7:     $(C_1^i, \ldots, C_t^i) = Enc_{pk}(S^i \otimes H(ID||C_1^{i-1}))$.
8:     $H(ID||C_1^{i-1})$ is replaced in the non-volatile memory by $H(ID||C_1^i)$.
9:     $C_{t+1}^i, C_{t+2}^i, \ldots, C_n^i$ are computed accordingly to the genuine secret sharing
       scheme, taking into consideration the shares $C_1^i, \ldots, C_t^i$.
10: end if
```

It is easy to observe that the attacker P_1 can determine the secret S^i $(i > 1)$ if he knows the first t shares of the current round and his share form the previous round:

Input: $C_1^i, \ldots, C_t^i$ the first t shares in round i, C_1^{i-1} the share of the attacker P_1 in round $i-1$, ID the random string and sk the secret key.
Output: S^i the secret in round i $(i > 1)$.

```
1: $S^i = Dec_{sk}((C_1^i, \ldots, C_t^i)) \otimes (H(ID||C_1^{i-1}))^{-1}$.
```

P_1 can benefit of an advantage only if he convinces $P_2, \ldots, P_t$ to divulge him their shares $C_2^i, \ldots, C_t^i$. We remark that on distinct runs of the protocol the roles of $P_2, ..., P_t$ can be played by different participants, but the attacker must know their identity. If this is the case, then a predefined algorithm that maps these participants for each round of the protocol must be implemented in the black-box and known to P_1.

Possibly the most convenient situation is when $P_2, \ldots, P_t$ are fixed and they are allies of P_1. This way, they will always provide their secret shares to the attacker who becomes able to reconstruct the secret. We highlight that although $P_2, \ldots, P_t$ are allies, they cannot find the secret unless they own the secret key sk. If it is desired that some of the allies have the same advantage in restoring the secret, then they will be given the secret key.

The attack is practical when t is small. For large values of t the attack may become difficult to implement or even useless. For example, in case of a (k, n) threshold scheme, when $t \geq k$ the attacker has no advantage in restoring the secret.

In case of ideal secret sharing schemes the value of t is lower bounded by 2 ($t \geq 2$) because a semantically secure public key cryptosystem for which the plaintexts space equals the ciphertexts space ($\mathcal{S} = \mathcal{C}$) does not exist. We emphasize that $t = 1$ may be possible for non ideal secret sharing schemes, since the space of all possible shares is larger than the space of all possible secrets. This means that the attacker could restore the secret by himself, without any help from other participants.

No matter the case, P_1 can never recover the first secret S^1. This is because the attacker's first share must be random in order to achieve the property of indistinguishability, which it is described in more detail in the next section.

4 Security analysis of the proposed SETUP Attack

The section analyzes the two requirements any SETUP mechanism must achieve: output indistinguishability and secret confidentiality.

4.1 Output Indistinguishability

The SETUP mechanism should be indistinguishable from the genuine one for everyone except the attacker. If the attack were easily identifiable, then the users would change the contaminated sharing device for a more trustable one.

In order to prove the indistinguishability, we show that the outputs of the contaminated device do not restrict the possible space of values and maintain the same distribution as the outputs of the genuine device.

The intuition behind the demonstration is the following. The shares $C_1^i, \ldots, C_t^i$ are computed starting from a random value α using computations that do not restrict the possible space of values and maintain the same distribution: the group operation in $\mathcal{S}$ and the encryption function from $\mathcal{S}$ to $\mathcal{C}^t$. This makes them indistinguishable by construction. The rest of the shares are computed in the same way as in the genuine scheme, therefore there are also indistinguishable.

We give next the indistinguishability proof for the proposed SETUP attack.

Theorem 2. *The proposed SETUP attack achieves output indistinguishability.*

Proof. From the construction of the SETUP attack, the proof is complete if we show the indistinguishability of the first t shares for all possible rounds $i \geq 1$.

We perform a proof by induction on the number of secrets that are shared.

In the first round, the share of the attacker is randomly chosen $C_1^1 = \alpha \in \mathcal{C}$ and all the other shares are computed as in the genuine version of the scheme. Therefore, the indistinguishability property holds.

We assume by induction that for a fixed $i > 1$ $C_1^{i-1}, \ldots, C_t^{i-1}$ are indistinguishable from uniformly distributed values in $\mathcal{C}^t$. Since H acts like a random oracle, the value $H(ID||C_1^{i-1})$ is random in $\mathcal{S}$. Due to the selection of the public key cryptosystem (it is semantically secure and it covers the whole space C^t), Enc_{pk} applied to a random message maintains the indistinguishability of the ciphertext from a random value in $\mathcal{C}^t$.

4.2 Confidentiality

A SETUP attack must achieve confidentiality against reverse engineering. In this scenario, the content of the non-volatile memory, which contains the public key pk, the random string ID and the hashed value $H(ID||C_1^{i-1})$ can be accessed. We highlight that C_1^{i-1} cannot be revealed from the hashed value under the assumption that H is a strong hash function. Otherwise, any $k-1$ coalition of participants can compute the secret that was shared in the previous round.

We show that the additional information extracted by reverse engineering brings no advantage. The scheme remains secure for unauthorized set of participants, even though they have access to the information stored in the memory of the device.

Theorem 3. *The contaminated secret sharing scheme remains as secure as the genuine version for anyone except the owners of the secret key (the attacker and, if desired, his allies).*

Proof. Let $\mathcal{P}$ be a coalition of r participants, $1 \leq r \leq n$, which gain access to the public key pk, the string ID and the hashed value $H(ID||C_1^{i-1})$ through performing reverse engineering on the device.

If $\mathcal{P}$ is an authorized set, then the theorem holds since its members can recover the secret in both the genuine and the contaminated version of the secret sharing scheme.

If $\mathcal{P}$ is an unauthorized set of participants, then, in the genuine scheme, its members are not able to compute any secret S^i, $i \leq 1$. We will next analyze two possible scenarios for the contaminated version.

In the first scenario, the attacker or at least one of his allies are not members of $\mathcal{P}$. Therefore, the only difference from the genuine scheme is that they may reveal the content of the non-volatile memory. However, the stored values are independent from the secret, so they do not provide any information regarding the secret itself. The only useful information may be the hashed value of the attacker's share. But this is kept secret under the assumption that H is a cryptographic strong hash function.

In the second scenario, the attacker and all of his allies are members of $\mathcal{P}$. Therefore, in addition to the information from the previous scenario, the users have also access to the encryption of the secret. However, this provides them no advantage under the assumption that the used cryptosystem is semantically secure.

5 SETUP Attack Applied to Shamir's Secret Sharing Scheme

We exemplify the applicability of the proposed SETUP attack for Shamir's secret sharing scheme. We motivate our choice by the fact that Shamir's scheme is the most popular scheme in the literature.

Shamir's scheme is ideal, therefore the attacker cannot succeed by himself (Section 3). We give an optimal solution, which requires a single ally.

In order to do that, we first have to define a public key cryptosystem that satisfies the requirements mentioned in Section 3 for $t = 2$ and $\mathcal{S} = \mathcal{C} = \mathbb{Z}_q$. This is achieved by the general method of constructing public key cryptosystem from trapdoor functions (Section 2.3), with some modifications.

Let $(\widetilde{Gen}, f)$ be a family of trapdoor permutations with $f_{pk} : \mathbb{Z}_q \to \mathbb{Z}_q$. Rivest, Shamir and Adleman gave a solution to restrict a RSA trapdoor permutation to $\mathbb{Z}_q$ in the original paper [14]: a plaintext is repeatedly encrypted until the result lies in $\mathbb{Z}_q$. Similarly, a ciphertext is repeatedly decrypted until it is obtained a value in $\mathbb{Z}_q$. The least-significant bit is a hardcore hc of the RSA family of trapdoor permutations. Therefore, we define the public key encryption scheme (Gen, Enc, Dec) as follows:

- Gen is the key generation algorithm, which on input 1^λ runs $\widetilde{Gen}$ in order to obtain the public key pk and the private key sk;
- Enc is the randomized encryption algorithm, which on inputs a public key pk and a message $M = (M_1, \ldots, M_l) \in \mathbb{Z}_q$, randomly chooses $x_1 \in X$, computes $x_i = f_{pk}(x_{i-1})$, $i > 1$ and then $C = (c_0, c_1, \ldots, c_l) = (x_{l+1}, hc_{pk}(x_1) \oplus M_1, \ldots, hc_{pk}(x_l) \oplus M_l)$, where l is chosen such that $|\mathbb{Z}_q| \approx |\{0,1\}^l|$. If $C \in \mathbb{Z}_q^2$ then outputs C as ciphertext. Otherwise, it restarts the encryption using a different value x_1.
- Dec is the deterministic decryption algorithm, which on inputs a private key sk and a ciphertext C outputs $M = (hc(x_1) \oplus c_1, \ldots, hc(x_l) \oplus c_l)$, where $x_i = f_{pk}^{-1}(x_{i+1})$, $1 \leq i \leq l$.

Theorem 4. *The proposed public key cryptosystem remains semantically secure.*

Proof. The proof is by contradiction. Suppose for the sake of contradiction that the proposed encryption system is not semantically secure. Let $Game0$ be the corresponding Ind-CPA experiment. Therefore, an adversary $\mathcal{A}$ has a non-negligible advantage in winning the Ind-CPA game:

$$Adv^{Ind-CPA}_{\mathcal{A},Game0} \geq \epsilon_1(\lambda)$$

Let's now consider $Game1$ the Ind-CPA experiment for the public key cryptosystem obtained from trapdoor permutations as explained in Section 2.3. We remark that for $C \in \mathbb{Z}_q^2$, $Game1$ is identical to $Game0$, so the adversary wins with non-negligible probability.

For l such that $|\mathbb{Z}_q| \approx |\{0,1\}^l|$, the probability that the challenger's response is a ciphertext that lies in $\mathbb{Z}_q^2$ is non-negligible:

$$Pr[C \in \mathbb{Z}_q^2] = \frac{|\mathbb{Z}_q^2|}{|\{0,1\}^{2l}|} \geq \epsilon_2(\lambda)$$

Therefore, the advantage of the attacker in winning $Game1$ is lower bounded by the non-negligible value:

$$Adv^{Ind-CPA}_{\mathcal{A},Game1} \geq \epsilon_1(\lambda)\epsilon_2(\lambda)$$

This contradicts Theorem 1, so the proposed public key cryptosystem is semantically secure.

We can now introduce the SETUP attack for Shamir's secret sharing scheme:

```
Input: $n$ the number of participants, $q \geq n+1$ a prime number and $S^i \in \mathbb{Z}_q$, $i \geq 1$, the secrets to be shared.
Output: for each round $i \geq 1$, $C_1^i, C_2^i, \ldots, C_n^i$ the shares corresponding to the secret $S^i$.
1: $n$ distinct elements $x_1, x_2, \ldots, x_n \in \mathbb{Z}_q$ are chosen for the participants ($x_j$ for $P_j$, $1 \leq j \leq n$).
2: if i==1 then
3:     A $k-1$ degree random polynomial
           $f^1(x) = a_0^1 + a_1^1 x + \ldots. + a_{k-1}^1 x^{k-1} \pmod q$
       is picked, where $a_0^1 = S^1$ and $a_j^1 \in \mathbb{Z}_q$ $(1 \leq j \leq k-1)$.
4:     Each participant $P_j$ $(1 \leq j \leq n)$ receives his share:
           $C_j^1 = f^1(x_j)$
5:     $H(ID||C_1^1)$ is stored in the non-volatile memory.
6: else
7:     $(C_1^i, C_2^i) = Enc_{pk}(S^i \cdot H(ID||C_1^{i-1}) \pmod q)$.
8:     $H(ID||C_1^{i-1})$ is replaced in the non-volatile memory by $H(ID||C_1^i)$.
9:     $a_j^i \in \mathbb{Z}_q$, $(3 \leq j \leq k-1)$ are randomly chosen.
10:    $a_1^i$ and $a_2^i$ are the solutions for the system of equations:
           $\begin{cases} C_1^i = f^i(x_1) \\ C_2^i = f^i(x_2) \end{cases}$
11:    The $k-1$ degree polynomial of the round is:
           $f^i(x) = S^i + a_1^i x + \ldots. + a_{k-1}^i x^{k-1} \pmod q$
```

```
12:     Each participant $P_j (1 \leq j \leq n)$ receives his share:
                $C_j^i = f^i(x_j)$
13: end if
```

It is easy to see that the contaminated secret sharing scheme is correctly defined, in the sense that $C_j^i = f^i(x_j)$, $\forall 1 \leq j \leq n, i \geq 1$.

The attacker can compute with probability 1 any shared secret, except the first one, if he knows the share of his ally:

Input: C_1^i, C_2^i the shares of the attacker and his ally in round i, C_1^{i-1} the share of the attacker P_1 in round $i-1$, ID the random string and sk the secret key.

Output: S^i the secret in round i $(i > 1)$.

```
1: $S^i = Dec_{sk}((C_1^i, C_2^i)) \cdot (H(ID||C_1^{i-1}))^{-1} \pmod q$.
```

6 SETUP Attack Prevention Techniques

We have shown that, under certain condition, a SETUP attack is possible in secret sharing schemes that use randomly selected values to compute the shares. As the majority of the assumptions we have made are easy to meet in practice, they do not restrict the applicability of the proposed attack. For example, the usage of a secret sharing device for sharing multiple secrets is natural, as well as the attempt of one participant's (the attacker) to recover the secret.

However, we have also introduced a restriction in the sense that the attacker should always be the first participant (P_1). The order of participants is not important, so we could have considered any other participant as being the attacker. The true restriction is given by the fact that the contaminated sharing device mechanism knows to distribute the proper shares to the attacker and his allies. If the attacker does not receive the maliciously computed share that contains the encrypted secret leaked as a random value and he does not know which are the shares of his allies, then the attack cannot be mounted. The precise distribution of the contaminated share to the corresponding participants is possible if the sharing device is in charge of the shares distribution. Therefore, an immediate protection against a SETUP attack is to use a sharing device that computes the shares, but does not distribute them to the participants. The distribution remains the responsibility of the dealer.

Although the dealer is in charge with the distribution, his improper behavior may maintain the applicability of the attack. Consider for example that he inputs the secret into the contaminated sharing device, maps each output to a participant in the sense that the first output represents the share that will be given to a participant, the second output represents the share that will be given to another participant and so on and maintains the same mapping for several rounds. The probability that the attack succeeds is given by the probability that he manages to restore the valid ciphertext as a concatenation of his and his allies shares:

$$Pr_{\mathcal{A}} = \frac{1}{A_n^t} = \frac{(n-t)!}{n!}$$

Of course, the success of the attack considerable diminishes in case the dealer performs the distribution as described before. For example, SETUP attack succeeds with probability 5% in Shamir's secret scheme with $n = 5$ participants.

However, a slightly modification on the contaminated device raises the success probability in the proposed scenario by a factor of n/t. It consists in computing as many groups of shares as possible in the same way as the shares of the attacker and his allies are computed. This way, no matter how the dealer performs the mapping, if the attacker and his allies receive as input such shares, then the attacker will be able to recover the secret. The attack remains secure against reverse engineering and maintains outputs indistinguishability.

The best prevention techniques against the proposed SETUP attack is achieved if the dealer randomly maps the outputs to the participants in each round. The existence of a SETUP attack that permits the attacker to recover the secret in the conditions of a random mapping remains an open problem.

7 Conclusions

We showed that, under certain conditions, SETUP attack can be embedded in secret sharing schemes that use uniformly randomly selected values to give the attacker an overwhelming advantage to access the shared secret: in case of ideal schemes the attack is performed by a coalition of a few participants (within at least one is the attacker), while in case of non-ideal schemes the attacker's knowledge may be enough to reveal the secret. To the best of our knowledge, we are the first to consider SETUP attack against secret sharing.

We proposed a general method of implementing the attack and analyzed its properties. In order to exemplify the applicability of our proposal, we successfully embedded SETUP in the most popular secret sharing scheme: Shamir's scheme becomes susceptible in case the attacker has only one ally.

In the last part of the paper, we considered some prevention techniques that can be successfully applied in practice to avoid the attack.

References

1. K.Anjan, J.Abraham, "Behavioral Analysis of Transport Layer Based Hybrid Covert Channel", Recent Trends in Network Security and Applications, Springer-Verlag, pp. 83-92, 2010.
2. M.Blum, S.Goldwasser, "An Efficient Probabilistic Public-Key Encryption Scheme Which Hides All Partial Information", Advances in Cryptology, Proceedings of CRYPTO '84, Springer, vol. 196, pp. 289-302, 1984.
3. W. Diffie, M. Hellman,"New directions in cryptography", IEEE Transactions on Information Theory, vol. 22(6), pp. 644-654, 1976.

4. M. Gogolewski, M. Klonowski, P. Kubiak, "Kleptographic attacks on e-voting schemes", Proceeding of the 2006 International Conference on Emeerging Trends in Information and Communication Security, pp. 494-508, 2006.
5. S. Goldwasser, S. Micalli, "Probabilistic encryption", J. Comp. Sys. Sci.28, pp. 270-299, 1984.
6. Z. Golebiewski, M. Kutylowski, F. Zagorski, "Stealing Secrets with SSL/TSL and SSH - Kleptographic Attacks", Cryptology and Network Security, LNCS, vol.4301/2006, pp.191-202, 2006.
7. D. Kucner, M. Kutylowski, "Stochastic kleptography detection", Public-key Cryptography and Computational Number Theory Proceedings of the International Conference, Stefan Banach International Mathematical Center, pp. 137-149, 2001.
8. D. Kucner, M. Kutylowski, "How to use un-trusty cryptographic devices", Tatra Mountains Mathematical Publications, vol.29, pp. 57-67, 2004.
9. J.Katz, Y.Lindell, "Introduction to Modern Cryptography", Chapman & Hall/CRC, pp.287-290,373-378, 2008.
10. E. Mohamed, H. Elkamchouchi, "Kleptographic Attacks on Elliptic Curve Cryptosystems", International Journal of Computer Science and Network Security, vol.10(6), pp. 213-215, 2010.
11. E. Mohamed, H. Elkamchouchi, "Kleptographic Attacks on Elliptic Curve Signatures", International Journal of Computer Science and Network Security, vol.10 (6), pp. 264-267, 2010.
12. E. Mohamed, H. Elkamchouchi, "Elliptic Curve Kleptography", International Journal of Computer Science and Network Security, vol.10 (6), pp. 183-185, 2010.
13. C.Patsakis, N.Alexandris, "A New SETUP for Factoring Based Algorithms", 6th International Conference on Intelligent Information Hiding and Multimedia Signal Processing, pp.200-203, 2010.
14. R.L. Rivest, A. Shamir, L.E. Adelman, "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems ", Communications of the ACM, vol.21 (2), pp. 120-126, 1978.
15. A. Shamir, "How to share a secret", Communications of the ACM vol.22 (11), pp. 612-613, 1979.
16. A.C.-C. Yao, "Theory and Applications of Trapdoor Functions", 23rd Annual Symposium on Foundations of Computer Science, IEEE, pp. 80-91, 1982.
17. A. Young, M. Yung, "The dark side of "black-box" cryptography or: Should we trust capstone?", Advanced in Cryptology - CRYPTO' 96, Springer-Verlag, pp. 89-103, 1996.
18. A. Young, M. Yung, "Kleptography: Using Cryptography Against Cryptography", Advances in Cryptology - CRYPTO '97, Springer-Verlag, pp. 62-74, 1997.
19. A. Young, M. Yung, " The prevalence of kleptographic attacks on discrete-log based cryptosystems", Advances in Cryptology - CRYPTO '97, Springer-Verlag, pp. 264-276, 1997.
20. A. Young, M. Yung, "Malicious Cryptography: Kleptographic Aspects", Proceedings of CT-RSA'2005. pp. 7-18, 2005.
21. Young, A., Yung, M.: Malicious Cryptography: Exposing Cryptovirology, pp. 231, 243-244, Wiley Publishing, 2004.
22. A. Young, M. Yung, "Space-efficient Kleptography Without Random Oracles", Proceedings of the 9th International Conference on Information Hiding, Springer-Verlag, pp. 112-129, 2007.
23. A. Young, M. Yung, "Kleptography from standard assumptions and application",Security and Cryptography for Networks, Springer-Verlag, pp. 271-290, 2010.