SETUP in Secret Sharing Schemes using Random Values

Ruxandra F. Olimid

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

Abstract. Secret sharing schemes divide a secret among multiple participants so that only authorized subsets of parties can reconstruct it. We show that SETUP (Secretly Embedded Trapdoor with Universal Protection) attack can be embedded in secret sharing 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's knowledge can be enough to reveal the secret. We exemplify the attack against Shamir's threshold scheme, which is the most well-known and used secret sharing scheme. Finally, we consider some prevention techniques against the proposed 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 [15]. 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 has been applied to different encryption systems, signatures schemes, key generation algorithms, e-voting and network protocols [1,3,4,7,8,9,10,15,16,17,18,20,21]. Defense techniques against this kind of attack were also analyzed [5,6].

In this paper, we consider the embedding of the SETUP mechanism in secret sharing schemes. A secret sharing cryptographic device divides a secret into multiple shares that are distributed to participants through secured channels. The secret can be subsequently 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 such 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 the help of a few other participants (without the cooperation of an authorized subset).

The attack is possible if the sharing device is modeled as a black-box (the inputs and outputs are available, but the internal algorithm is inaccessible). In such a model, the owner of the device is unable to determine the malicious behavior because of the indistinguishability of the outputs towards the genuine device. He inputs a secret to the cryptographic device that is in charge of the generation (and the distribution) of shares, without being aware that the adversary can gain access to the secret.

We propose a general method to perform 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 [11].

The paper is organized as follows. Section 2 gives the preliminaries. Section 3 introduces the SETUP attack against secret sharing schemes that use enough random values. Section 4 analyses the security of the proposed attack. Section 5 exemplifies the applicability against Shamir's scheme. Section 6 considers some prevention techniques against the proposed attack. Section 7 concludes.

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 insides of a device with the goal 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:

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 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. [16]*

A modified cryptosystem that implements SETUP is called *contaminated* [15].

2.2 Secret Sharing Schemes

A secret sharing scheme divides a secret S into n $(n > 1)$ shares, which are then securely distributed to the participants. The secret is reconstructed only when an authorized subset of parties 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.*

Hence, 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 in which a trusted party (called the dealer) computes the shares based on some random values. More precisely, the dealer:

1. selects some values uniformly at random;
2. computes the shares $C_1, \ldots, C_n$ (share C_j is distributed to participant P_j, $1 \leq j \leq n$) based on the previously selected values.

We show in Section 3 that if the previous two steps are performed in the black-box model and the number of the random values is sufficiently large, then a SETUP attack can be mounted against the secret sharing device. An example of ideal secret scheme that satisfy both these conditions is Shamir's [11]:

```
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: set $n$ distinct and public elements $x_1, x_2, \ldots, x_n \in \mathbb{Z}_q$ for the participants ($x_j$ for $P_j$, $1 \leq j \leq n$)
2: pick a $k-1$ degree random polynomial
       $f(x) = a_0 + a_1 x + \cdots + a_{k-1} x^{k-1} \pmod{q}$
   where $a_0 = S$ and $a_j \in \mathbb{Z}_q$, $(1 \leq j \leq k-1)$
3: compute the shares $C_j = f(x_j)$, $1 \leq j \leq n$
```

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:

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

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

3 The SETUP Attack

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

A trivial attack is immediate: in the distribution phase, the attacker receives the (encryption of the) secret instead of a valid share. The honest participants are unable to determine the dishonest behavior if the attacker skips reconstruction. Otherwise, the attacker is not able to provide a valid share and the attack is revealed.

We highlight that this is not the case when the encryption of the secret is a valid share. We use this remark and propose a technique that allows the attacker to reveal the secret with the help of only a few other participants. Except the attacker, any other party needs all the shares of participants belonging to an authorized subset to obtain the secret.

The proposed SETUP attack is 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 generates 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. At least two 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 will consider without loss of generality that the attacker is the first participant P_1. In case of ideal secret sharing, the attacker needs access to at least one more share of another participant, which may be his 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.

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 (the security parameter) 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 suffices, due to the general assumption that shares are distributed through secure channels (Assumption 4).

As extra requirements: (1) any value in $\mathcal{C}^t$ must be a valid ciphertext, i.e. $\forall C \in \mathcal{C}^t$ and pk public key, $\exists M \in \mathcal{S}$ such that $Enc_{pk}(M) = C$; (2) $t \in \mathbb{Z}^*$ is minimum such that the system remains semantically secure; (3) the high speed of the encryption is an advantage.

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

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

As Assumption 6 states, multiple secrets are shared. Each secret is shared during one round. Each 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 $S^1, S^2, \dots$ the secrets to be shared

Output: for each round $i \geq 1$, $C_1^i, C_2^i, \dots, C_n^i$ the shares that correspond to the secret S^i

```
1: if i==1 then
2:     compute $C_1^1, C_2^1, \dots, C_n^1$ accordingly to the genuine secret sharing scheme
```

```
3:     store $H(ID||C_1^1)$ in memory for further usage
4: else
5:     $(C_1^i, \ldots, C_t^i) = Enc_{pk}(S^i \otimes H(ID||C_1^{i-1}))$
6:     replace $H(ID||C_1^{i-1})$ by $H(ID||C_1^i)$ in the non-volatile memory
7:     compute $C_{t+1}^i, C_{t+2}^i, \ldots, C_n^i$ accordingly to the genuine secret sharing scheme
       such that $(C_1^i, \ldots, C_t^i, C_{t+1}^i, \ldots, C_n^i)$ is a valid set of shares
8: end if
```

It is easy to observe that the attacker P_1 can compute the secret S^i $(i > 1)$ if he knows the first t shares of the current round and his own 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: $S^i = Dec_{sk}((C_1^i, \ldots, C_t^i)) \otimes (H(ID||C_1^{i-1}))^{-1}$
```

Hence, P_1 reconstructs S 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.

The most convenient situation is when $P_2, \ldots, P_t$ are fixed and they are allies of P_1. In this scenario, they will always give their shares to the attacker. We highlight that although $P_2, \ldots, P_t$ are allies, they cannot find the secret unless they know the secret key sk. If it is desired that some of the allies have the same advantage in restoring the secret as P_1, then they will be given the secret key.

The attack is practical for small t. Otherwise, it might become difficult to implement or even useless. For example, in case of a (k, n)-threshold scheme with $t \geq k$, 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 $t = 1$ might 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 genuine to achieve the property of indistinguishability, which we consider in the next session.

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 trusted one.

To prove 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 genuine share C_1^1 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 accordingly to the genuine scheme, therefore there are also indistinguishable.

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

Theorem 1. *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.

For $i = 1$ the device runs the genuine sharing algorithm, therefore the indistinguishability property holds by construction.

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 uniformly 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

The SETUP contaminated device must achieve confidentiality against reverse engineering. In this scenario, the content of the non-volatile memory (the public key pk, the random string ID and the hashed value $H(ID||C_1^{i-1})$) is accessible. We highlight that C_1^{i-1} cannot be revealed from the hashed value under the assumption that H is a strong hash function and the private key sk cannot be disclosed from the public key pk under the assumption that the cryptosystem is secure.

We show next that reverse engineering the device brings no advantage: the scheme remains secure for unauthorized set of participants even though they access to the information stored in the non-volatile memory.

Theorem 2. *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, $1 \leq r \leq n$ that gain access to the public key pk, the string ID and the hashed value $H(ID||C_1^{i-1})$ by reverse engineering.

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 case of the genuine scheme, its members are not able to compute any secret S^i, $i \geq 1$. We will analyze next 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 can access 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 single useful information might be the hashed value of the attacker's share. But the share 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 brings them no advantage under the assumption that the 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:

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: set $n$ distinct and public elements $x_1, x_2, \ldots, x_n \in \mathbb{Z}_q$ for the participants
   ($x_j$ for $P_j$, $1 \leq j \leq n$)
2: if i==1 then
3:     pick a $k-1$ degree random polynomial
           $f^1(x) = a_0^1 + a_1^1 x + \ldots. + a_{k-1}^1 x^{k-1} \pmod q$
       where $a_0^1 = S^1$ and $a_j^1 \in \mathbb{Z}_q$ $(1 \leq j \leq k-1)$
4:     compute the shares $C_j^1 = f^1(x_j)$, $1 \leq j \leq n$
5:     store $H(ID||C_1^1)$ in the non-volatile memory
6: else
```

```
 7:    $(C_1^i, C_2^i) = Enc_{pk}(S^i + H(ID||C_1^{i-1}) \pmod q)$
 8:    replace $H(ID||C_1^{i-1})$ by $H(ID||C_1^i)$ in the non-volatile memory
 9:    choose uniformly random $a_j^i \in \mathbb{Z}_q$, $(3 \leq j \leq k-1)$
10:    compute $a_1^i$ and $a_2^i$ as 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:    set the $k-1$ degree polynomial of the round to:
           $f^i(x) = S^i + a_1^i x + \ldots. + a_{k-1}^i x^{k-1} \pmod q$
12:    compute the shares $C_j^i = f^i(x_j)$, $1 \leq j \leq n$
13: end if
```

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

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

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

For completeness, we consider ElGamal cryptosystem as example of a possible public-key encryption system that satisfies the requirements mentioned in Section 3 for $t = 2$ and $\mathcal{S} = \mathcal{C} = \mathbb{Z}_q$ [2].

We give next an experimental evaluation of the attack.

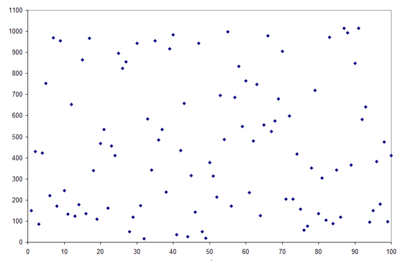
We consider the genuine $(3, 4)$-Shamir threshold secret sharing scheme and its contaminated version. We motivate our choice as follows. First, $k = 3$ is the lowest bound that allows the adversary to recover the secret, while it remains hidden for the authorized subsets (note that for $k = 2$ the SETUP attack is useless since any group of 2 parties - in particular the adversary and his ally - can reconstruct the secret). Second, $n = k + 1$ is minimum for a given k such that the scheme does not degenerate to the particular case of an *all-or-nothing scheme* that requires all the shares for reconstruction.

The rest of the parameters were chosen accordingly to the natural implementation of Shamir's scheme ($x_1 = 1$, $x_2 = 2$, $x_3 = 3$, $x_4 = 4$) and small enough to permit visual inspection ($p = 1019$). We ran both the genuine and contaminated versions for 100 rounds on the same input $S = 0$. We intentionally maintained the same secret for all rounds to show that even this particular case does not cancel output indistinguishability.

The implementation was done in Python, using SHA-224 as hash function (`hashlib.sha224`) and the predefined random selection of integers (`random.randrange`) [12,13,14].

Figure 1 plots the shares of the first two parties for each round of the genuine scheme. Similarly, Figure 2 plots the shares of the attacker and his ally that were output by the contaminated scheme within the same settings. It is easy to observe that the contaminated shares remain distributed uniformly at random.

To conclude, we remark that the practical implementation supports the theory results.


(a) Shares of the party P_1

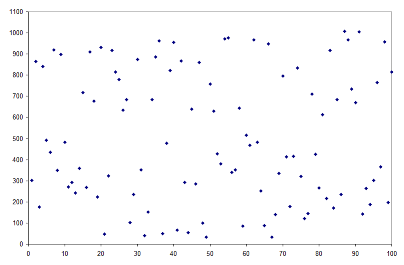
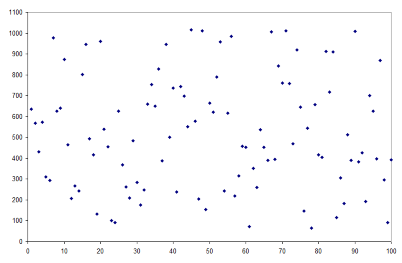

(b) Shares of the party P_2

Fig. 1. Genuine Shamir Secret Sharing Scheme


(a) Shares of the attacker P_1

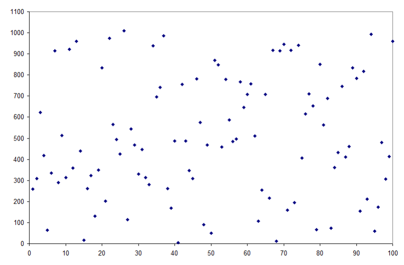

(b) Shares of the attacker's ally P_2

Fig. 2. Contaminated Shamir Secret Sharing Scheme

6 SETUP Attack Prevention Techniques

We showed that, under certain condition, a SETUP attack is possible in secret sharing schemes that use enough random values to compute the shares. Because of the indistinguishability of the outputs of the contaminated device and the genuine one, a user cannot notice its improper behavior and stop using it in practice. More, the majority of the assumptions we have made are easy to meet in practice and therefore they do not restrict the applicability of the proposed attack.

However, we show that a proper usage of the contaminated device significantly lowers its applicability.

For example, the usage of a secret sharing device for sharing multiple secrets is natural in practice. Hence, the restriction that SETUP contaminated version gives no information about the secret shared in the first round is insignificant. It may seem naturally that a restart after each sharing would therefore lead to the futility of the attack. However, this is not always true. The SETUP contaminated device could be designed to maintain the internal value $H(ID||C_1^{i-1})$ even after restart or reset to factory defaults.

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 can 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 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 $\lfloor n/t \rfloor$. 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 technique against the proposed SETUP attack is to randomly map the outputs to the participants for 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.

For completeness, we consider another prevention technique in the same settings (the contaminated device outputs the shares, but doest not distribute them to the parties). We highlight that an attacker uses his previous share to compute the secret. An immediate prevention usage is to input at least one dummy secret between any two genuine secrets to be share and not distribute the dummies shares to the users. This way, the attacker misses his previous share and hence has no advantage to compute the secret. The drawback of this solution is its low efficiency.

7 Conclusions

We introduced SETUP attack against secret sharing schemes that use random values to give the attacker an overwhelming advantage to access the 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 can be enough to reveal the secret. To the best of our knowledge, we are the first to consider such an attack.

We described a general method of 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 vulnerable when the attacker has 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. T. El Gamal,"A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms", Advances in Cryptology, Proceedings of CRYPTO '84, Springer, vol. 196, pp. 10–19, 1984.
3. 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.
4. 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.
5. 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.
6. D. Kucner, M. Kutylowski, "How to use un-trusty cryptographic devices", Tatra Mountains Mathematical Publications, vol.29, pp. 57-67, 2004.
7. 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.

8. 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.
9. E. Mohamed, H. Elkamchouchi, ”Elliptic Curve Kleptography”, International Journal of Computer Science and Network Security, vol.10 (6), pp. 183-185, 2010.
10. 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.
11. A. Shamir, ”How to share a secret”, Communications of the ACM vol.22 (11), pp. 612-613, 1979.
12. Python Programming Language - https://www.python.org/. last accessed: april 22, 2014.
13. Python hashlib - Secure Hashes and Message Digests - https://docs.python.org/2/library/hashlib.html. last accessed: april 22, 2014.
14. Python random - Generate Pseudo-Random Numbers - https://docs.python.org/2/library/random.html. last accessed: april 22, 2014.
15. 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.
16. A. Young, M. Yung, ”Kleptography: Using Cryptography Against Cryptography”, Advances in Cryptology - CRYPTO ’97, Springer-Verlag, pp. 62-74, 1997.
17. 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.
18. A. Young, M. Yung, ”Malicious Cryptography: Kleptographic Aspects”, Proceedings of CT-RSA’2005. pp. 7-18, 2005.
19. Young, A., Yung, M.: Malicious Cryptography: Exposing Cryptovirology, pp. 231, 243-244, Wiley Publishing, 2004.
20. 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.
21. A. Young, M. Yung, ”Kleptography from standard assumptions and application”,Security and Cryptography for Networks, Springer-Verlag, pp. 271-290, 2010.