On the Security Proof of CKO+21 Secret Sharing Scheme*

Yupu Hu[1], Shanshan Zhang[1], Baocang Wang[1], and Siyue Dong[1]

State Key Laboratory of Integrated Services Networks, Xidian University, Xi'an 710071, China
yphu@mail.xidian.edu.cn; sszhang0801@163.com; bcwang@xidian.edu.cn; 359442088@qq.com

Abstract. On CRYPTO2021, Nishanth Chandran, Bhavana Kanukurthi, Sai Lakshmi Bhavana Obattu, and Sruthi Sekar presented a novel secret sharing scheme, called CKO+21 scheme. This scheme makes use of Shamir secret sharing schemes and randomness extractors as its basic components, to generate a multi-layer encapsulation structure. The authors claimed that CKO+21 scheme satisfied "leakage resilience", that is, the privacy still held under both "not enough revealing" and "appropriate leakage". More important is that authors presented a bulky proof for the security of CKO+21 scheme.

In this paper we only consider the simple case of (n, t) threshold secret sharing. We find following 5 facts about CKO+21 scheme, which are the basic reasons we negate the security proof of CKO+21 scheme. (1) In the expression of share of CKO+21 scheme, some bottom Shamir share is simply included, rather than encapsulated. (2) The leakage of the share is not a random leakage, but rather related to the inquiry of the attacker, that is, a chosen leakage. (3) The permitted leakage length of each share is proportional to the share length. (4) The bottom Shamir scheme has such special feature: when the length of the share l^* is kept unchanged, it can make the number of shares n, the threshold value t, and the difference value $n - t + 1$ any large, as long as $t < n$. (5) There is no additional assumption for the bottom Shamir scheme, especially no clear negating its "leakage recoverability" and "contaminated leakage irrecoverability", defined in this paper.

In this paper we point that, CKO+21 scheme didn't successfully prove its security. As long as the bottom Shamir secret sharing scheme satisfies both "leakage recoverability" and "contaminated leakage irrecoverability", the security proof of CKO+21 scheme is wrong. It needs to be pointed out that "leakage recoverability" and "contaminated leakage irrecoverability" cannot be naturally negated by "privacy" of Shamir scheme, and up to now there is not a proof that Shamir scheme doesn't satisfy "leakage recoverability" or "contaminated leakage irrecoverability".

The detailed contribution of this paper is as follow. CKO+21 scheme designed several leakage models: LeakB_0,LeakA_1,LeakB_1,LeakA_2,LeakB_2,$\cdots$,LeakA_h,LeakB_h,LeakC, where

* Supported by National Natural Science Foundations of China (61972457, U19B2021); Key Research and Development Program of Shaanxi (2020ZDLGY08-04); Innovation Scientists and Technicians Troop Construction Projects of Henan Province.

LeakB_0 is the practical leakage model, LeakC is a leakage model independent of the secret message. CKO+21 scheme claimed that an attacker cannot distinguish two adjacent leakage models, so the scheme is "leakage resilient". We point that, if the bottom Shamir scheme satisfies both "leakage recoverability" and "contaminated leakage irrecoverability", the attacker can distinguish LeakB_0 and LeakA_1 with non-negligible probability.

Besides, if the bottom Shamir scheme doesn't satisfy "leakage recoverability". Shamir scheme itself has some ability to resist leakage, and the bulky structure of CKO+21 scheme is not necessary.

When leakage function is extended to general function, the security proof of CKO+21 scheme can be more easily negated. Because CKO+21 scheme didn't clearly restrict the range of leakage function (In fact, leakage function should be restricted within the range of simple functions), this paper chooses a $P/poly$ function as the leakage function, enabling an attacker to distinguish LeakB_0 and LeakA_1 simpler and quicker. Detailedly speaking, under the first explaining of Shamir parameters, the attacker inquires the higher τ bits of a modular p linear function of the bottom Shamir share from each share, then distinguishes LeakB_0 and LeakA_1 simpler and quicker.

Keywords: Secret Sharing (SS) · Random Extractor · Leakage Resilient Secret Sharing (LRSS).

1 Introduction

1.1 Motivation

Secret sharing [1, 2] is such a system that, a secret message is transformed into n shares, with a recovering algorithm, such that (1) correctness: the secret message can be correctly recovered if "revealed shares are enough", and (2) privacy: nothing about the secret message can be obtained if "revealed shares are not enough". A simple (n, t) threshold secret sharing is such: from total n shares, the secret message can be correctly recovered if not less than t shares are revealed, and nothing about secret message can be obtained if not more than $t-1$ shares are revealed. Because of the danger of leakage and tampering, traditional secret sharing schemes are reformed into "leakage resilient secret sharing schemes LRSS " [3–16]. A leakage resilient secret sharing scheme is such that nothing about secret message can be obtained if it satisfies both "revealed shares are not enough" and "appropriate leakage of other shares".

On CRYPTO2021, Nishanth Chandran, Bhavana Kanukurthi, Sai Lakshmi Bhavana Obattu, and Sruthi Sekar presented a novel secret sharing scheme [16], called CKO+21 scheme. This scheme makes use of Shamir secret sharing schemes and randomness extractors as its basic components, to generate a multi-layer encapsulation structure. The authors claimed that CKO+21 scheme satisfied "leakage resilience", that is, the privacy still held under both "not enough revealing" and "appropriate leakage". For the simple case of (n, t) threshold secret sharing, they claimed that nothing about secret message can be obtained if both "not more than $t-1$ shares are revealed" and "other $n-t+1$ shares are appropriately leaked". More important is that authors presented a bulky proof for the security of CKO+21 scheme. For this they designed several leakage models: LeakB_0, LeakA_1, LeakB_1, LeakA_2, LeakB_2, $\cdots$, LeakA_h, LeakB_h, LeakC, where LeakB_0 is the practical leakage model, LeakC is a leakage model independent of the secret message. CKO+21 scheme claimed that an attacker cannot distinguish two adjacent leakage models, that is, an attacker cannot distinguish LeakB_0 and LeakA_1, LeakA_1 and LeakB_1, LeakB_1 and LeakA_2, LeakA_2 and LeakB_2, $\cdots$, LeakA_h and LeakB_h, LeakB_h and LeakC. So the scheme is "leakage resilient".

In this paper we only consider the simple case of (n, t) threshold secret sharing. We find following 5 facts about CKO+21 scheme, which are the basic reasons we negate the security proof of CKO+21 scheme. Five facts are stated as follows.

(1) In the expression of share of CKO+21 scheme, some bottom Shamir share is simply included, rather than encapsulated.

(2) The leakage of the share is not a random leakage, but rather related to the inquiry of the attacker, that is, a chosen leakage. By combining last fact, an attacker can choose to inquire "some bits of Shamir share which is simply included".

(3) The permitted leakage length of each share is proportional to the share length.

(4) The bottom Shamir scheme has such special feature: when the length of the share l^* is kept unchanged, it can make the number of shares n, the threshold value t, and the difference value $n-t+1$ any large, as long as $t < n$. By combining last

fact, the total leakage length the attacker obtains can be far larger than the share length of the bottom Shamir scheme.

(5) There is no additional assumption for the bottom Shamir scheme, especially no clear negating its "leakage recoverability" and "contaminated leakage irrecoverability". "Leakage recoverability" is defined as follow. Suppose an attacker obtains any $t-1$ revealed shares $\{Share^{(1)}, Share^{(2)}, \cdots, Share^{(t-1)}\}$ and the leakage part $\{share^{*(1)}, share^{*(2)}, \cdots, share^{*(n-t+1)}\}$ of other $n-t+1$ shares $\{Share^{*(1)}, Share^{*(2)}, \cdots, Share^{*(n-t+1)}\}$. Then there is an efficient algorithm to recover the secret message with non-negligible probability. Under the condition that "leakage recoverability" is true, "contaminated leakage irrecoverability" is defined as follow. Suppose the attacker obtains any $t-1$ revealed shares $\{Share^{(1)}, Share^{(2)}, \cdots, Share^{(t-1)}\}$ and the leakage part of other shares $\{Share^{*(1)}, Share^{*(2)}, \cdots, Share^{*(n-t+1)}\}$, but from $\{Share^{*(1)}, Share^{*(2)}, \cdots, Share^{*(n-t+1)}\}$ there are $t-1$ shares which come from another secret message (That is, the leakage is contaminated leakage. Of course we should assume $t-1 \leq n-t+1$). Then the probability with which the attacker uses above mentioned efficient algorithm to output a "secret message" is negligible. The reasonability of "leakage recoverability" can be obtained by considering last fact, where we can make the leakage part $\{share^{*(1)}, share^{*(2)}, \cdots, share^{*(n-t+1)}\}$ has total length far larger than the length of a single share. "Contaminated leakage irrecoverability" can also be easily understood, for over large invalid data may produce incompatibility, leading the originally efficient algorithm into a forced stop or an infinite loop.

1.2 Contribution

In this paper we point that CKO+21 scheme didn't successfully prove its security, that is, its proof about "leakage resilience" has a loophole. As long as the bottom Shamir secret sharing scheme satisfies both "leakage recoverability" and "contaminated leakage irrecoverability", the security proof of CKO+21 scheme is wrong. It needs to be pointed out that "leakage recoverability" and "contaminated leakage irrecoverability" cannot be naturally negated by "privacy" of Shamir scheme, and

up to now there is not a proof that Shamir scheme doesn't satisfy "leakage recoverability", or although it does satisfy "leakage recoverability" but not "contaminated leakage irrecoverability".

Detailed contribution of this paper is as follow. We point that, if the bottom Shamir scheme satisfies both "leakage recoverability" and "contaminated leakage irrecoverability", an attacker can distinguish LeakB_0 and LeakA_1 with non-negligible probability.

This paper doesn't claim that CKO+21 scheme is not secure, but only negates its security proof. This paper doesn't construct detailed algorithm to realize "leakage recoverability" and "contaminated leakage irrecoverability" either, but only shows that no one said such algorithm doesn't exist, and that a necessary condition of CKO+21 security proof is to prove such algorithm doesn't exist. An interesting question is whether such security proof can be revised. Besides, if bottom Shamir scheme doesn't satisfy "leakage recoverability", Shamir scheme itself has some ability to resist leakage, and bulky structure of CKO+21 scheme is not necessary.

Another contribution of this paper is that, when leakage function is extended to general function, we present a simpler method to negate the security proof of CKO+21 scheme. In engineering practice, leakage function should be restricted within the range of simple functions, for example, directly leaking a bit, or leaking exclusive-or of two bits. Generally, complicated leakage function is not practical. However, CKO+21 scheme [16] didn't clearly restrict the range of leakage function. It is a chance for us to discuss complicated leakage function. This paper chooses a $P/poly$ function as the leakage function, enabling an attacker to distinguish LeakB_0 and LeakA_1 simpler and quicker. Detailedly speaking, under the first explaining of Shamir parameters, the attacker inquires the higher τ bits of a modular p linear function of the bottom Shamir share from each share, then distinguishes LeakB_0 and LeakA_1 simpler and quicker.

1.3 Organization

Section 2 is preliminary, stating secret sharing, leakage, leakage recoverability and contaminated leakage irrecoverability, Shamir scheme. In this section an emphasis is the analysis of leakage recoverability of Shamir scheme. Section 3 states CKO+21 secret sharing scheme [16], including bottom components, important parameter, shares generation, secret message recovering, and security proof. Because the result of this paper only has relation with two leakage models LeakB_0 and LeakA_1, this section only states such two leakage models in detail. Other leakage models for the security proof of CKO+21 scheme can be found in subsection 4.3 of [16]. Section 4 presents the conclusion of this paper, that is, for some specific leakage inquiring and "leakage recoverability/contaminated leakage irrecoverability of the bottom Shamir scheme", an attacker can distinguish LeakB_0 and LeakA_1 with non-negligible probability. Section 5 is negating the security proof of CKO+21 scheme for leakage functions which are extended to general functions. This section also shows that the attacker can distinguish LeakB_0 and LeakA_1, but the negating procedure is simpler and quicker.

2 Preliminary

2.1 Secret Sharing

Definition 1. *(Partially ordered set class) Suppose the set* $\{1, 2, \cdots, n\}$ *has a subset class* $\mathcal{A}$*. If for any* $A \in \mathcal{A}$*, any* $B \supset A$*,we always have* $B \in \mathcal{A}$*, then* $\mathcal{A}$ *is called a partially ordered set class of* $\{1, 2, \cdots, n\}$*.*

(Following Definition 2 and Definition 3 simply state secret sharing, which are enough for the reasoning of this paper. Formal definition needs more strict description of "indistinguishability".)

Definition 2. *(*$(n, \mathcal{A})$ *secret sharing scheme) Suppose* $\mathcal{A}$ *is a partially ordered set class of* $\{1, 2, \cdots, n\}$*. A scheme is called an* $(n, \mathcal{A})$ *secret sharing scheme, if any secret message* m *can be transformed into* n *shares* $\{Share_1, Share_2, \cdots, Share_n\}$*,*

such that (1) correctness: for any $A \in \mathcal{A}$, there is an efficient algorithm to recover m by $\{Share_i, i \in A\}$; (2) privacy: for any $A \notin \mathcal{A}$, nothing about m can be obtained by $\{Share_i, i \in A\}$.

Definition 3. *((n,t) threshold secret sharing scheme) A scheme is called an (n,t) threshold secret sharing scheme, if any secret message m can be transformed into n shares $\{Share_1, Share_2, \cdots, Share_n\}$, such that (1) correctness: by any not less than t shares, there is an efficient algorithm to recover m; (2) privacy: by any not more than $t-1$ shares, nothing about m can be obtained.*

Definition 4. *(The rate of a secret sharing scheme) The rate of a secret sharing scheme is defined as l/l^*, where l is the bit length of the secret message, l^* is the bit length of the longest share.*

2.2 Leakage, leakage rate, leakage recoverability, contaminated leakage irrecoverability

From now on we only consider (n,t) threshold secret sharing scheme. Suppose some $t-1$ shares belong to malice users, and they try to obtain leakage of other shares, for the purpose of recovering the secret message.

Definition 5. *(Leakage function and leakage rate) Leakage function of the share value Share is defined as $f(Share)$. Leakage rate is defined as τ/l^*, where τ is the bit length of the leakage function value, l^* is the bit length of the longest share.*

(Following Definition 6 simply states leakage recoverability and contaminated leakage irrecoverability, which is enough for the reasoning of this paper. Formal definition needs more strict description of "difficulty".)

Definition 6. *(Leakage recoverability and contaminated leakage irrecoverability) Suppose the attacker owns any $t-1$ shares $\{Share^{(1)}, Share^{(2)}, \cdots, Share^{(t-1)}\}$ and leakage $\{share^{*(1)}, share^{*(2)}, \cdots, share^{*(n-t+1)}\}$ of other $n-t+1$ shares $\{Share^{*(1)}, Share^{*(2)}, \cdots, Share^{*(n-t+1)}\}$.*

(1) If there is an efficient algorithm that recovers the secret message with non-negligible probability, we say that the scheme satisfies leakage recoverability.

(2) Suppose the scheme satisfies leakage recoverability, and from $\{Share^{*(1)}, Share^{*(2)}, \cdots, Share^{*(n-t+1)}\}$ *there are* $t-1$ *shares which come from another secret message (That is, the leakage is contaminated leakage. Of course we should assume* $t-1 \leq n-t+1$*). If the attacker uses above mentioned efficient algorithm to output a "secret message" with negligible probability, we say that the scheme satisfies contaminated leakage irrecoverability.*

2.3 The first secret sharing scheme——Shamir (n, t) threshold secret sharing scheme [1]

The system chooses a prime number p (p should be exponentially large). The secret message m is a number from the field $\mathbb{GF}(p)$. Randomly choose $t-1$ numbers $\{c_1, c_2, \cdots, c_{t-1}\}$ from $\mathbb{GF}(p)$, to obtain a mask polynomial $P(x)$ of m, $P(x) = m + c_1 x + c_2 x^2 + \cdots + c_{t-1} x^{t-1} \,(\mathrm{mod} p)$. Randomly choose n numbers $\{x_1, x_2, \cdots, x_n\}$ from $\mathbb{GF}(p)$, and compute $\{P(x_1), P(x_2), \cdots, P(x_n)\}$. Then n users respectively own $\{(x_1, P(x_1)), (x_2, P(x_2)), \cdots, (x_n, P(x_n))\}$. It is clear that, any t users can obtain $\{m, c_1, c_2, \cdots, c_{t-1}\}$ by solving the linear equations group over $\mathbb{GF}(p)$; any $t-1$ users can only obtain a linear equation of $\{m, c_1\}$ over $\mathbb{GF}(p)$; any $t-2$ users can only obtain a linear equation of $\{m, c_1, c_2\}$ over $\mathbb{GF}(p)$; any $t-3$ users can only obtain a linear equation of $\{m, c_1, c_2, c_3\}$ over $\mathbb{GF}(p)$; $\cdots$; any 1 user can only obtain a linear equation of $\{m, c_1, c_2, \cdots, c_{t-1}\}$ over $\mathbb{GF}(p)$. Because Shamir scheme didn't clearly illustrate which is the share, there are two different explainings on parameter setting of Shamir scheme.

The first explaining: the system public parameters are $\{p, n, t, x_1, x_2, \cdots, x_n\}$, and n users respectively own the shares $\{P(x_1), P(x_2), \cdots, P(x_n)\}$. In this case secret message and share have equal length, $l = l^* = \lceil \log p \rceil$, and the rate is $l/l^* = 1$.

The second explaining: the system public parameters are $\{p, n, t\}$, and n users respectively own the shares $\{(x_1, P(x_1)), (x_2, P(x_2)), \cdots, (x_n, P(x_n))\}$. In this case secret message has the length $l = \lceil \log p \rceil$, the share has the length $l^* = 2\lceil \log p \rceil$, and the rate is $l/l^* = 1/2$.

It is easy to understand about Shamir scheme that, when the length of the share l^* is kept unchanged, it can make the number of shares n, the threshold value t, and the difference value $n-t+1$ any large, as long as $t < n$.

2.4 Leakage recoverability of Shamir scheme

Consider the first explaining. When an attacker obtains $t-1$ revealed shares, he obtains a linear equation of $\{m, c_1\}$ over $\mathbb{GF}(p)$: $a_0 m + a_1 c_1 + b \,(\mathrm{mod} p) = 0$, where $\{a_0, a_1, b\}$ are known by the attacker. Now suppose the attacker obtains leakage of other $n-t+1$ shares, the permitted leakage length of each share is $\tau(\tau < l^*)$. The attacker cannot obtain new (linear or nonlinear) equation of $\{m, c_1\}$ over $\mathbb{GF}(p)$, because each share is not totally leaked. However, the attacker obtains $(n-t+1)\,\tau$ new nonlinear equations of $\{m, c_1\}$ over $\mathbb{GF}(2)$. According to former state, we can assume $(n-t+1)\,\tau$ is far larger than l^*. Our question is that, is there an efficient algorithm to obtain $\{m, c_1\}$ with non-negligible probability, by a linear equation of $\{m, c_1\}$ over $\mathbb{GF}(p)$ and $(n-t+1)\,\tau$ nonlinear equations of $\{m, c_1\}$ over $\mathbb{GF}(2)$? Such question seems no way to answer.

Then we consider the second explaining. When the attacker obtains $t-1$ revealed shares, he obtains a linear equation of $\{m, c_1\}$ over $\mathbb{GF}(p)$. Suppose the attacker obtains leakage of other $n-t+1$ shares, then he obtains $(n-t+1)\,\tau$ new nonlinear equations of $\{m, c_1, x_1, x_2, \cdots, x_n\}$ over $\mathbb{GF}(2)$. Our another question is that, is there an efficient algorithm to obtain m with non-negligible probability, by a linear equation of $\{m, c_1\}$ over $\mathbb{GF}(p)$ and $(n-t+1)\,\tau$ nonlinear equations of $\{m, c_1, x_1, x_2, \cdots, x_n\}$ over $\mathbb{GF}(2)$? Although such solving task seems much more difficult than last one, there is no way to say it is a hard task.

More important is that, up to now there is no one to answer these two questions. In other words, leakage recoverability of Shamir scheme has never been negated.

3 CKO+21 secret sharing scheme

3.1 A bottom component: randomness extractor

Randomness extractor $\mathsf{Ext}(w,s)$ is a mechanism to generate random bit string, where w is a public parameter called source, s is a secret parameter called seed. The functionality of randomness extractor is similar to stream cipher, for which (w,s) has fixed length while the output $\mathsf{Ext}(w,s)$ has any length. CKO+21 scheme uses h randomness extractors $\{\mathsf{Ext}^1(.,.),\mathsf{Ext}^2(.,.),\cdots,\mathsf{Ext}^h(.,.)\}$, with decreasing output length (without losing generality, we can simply understand in such way: the length of $\mathsf{Ext}^{h-1}(.,.)$ is double of $\mathsf{Ext}^h(.,.)$, the length of $\mathsf{Ext}^{h-2}(.,.)$ is triple of $\mathsf{Ext}^h(.,.)$,$\cdots$, the length of $\mathsf{Ext}^1(.,.)$ is h times of $\mathsf{Ext}^h(.,.)$. h randomness extractors respectively use seeds $\{s^1,s^2,\cdots,s^h\}$, that is, CKO+21 scheme uses $\{\mathsf{Ext}^1(.,s^1),\mathsf{Ext}^2(.,s^2),\cdots,\mathsf{Ext}^h\left(.,s^h\right)\}$. These randomness extractors satisfy some special properties, called "adaptive randomness extractors". Because the work of this paper has no relation with randomness extractor, we don't restate these special properties.

3.2 Another bottom component: Shamir secret sharing scheme

CKO+21 scheme repeatedly uses Shamir secret sharing scheme $h+1$ times, respectively denoted as$\{\{\mathsf{MShare}(.),\mathsf{MRec}(.)\}$, $\{\mathsf{SdShare}^1(.),\mathsf{SdRec}^1(.)\}$, $\{\mathsf{SdShare}^2(.),$ $\mathsf{SdRec}^2(.)\}$, $\cdots$, $\left\{\mathsf{SdShare}^h(.),\mathsf{SdRec}^h(.)\right\}\}$, where MShare generates shares of secret message m, MRec recovers secret message m, $\mathsf{SdShare}^i$ generates shares of the seed s^i for randomness extractor Ext^i, SdRec^i recovers the seed s^i for randomness extractor Ext^i, $i\in\{1,2,\cdots,h\}$.

3.3 About the important parameter h

The meaning of h is such: CKO+21 scheme is an h-layer encapsulation structure, where $h=\left\lceil\frac{n-t+1}{t-1}\right\rceil$. Roughly speaking, $h(t-1)\approx n-t+1$. For a fixed t, the larger the number of users n, the larger the number of encapsulation layers h.

3.4 Generating shares of CKO+21 secret sharing scheme[[16], subsection 4.2, Fig.3]

Suppose a secret message is m. Then the share generation procedure is composed of the following 6 steps.

Step 1: Compute $(m_1, \cdots, m_n) \leftarrow \mathsf{MShare}(m)$;

Step 2: For each $i \in \{1, 2, \cdots, h\}$, choose random seed s^i, and compute $(sd_1^i, \cdots, sd_n^i) \leftarrow \mathsf{SdShare}^i(s^i)$;

Step 3: For each $i \in \{1, 2, \cdots, h\}$, $j \in \{1, 2, \cdots, n\}$, choose random source w_j^i;

Step 4: For each $j \in \{1, 2, \cdots, n\}$,

(4.1)Define $y_j^{h+1} = m_j$;

(4.2) For each $i \leftarrow h$ to 1, compute $x_j^i = y_j^{i+1} \oplus \mathsf{Ext}^i\left(w_j^i, s^i\right)$, $y_j^i = \left(x_j^i, sd_j^i\right)$;

Step 5: For each $j \in \{1, 2, \cdots, n\}$, define $Sh_j = (w_j^1, \cdots, w_j^h, y_j^1) = (w_j^1, \cdots, w_j^h, x_j^1, sd_j^1)$;

Step 6: Output n shares $\{Sh_1, \cdots, Sh_n\}$.

Another description: we can describe the share generation procedure as an h-layer encapsulation, as following Fig.1.

Encapsulation initialization: $y_j^{h+1} = m_j$;

The first layer encapsulation: $(w_j^h, y_j^{h+1} \oplus \mathsf{Ext}^h\left(w_j^h, s^h\right), sd_j^h) = (w_j^h, y_j^h)$;

The second layer encapsulation: $(w_j^{h-1}, w_j^h, y_j^h \oplus \mathsf{Ext}^{h-1}\left(w_j^{h-1}, s^{h-1}\right), sd_j^{h-1}) = (w_j^{h-1}, w_j^h, y_j^{h-1})$;

$\cdots$;

The k'th layer encapsulation: $(w_j^{h-k+1}, \cdots, w_j^h, y_j^{h-k+2} \oplus \mathsf{Ext}^{h-k+1}\left(w_j^{h-k+1}, s^{h-k+1}\right), sd_j^{h-k+1}) = (w_j^{h-k+1}, \cdots, w_j^h, y_j^{h-k+1})$;

$\cdots$;

The h'th layer encapsulation: $(w_j^1, \cdots, w_j^h, y_j^2 \oplus \mathsf{Ext}^1\left(w_j^1, s^1\right), sd_j^1) = (w_j^1, \cdots, w_j^h, y_j^1)$.

Fig. 1. h-layer encapsulation

We have two notes. Note 1: In step 4 it is an h-layer encapsulation, each layer encapsulation makes share length longer (without losing generality, we can simply understand in such way: the length of y_j^h is double of y_j^{h+1}, the length of y_j^{h-1} is triple of y_j^{h+1}, $\cdots$, the length of y_j^1 is $h+1$ times of y_j^{h+1}). Note 2: in the final generated share expression, the bottom Shamir share sd_j^1 is simply included, rather

than encapsulated. In other words, when the attacker chooses leakage target, he can choose to inquire some bits of sd_j^1.

3.5 Recovering secret message of CKO+21 secret sharing scheme[[16], subsection 4.2, Fig.3]

Suppose t shares are obtained. Without losing generality, suppose $\{Sh_1, \cdots, Sh_t\}$ are obtained, where $Sh_j = (w_j^1, \cdots, w_j^h, x_j^1, sd_j^1)$.

Step 1: For each $i \leftarrow 1$ to h, compute

(1.1) $s^i \leftarrow \mathsf{SdRec}^i(sd_1^i, \cdots, sd_t^i)$.

(1.2) For each $j \in \{1, 2, \cdots, t\}$, $y_j^{i+1} \leftarrow x_j^i \oplus \mathsf{Ext}^i\left(w_j^i, s^i\right)$. If $i \leq h-1$, denote $y_j^{i+1} = \left(x_j^{i+1}, sd_j^{i+1}\right)$; if $i = h$, denote $y_j^{i+1} = m_j$.

Step 2: Compute $m \leftarrow \mathsf{MRec}(m_1, \cdots, m_t)$.

Step 3: Output the secret message m.

3.6 Leakage model $\mathsf{LeakB_0}$, regulated by CKO+21 scheme[[16], subsection 4.1, Fig.2]

First, an attacker makes leakage inquiring to $n-t+1$ shares, that is, for each share $Share$ he chooses a leakage function f, inquires the value of $f(Share)$, then he records {serial number of $Share$, f, $f(Share)$}. Second, the attacker asks the remaining $t-1$ shares to be revealed, then he records each {serial number of $Share$, $Share$}. Now the total information the attacker owns is {serial number of $Share$, $Share$} of $t-1$ shares and {serial number of $Share$, f, $f(Share)$} of other $n-t+1$ shares. For example, the total information the attacker owns is $\{\{1, Sh_1\}, \cdots, \{t-1, Sh_{t-1}\}\}$ and $\{\{t, f_t, f_t(Sh_t)\}, \cdots, \{n, f_n, f_n(Sh_n)\}\}$.

We have two notes. Note 1: leakage function $f : \{0,1\}^{l^*} \to \{0,1\}^{\tau}$, where l^* is the length of the share, τ is the permitted length of leakage for each share. CKO+21 scheme regulates leakage rate τ/l^* to be a constant, that is, leakage rate τ/l^* has no relation with $\{n, t\}$. Note 2: leakage function f is not randomly sampled, but "adaptively chosen" by the attacker.

3.7 Security Proof of CKO+21 secret sharing scheme[[16], subsection 4.3]

Authors of CKO+21 scheme presented a bulky proof for the security of CKO+21 scheme. For this they designed several leakage models: LeakB_0, LeakA_1, LeakB_1, LeakA_2, LeakB_2, $\cdots$, LeakA_h, LeakB_h, LeakC, where LeakB_0 is the practical leakage model (that is, the leakage model stated in our subsection 3.6), LeakC is a leakage model independent of the secret message. CKO+21 scheme claimed that the attacker cannot distinguish two adjacent leakage models, that is:

An attacker cannot distinguish LeakB_0 and LeakA_1 (LeakB_0 and LeakA_1 are very alike);

An attacker cannot distinguish LeakA_1 and LeakB_1 (LeakA_1 and LeakB_1 are very alike);

An attacker cannot distinguish LeakB_1 and LeakA_2 (LeakB_1 and LeakA_2 are very alike);

An attacker cannot distinguish LeakA_2 and LeakB_2 (LeakA_2 and LeakB_2 are very alike);

$\cdots$;

An attacker cannot distinguish LeakA_h and LeakB_h (LeakA_h and LeakB_h are very alike);

An attacker cannot distinguish LeakB_h and LeakC (LeakB_h and LeakC are very alike).

The final conclusion is that an attacker cannot distinguish LeakB_0 and LeakC (LeakB_0 and LeakC are very alike). So the scheme is "leakage resilient".

In [16], subsection 4.3, Fig.4 and Fig.5, detailed description of { LeakB_0, LeakA_1, LeakB_1, LeakA_2, LeakB_2, $\cdots$, LeakA_h, LeakB_h, LeakC } are presented. Because this paper only shows that LeakB_0 and LeakA_1 are distinguishable, and LeakB_0 has been stated in our subsection 3.6, in the follow we only state LeakA_1.

For CKO+21 scheme, suppose n shares $\{Sh_1, \cdots, Sh_n\}$ are generated by secret message m, sources $\{w_j^i, i \in \{1, 2, \cdots, h\}, j \in \{1, 2, \cdots, n\}\}$, and seeds $\{s^1, s^2, \cdots, s^h\}$. Suppose other n shares $\{Sh_1^*, \cdots, Sh_n^*\}$ are generated by secret message m, sources

$\left\{w_j^i, i \in \{1,2,\cdots,h\}, j \in \{1,2,\cdots,n\}\right\}$, and seeds $\{s^{*1}, s^2, \cdots, s^h\}$, where s^{*1} is an all-0 vector. That is, the unique difference of $\{Sh_1^*, \cdots, Sh_n^*\}$ and $\{Sh_1, \cdots, Sh_n\}$ is the difference of s^{*1} and s^1. For $t-1$ serial numbers, the attacker obtains {serial number of $Share$, $Share$} of corresponding $t-1$ shares of $\{Sh_1, \cdots, Sh_n\}$; for other $n-2t+2$ serial numbers, the attacker obtains {serial number of $Share$, f, $f(Share)$} of corresponding $n-2t+2$ shares of $\{Sh_1, \cdots, Sh_n\}$; for the remaining $t-1$ serial numbers, the attacker obtains {serial number of $Share^*$, f^*, $f^*(Share^*)$} of corresponding $t-1$ shares of $\{Sh_1^*, \cdots, Sh_n^*\}$. However, the attacker doesn't know which serial numbers are from $\{Sh_1^*, \cdots, Sh_n^*\}$. For example, the attacker obtains $\{\{1, Sh_1\},\cdots,\{t-1, Sh_{t-1}\},\{t, f_t, f_t(Sh_t)\},\cdots,\{n-t+1, f_{n-t+1}, f_{n-t+1}(Sh_{n-t+1})\},\{n-t+2, f_{n-t+2}^*, f_{n-t+2}^*\left(Sh_{n-t+2}^*\right)\},\cdots,\{n, f_n^*, f_n^*(Sh_n^*)\}\}$, but he doesn't know that final $t-1$ terms are from $\{Sh_1^*, \cdots, Sh_n^*\}$ rather than $\{Sh_1, \cdots, Sh_n\}$. Such leakage model is called LeakA_1.

We have a note here: CKO+21 scheme didn't regulate "the attack doesn't know which serial numbers are from the contaminated leakage". In fact, if we assume an attacker of LeakA_1 does know the serial numbers of the contaminated leakage, it is easier to distinguish LeakB_0 and LeakA_1.

4 Distinguishability of LeakB_0 and LeakA_1

4.1 Chosen leakage: leaking the right side τ bits of each share

Because the leakage is a chosen leakage rather than a random leakage, the attacker can inquire the right side τ bits of each share. On the other hand, the right side of each share is just the right side of bottom share of the seed s^1 (or the seed s^{*1}), so that the leakage is just the right side τ bits of bottom share of the seed s^1 (or the seed s^{*1}).

4.2 Distinguishing LeakB_0 and LeakA_1

Now suppose the bottom Shamir secret sharing scheme $\{\mathsf{SdShare}^1(.), \mathsf{SdRec}^1(.)\}$ satisfies both "leakage recoverability" and "contaminated leakage irrecoverability",

and the leakage of each share is just the leakage of the bottom share of the seed s^1 (or the leakage of the bottom share of the seed s^{*1}). For LeakB_0, the attacker uses the efficient algorithm to obtain an output with non-negligible probability. For LeakA_1, the attacker uses same efficient algorithm to obtain an output with negligible probability.

4.3 More notes

Note 1: The conclusion of this paper has no relation with randomness extractor, that is, excellent randomness extractor cannot resist our conclusion.

Note 2: This paper doesn't show the insecurity of CKO+21 scheme, but only negates its security proof. In fact, "leakage recoverability" and "contaminated leakage irrecoverability" are not enough to negate the security of CKO+21 scheme, but only open the outside layer encapsulation.

Note 3: This paper doesn't construct detailed algorithm to realize "leakage recoverability" and "contaminated leakage irrecoverability" either, but only shows that no one said such algorithm doesn't exist, and that a necessary condition of CKO+21 security proof is to prove such algorithm doesn't exist.

Note 4: An interesting question is whether such security proof can be revised.

Note 5: If the bottom Shamir scheme doesn't satisfy "leakage recoverability", Shamir scheme itself has some ability to resist leakage, and bulky structure of CKO+21 scheme is not necessary.

5 Negating the security proof of CKO+21 scheme for general leakage function

5.1 A leakage feature of Shamir scheme under the first explaining

In this section we take the first explaining on parameter setting of Shamir scheme. Then for any $j \in \{t, t+1, \cdots, n\}$, matrix

$$\begin{bmatrix} 1 & x_1 & x_1^2 & \cdots & x_1^{t-1} \\ 1 & x_2 & x_2^2 & \cdots & x_2^{t-1} \\ \cdots & & & & \\ 1 & x_{t-1} & x_{t-1}^2 & \cdots & x_{t-1}^{t-1} \\ 1 & x_j & x_j^2 & \cdots & x_j^{t-1} \end{bmatrix} (\mathrm{mod} p)$$

is the system public parameter of Shamir scheme, so the attacker knows the reverse matrix

$$\begin{bmatrix} u_{11}^{(j)} \cdots u_{1t}^{(j)} \\ \vdots \quad\quad \vdots \\ u_{t1}^{(j)} \cdots u_{tt}^{(j)} \end{bmatrix} = \begin{bmatrix} 1 & x_1 & x_1^2 & \cdots & x_1^{t-1} \\ 1 & x_2 & x_2^2 & \cdots & x_2^{t-1} \\ \cdots & & & & \\ 1 & x_{t-1} & x_{t-1}^2 & \cdots & x_{t-1}^{t-1} \\ 1 & x_j & x_j^2 & \cdots & x_j^{t-1} \end{bmatrix}^{-1} (\mathrm{mod} p),$$

The attacker also knows the following relation

$$\begin{bmatrix} u_{11}^{(j)} \cdots u_{1t}^{(j)} \\ \vdots \quad\quad \vdots \\ u_{t1}^{(j)} \cdots u_{tt}^{(j)} \end{bmatrix} \begin{bmatrix} P(x_1) \\ \vdots \\ P(x_{t-1}) \\ P(x_j) \end{bmatrix} (\mathrm{mod} p) = \begin{bmatrix} m \\ c_1 \\ \vdots \\ c_{t-1} \end{bmatrix},$$

where m is the secret message, $c_1 \sim c_{t-1}$ are other coefficients of the mask polynomial $P(x)$.

Suppose an attacker obtains shares $\{P(x_1), P(x_2), \cdots, P(x_{t-1})\}$, and, by leakage inquiring, obtains the higher τ bits of the modular p linear function $u_{1t}^{(j)} \cdot P(x_j)(\mathrm{mod}\, p)$ of the share $P(x_j)$ (Notice that, from these τ bits, each bit is a $P/poly$ function of $P(x_j)$ over the field $\mathbb{GF}(2)$, rather than a simple function). Take these

higher τ bits and set the lower bits all 0, then we obtain a number over $\mathbb{GF}(p)$, denoted as v_j. Then the attacker computes

$$u_{11}^{(j)} \cdot P(x_1) + \cdots + u_{1,t-1}^{(j)} \cdot P(x_{t-1}) + v_j (\mathrm{mod} p).$$

It is easy to see that the top bit of the computing result equals to the top bit of

$$u_{11}^{(j)} \cdot P(x_1) + \cdots + u_{1,t-1}^{(j)} \cdot P(x_{t-1}) + u_{1t}^{(j)} \cdot P(x_j) (\mathrm{mod} p)$$

with the probability $1 - 2^{-\tau}$. In other words, the top bit of the above computing result equals to the top bit of m with the probability $1 - 2^{-\tau}$.

For $j \in \{t, t+1, \cdots, n\}$, the attacker obtains $n - t + 1$ computing results, and top bits of these computing results take one value (that is, the top bit of m) with proportion $1 - 2^{-\tau}$, another value with proportion $2^{-\tau}$.

5.2 A contaminated leakage feature of Shamir scheme under the first explaining

Still suppose an attacker obtains $\{P(x_1), P(x_2), \cdots, P(x_{t-1})\}$ and respective higher τ bits of $\{u_{1t}^{(j)} \cdot P(x_j)(\mathrm{mod}\, p), j \in \{t, t+1, \cdots, n\}\}$, but from the corresponding $\{P(x_j), j \in \{t, t+1, \cdots, n\}\}$ there are $t-1$ terms which are shares of another secret message.

Then the attacker takes the operations described in subsection 5.1, to obtain $n-t+1$ top bits of the computing results. It is easy to see that $t-1$ bits are completely random, and other $n-2t+2$ bits take one value (that is, the top bit of m) with proportion $1-2^{-\tau}$, another value with proportion $2^{-\tau}$. From all of the above, the top bits of the computing results take one value (that is, the top bit of m) with proportion $(1-2^{-\tau}) \cdot \frac{n-2t+2}{n-t+1} + \frac{1}{2}\frac{t-1}{n-t+1}$, another value with proportion $2^{-\tau} \cdot \frac{n-2t+2}{n-t+1} + \frac{1}{2}\frac{t-1}{n-t+1}$. Because $\{1-2^{-\tau}, 2^{-\tau}\}$ and $\left\{(1-2^{-\tau}) \cdot \frac{n-2t+2}{n-t+1} + \frac{1}{2}\frac{t-1}{n-t+1}, 2^{-\tau} \cdot \frac{n-2t+2}{n-t+1} + \frac{1}{2}\frac{t-1}{n-t+1}\right\}$ have clear difference, the attacker can easily check whether the leakage is a contaminated leakage.

5.3 Distinguishing $\mathsf{LeakB_0}$ and $\mathsf{LeakA_1}$

An attacker obtains $t-1$ shares. He also obtains, by leakage inquiring, respective higher τ bits of modular p linear functions of the bottom Shamir shares of seed s^1

of other $n-t+1$ shares, as described in subsections 5.1∼5.2. It is easy to see that LeakB_0 and LeakA_1 can be immediately distinguished by distributions $\{1-2^{-\tau}, 2^{-\tau}\}$ and $\left\{(1-2^{-\tau})\cdot\frac{n-2t+2}{n-t+1}+\frac{1}{2}\frac{t-1}{n-t+1}, 2^{-\tau}\cdot\frac{n-2t+2}{n-t+1}+\frac{1}{2}\frac{t-1}{n-t+1}\right\}$.

References

1. Shamir, A.: How to share a secret. Commun. ACM 22, 612–613 (1979) https://doi.org/10.1145/359168.359176
2. Benaloh, J., Leichter, J.: Generalized secret sharing and monotone functions. In: Goldwasser, S. (ed.) CRYPTO 1988. LNCS, vol. 403, pp. 27–35. Springer, New York (1990). https://doi.org/10.1007/0-387-34799-2_3
3. Aggarwal, D., et al.: Stronger leakage-resilient and non-malleable secret sharing schemes for general access structures. In: Boldyreva, A., Micciancio, D. (eds.) CRYPTO 2019. LNCS, vol. 11693, pp. 510–539. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-26951-7_18
4. Badrinarayanan, S., Srinivasan, A.: Revisiting non-malleable secret sharing. In: Ishai, Y., Rijmen, V. (eds.) EUROCRYPT 2019, Part I. LNCS, vol. 11476, pp. 593–622. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-17653-2_20
5. Bellare, M., Rogaway, P.: Robust computational secret sharing and a unified account of classical secret-sharing goals. In: Proceedings of the 14th ACM Conference on Computer and Communications Security. CCS 2007, Association for Computing Machinery (2007). https://doi.org/10.1145/1315245.1315268
6. Benhamouda, F., Degwekar, A., Ishai, Y., Rabin, T.: On the local leakage resilience of linear secret sharing schemes. In: Shacham, H., Boldyreva, A. (eds.) CRYPTO 2018. LNCS, vol. 10991, pp. 531–561. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-96884-1_18
7. Brian, G., Faonio, A., Obremski, M., Simkin, M., Venturi, D.: Non-malleable secret sharing against bounded joint-tampering attacks in the plain model. In: Micciancio, D., Ristenpart, T. (eds.) CRYPTO 2020. LNCS, vol. 12172, pp. 127–155. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-56877-1_5
8. Brian, G., Faonio, A., Venturi, D.: Continuously non-malleable secret sharing for general access structures. In: Hofheinz, D., Rosen, A. (eds.) TCC 2019. LNCS, vol. 11892, pp. 211–232. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-36033-7_8
9. Chattopadhyay, E., et al.: Extractors and secret sharing against bounded collusion protocols. In: 61st IEEE Annual Symposium on Foundations of Computer Science, FOCS 2020 (2020). https://doi.org/10.1109/FOCS46700.2020.00117
10. Dziembowski, S., Pietrzak, K.: Intrusion-resilient secret sharing. In: Proceedings of the 48th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2007 (2007). https://doi.org/10.1109/FOCS.2007.35
11. Faonio, A., Venturi, D.: Non-malleable secret sharing in the computational setting: adaptive tampering, noisy-leakage resilience, and improved rate. In: Boldyreva, A., Micciancio, D. (eds.) CRYPTO 2019.

LNCS, vol. 11693, pp. 448–479. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-26951-7_16

12. Goyal, V., Kumar, A.: Non-malleable secret sharing. In: Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2018 (2018). https://doi.org/10.1145/3188745.3188872
13. Kumar, A., Meka, R., Sahai, A.: Leakage-resilient secret sharing against colluding parties. In: 60th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2019 (2019). https://doi.org/10.1109/FOCS.2019.00045
14. Lin, F., Cheraghchi, M., Guruswami, V., Safavi-Naini, R., Wang, H.: Non-malleable secret sharing against affine tampering. CoRR abs/1902.06195 (2019). http://arxiv.org/abs/1902.06195
15. Srinivasan, A., Vasudevan, P.N.: Leakage resilient secret sharing and applications. In: Boldyreva, A., Micciancio, D. (eds.) CRYPTO 2019. LNCS, vol. 11693, pp. 480–509. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-26951-7_17
16. Chandran, N. , et al. Adaptive Extractors and Their Application to Leakage Resilient Secret Sharing. In: Advances in Cryptology-CRYPTO 2021. Springer Berlin Heidelberg, 2021: 595-624. https://doi.org/10.1007/978-3-030-84252-9_20