國立臺灣師範大學理學院

資訊工程學系

碩士論文

指導教授: 紀博文 博士

可驗證零知識範圍證明

Authenticated Zero-Knowledge Range Proof

研究生: 秦昊 撰

中華民國 110 年 8 月

致謝

首先誠摯的感謝指導教授紀博文博士，紀老師悉心的教導使我得以一窺資訊安全領域的深奧，不時的討論並指點我正確的方向，使我在這些年中獲益匪淺。老師對學問的嚴謹更是我輩學習的典範。兩年裏的日子，實驗室裏共同的生活點滴，學術上的討論、言不及義的閒扯、讓人又愛又怕的宵夜、趕作業的革命情感、因為睡太晚而遮遮掩掩閃進實驗室。感謝眾位學長姐、同學、學弟妹的共同砥礪，你/妳們的陪伴讓兩年的研究生活變得絢麗多彩。感謝學長不厭其煩的指出我研究中的缺失，且總能在我迷惘時為我解惑，恭喜我們順利走過這兩年。最後，謹以此文獻給我摯愛的雙親。

摘要

零知識範圍證明是個很好用的基礎密碼學演算法。零知識範圍證明可以被用來證明某些想要隱藏的機密在特別的範圍區間之中然而不會洩漏任何跟想要隱藏的機密有關的資訊，但是剛剛提到的特別的範圍區間是公開資訊，這是一個頗嚴重的問題，任何人都可以很輕鬆地選一個在範圍區間內的數字並且宣稱此數字是屬於使用者本身的機密，因為零知識的特性，沒有任何人可以質疑零知識證明的機密的真偽。為了解決這個嚴重的問題，我們整合零知識證明和簽章演算法，在證明者產生證明之前，必須先請第三方可信任團體進行和機密相關的簽章，之後驗證者在驗證零知識範圍證明之前，可以先驗證此簽章是否為證明者本人。我們堅信著可驗證零知識範圍證明一定會對之後的應用非常的有所幫助。

關鍵字：零知識證明、已認證證明、範圍證明.

Abstract

Zero-Knowledge range proof is a useful cryptographic primitive. It can be used to show some secret lies in a specific range without leaking the secret itself. The problem is that the range is public information. Everyone can easily pick a number in the range and claim that the number belongs to the user. Because of the zero-knowledge property, no one can challenge the proof generated from a fake number. To solve this problem, we integrate a signature service with the zero-knowledge proof protocol. Before a prover generates a proof, a trusted-third party needs to create some authenticated primitives, which are related to the message, for the proof generation. So a verifier can check if the proof is authenticated before accepting the proof. We believe that proposed Authenticated Zero Knowledge Range Proof can be beneficial to many applications in the world.

Keywords: Zero-Knowledge Proof, Authenticated Proof, Range Proof.

Contents

List of Tables

List of Figures

Chapter 1 Introduction

1.1 Introduction

Zero-Knowledge Proof (ZKP) is a useful cryptographic primitive. ZKP ensures that a prover can show its knowledge of some secret without revealing the secret directly. Goldwass et al. first proposed ZKP in 1983 [12]. They used two interactive Turing machines with a read-only input tape to simulate a prover and a verifier. They found that with interactions in the proving process, the knowledge of the secret will be hidden. Santis et al. tried to remove the interaction process and proposed the first non-interactive ZKP in 1988 [9]. Santis et al. based on the quadratic residuosity assumption to realize a non-interactive ZKP scheme. There are lots of ZKP works proposed, like [2], [19], [1].

Nowadays, ZKP has been widely used in digital identification mechanisms. For example; Blockchain can use ZKPs to validate the transaction while all other details, like the sender and the receiver identity, are kept secret, [22], [21]. Another example is the smart contract scenario. By ZKPs, users can verify the correctness of contract execution

and money conservation, [15], [16].

Zero-Knowledge Range Proof (ZKRP) is a special type of ZKP. Instead of proving the ownership of a secret, ZKRP is used to show that a secret lies in a given range. Similarly, the proof does not reveal any other additional information. Boudot proposed a zero-knowledge proof scheme that can commit a number lies in an interval [4]. This is the first ZKRP scheme. Boudot based on Fujisaki-Okamoto commitment scheme [10] to construct two proof systems, EL which can commit two numbers are equivalent ,and SQR which can commit a number is a square number. Boudot used these two commit schemes to prove that a number is in a given range. Peng et al. used the secret order principle and computational bindingness through proof of knowledge in cyclic groups to build a ZKRP [4]. Chaabouni et al. replaced the random oracle model with a common reference string (CRS) model to build a non-interactive ZKRP [8]. Bünz et al. proposed a bulletproof system that only relies on the discrete logarithm assumption [5, 6]. Koens et al. applied ZKRP on Ethereum for verifying the smart contract result [14]. Tsai et al. improved previous ZKRP schemes to support arbitrary ranges [20].

There is a ZKRP application example. If Alice wants to run for the organization representative, according to the organization law, Alice must show that her age satisfies the candidate condition. Since age is a secret for all women, Alice does not want to reveal her age to others. So Alice can generate a ZKRP to convince others that her age satisfies

the candidate requirement without releasing her age. The other example about the application of ZKRP is that in some context (for instance; smart contract) of payment systems between two organizations. If organization A wants to transfer payment to organization B, then it can utilize ZKRP to prove that the amount of payment in the transaction is positive, otherwise, if the amount is negative, the payment is from B to A. Moreover, If the government or council want to purchase lots of new computer device, they may issue their tender document for the IT company smart contract. Generally, there is lower bound of tender document, If company C wants to participate the tender, it needs to give the budget, but it does not want to leak any information about the budget. Then it can use the ZKRP to prove that it's budget satisfying the lower bound of the tender in the smart contract.

However, there is a serious problem with ZKRP. That is, the range information is public and everyone can randomly pick a valid number that lies in the range as the secret. Use the above story as an example. Suppose the organization representative's age cannot be over 60 years old and Alice is now 65 years old. Alice can claim that she is now 50 years old, and she uses 50 as her secret, following the ZKRP generation algorithm to provide proof. The proof can pass the ZKRP verification process since it comes from a valid value, which is 50 in this case, and the value does not belong to Alice.

A trivial solution is to find a trusted party to endorse the ZKRP first, like a proof signature, and the receiver can check the signature first, then running the verification process.

However, according to the definition of ZKRP, it reveals nothing except that the secret is in a range. **How can the trusted party sign something it does not know? Note that it is not the blind signature scenario since here we assume the prover may lie to others with fake secrets.**

1.2 Contribution

To solve this problem, in this paper we propose an Authenticated Zero-Knowledge Range Proof (AZKRP) mechanism. We use the concept, authentication-then-proof, to build a ZKRP. When a user wants to generate a ZKRP from some secret, the secret, not the proof, must be authenticated by a trusted third party (TTP), which can be treated as a signature. The signature does not reveal the secret. In the proof generation process, the prover must use the signature as a part of the operation argument. When verification, the verifier needs to check the TTP signature first to make sure that the proof comes from an authenticated secret. If the signature verification fails, the verifier simply drops the proof. Otherwise, the verifier runs the ZKRP verification process to check if the value is in the range.

Our scheme is based on a non-interactive zero-knowledge range proof scheme proposed by Tsai et al.[20]. We enhance their work with a pairing-based signature scheme. Our contributions are as follows.

1. We develop an authenticated zero-knowledge range proof without leaking any information about the secret except the range check.

2. Our scheme supports the offline TTP signature. That is, TTP does not need to participate in the proof generation process.

3. Our scheme supports multiple-times use signature. A signed secret can be used to generate multiple zero-knowledge range proofs and all proofs are different to outsiders even when they are generated from the same secret.

The rest of this paper is organized as follows. In Chapter 2, we describe some preliminaries that will be used in our construction. In Chapter 3, we give our construction in detail. In Chapter 4, we evaluate the properties of our scheme security, including the correctness, soundness, and zero-knowledge proof. In Chapter 5, we compare our scheme to other ZKRP schemes and give a efficiency analysis. The last Chapter is our conclusion.

Chapter 2 Preliminaries

2.1 Notation

We denote the range between a and b as (a, b), and define $a||b$ the concatenation of string a and b. We denote function neg be the negligible function. We define $X \xleftarrow{R} Y$ that X is randomly chosen by Y. $X \xleftarrow{R} Y$ is a set up algorithm for choosing random number.

2.2 Prime Order Bilinear Group

Let $\mathbb{G}$ and $\mathbb{G}_T$ be two multiplicative cyclic groups of prime order p, with map function $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$. Let g be a generator of $\mathbb{G}$. $\mathbb{G}$ is a bilinear map group if $\mathbb{G}$ and e have the following properties:

- Bilinearity: $\forall u, v \in \mathbb{G}$ and $a, b \in \mathbb{Z}_p$, $e(u^a, v^b) = e(u, v)^{ab}$.
- Non-degeneracy: $e(g, g) \neq 1$.
- Computability: the group action in $\mathbb{G}$ and function e can be computed efficiently.

2.3 Composite Order Bilinear Group

The composite order bilinear group was first introduced in [3]; we use it to construct our scheme. First, we provide a brief introduction. Let $\mathbb{G}$ and $\mathbb{G}_T$ be two multiplicative cyclic groups of composite order $N = p_1 p_2 \dots p_m$, where $p_1, p_2, \dots, p_m$ are distinct primes, with bilinear map function $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$. For each prime p_i, $\mathbb{G}$ has a subgroup $\mathbb{G}_{p_i}$ of order p_i. We let $g_1, g_2, \dots, g_m$ be the generators of these subgroups respectively. Each element in $\mathbb{G}$ can be expressed in the form of $g_1^{a_1} g_2^{a_2} \dots g_m^{a_m}$, where $a_1, a_2, \dots, a_m \in \mathbb{Z}_N$. If a_i is congruent to zero modulo p_i, we say that this element has no $\mathbb{G}_{p_i}$ component. We say an element is in $\prod_{i \in S} \mathbb{G}_{p_i}$, where S is a subset from $1 \dots m$, if $\forall i \in S$, a_i is not congruent to zero modulo p_i.

The most important property of the composite bilinear groups is **orthogonality** between all subgroups under bilinear map e. This means that if $u \in \mathbb{G}_{p_i}, v \in \mathbb{G}_{p_j}$ and $i \neq j$, then $e(u, v) = 1$, where 1 is the identity element in $\mathbb{G}_T$.

2.4 Randomizable Signature

Camenisch and Lysyanskaya proposed a new pairing-based signature scheme in 2004 [7]. The most interesting feature of this signature scheme is that the signature can be randomized. That is, given a valid signature of some message, anyone can generate another

valid signature for the same message. We call this a randomizable signature. Pointcheval and Sanders developed another randomizable signature scheme by using only two elements and had better performance [17]. They also used the Strong Diffie-Hellman (SDH) assumption to construct another similar scheme [18].

We introduce a basic Pointcheval-Sanders(PS) signature scheme as follows:

PS signature scheme bases on r-vector messages $(m_1, \cdots, m_r) \in \mathbb{Z}_p^r$, and PS signature scheme security relies on the **Definition 1** (PS Assumption). We introduce the algorithm as follows:

- **Keygen**$(1^k) \longrightarrow pp$; where $pp \longleftarrow (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e)$
- **Keygen**$(pp) \longrightarrow (sk, pk)$; where $\mathbb{G}_i^*$ is the subset of generators $\mathbb{G}_i \setminus \{1_{\mathbb{G}_i}\}$ for $i = 1, 2$;
  - $\tilde{g} \xleftarrow{\$} \mathbb{G}_2^*$ and $(x, y_1, \cdots, y_r) \xleftarrow{\$} (\mathbb{Z}_p^*)^r + 1$
  - $(\tilde{g}, \tilde{X}, \tilde{Y}_1, \cdots, \tilde{Y}_r) \longleftarrow (\tilde{g}^x, \tilde{g}^{y_1}, \cdots, \tilde{g}^{y_r})$
  - $sk \longleftarrow (x, y_1, \cdots, y_r)$ and $pk \longleftarrow (\tilde{g}, \tilde{X}, \tilde{Y}_1, \cdots, \tilde{Y}_r)$
- **Sign**$(sk, (m_1, \cdots, m_r)) \longrightarrow \sigma$; where
  - $h \xleftarrow{\$} \mathbb{G}_1^*$
  - $\sigma \longleftarrow (h, h^{(x+\sum y_j \cdot m_j)})$

- **Verify**$(pk, (m_1, \cdots, m_r), \sigma) \longrightarrow (0, 1)$; where

  - verify $\sigma_1 \neq 1_{\mathbb{G}_1}$ and $e(\sigma_1, \tilde{X} \cdot \prod \tilde{Y}_j^{m_j}) = e(\sigma_2, \tilde{g})$

Definition 1 (Security of Signature Scheme)**.** *The standard security notion for a signature scheme is existential unforgeability under chosen message attacks(EUF-CMA) [13]: it means that it is hard to output a valid (m, σ) for a secret m never asked to the signing oracle, even given access to a signing oracle. It is defined using the following game between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$:*

- **Setup**: $\mathcal{C}$ runs the **Setup** and the **Keygen** algorithms to obtain (pp, sk, pk) and gives the pk to the adversary.

- **Queries**: $\mathcal{A}$ adaptively requests signatures on at most q secret $(m_1, \cdots, m_q)$ and then $\mathcal{C}$ answer each.

- **Output**: $\mathcal{A}$ eventually outputs a secret-signature pair (m^*, σ^*) and wins the game if **Verify**$(pk, m^*, \sigma^*) = 1$ while $m^* \neq m_i \; \forall i \in [1, q]$.

Definition 2 (Unlinkability)**.** *For any adversaries $\mathcal{A}$, the probability that $\mathcal{A}$ can use lots of valid signature σ to get the secret key is negligible.*

Definition 3 (Unforgeability)**.** *For any adversaries $\mathcal{A}$, the probability that $\mathcal{A}$ can gerner-ate a valid pair signature without secret key is negligible.*

We can note that a signature $\sigma = (\sigma_1, \sigma_2)$ is randomizable, only both raising the same non-zero power. (for example $\sigma' = (\sigma_1^3, \sigma_2^3)$). The invariant is the discrete logarithm of σ_2 in basis σ_1, hence, the unlinkability relies on the Decisional Diffie – Hellman(DDH) assumption. The unforgeability has been proven to hold under the PS assumption. In this paper, we apply the randomizable signature scheme to make the commitment signature in each proof.

2.5 Two commitments hide the same secret. (EL Proof)

EL Proof was proposed by Boudot in 2000 [4], It can be used to validate that two commitments hide the same secret. Because EL Proof is a kind of zero-knowledge proof, it means that only prover knows the secret m about the proof. Here, we combine EL Proof with bilinear group map.

Alice wants to convince Bob that two commitments $A = {u_1}^m \cdot {v_1}^{s_1}, B = {u_2}^m \cdot {v_2}^{s_2}$ both hide the same secret m, here $u_1 = e(g_1, t_1), v_1 = e(h_1, t_1)$ and $u_2 = e(g_1', t_1'), v_2 = e(h_1', t_1')$ where $g_1, g_1', t_1, t_1' \in \mathbb{G}_1$ and $s_1, s_2 \xleftarrow{R} \mathbb{N}$. We denote the algorithm of generating EL proof as $EL \xleftarrow{R} SS(m, s_1, s_2, u_1, v_1, u_2, v_2, A = {u_1}^m \cdot {v_1}^{s_1}, B = {u_2}^m \cdot {v_2}^{s_2})$. We denote the algorithm of verifying EL proof as $vEL \xleftarrow{R} vSS(EL, u_1, v_1, u_2, v_2, A, B)$. Prover do the following steps;

- Prover knows secret m and s_1, s_2, then Prover wants to convince Verifier that $(A = {u_1}^m \cdot {v_1}^{s_1}, B = {u_2}^m \cdot {v_2}^{s_2})$ both hide the same secret m. Hence, Prover do the algorithm $SS(m, s_1, s_2, u_1, v_1, u_2, v_2, A = {u_1}^m \cdot {v_1}^{s_1}, B = {u_2}^m \cdot {v_2}^{s_2})$ as following;

  1. Choose random integers θ, τ_1, τ_2 Then compute $C_1 = {u_1}^{\theta} \cdot {v_1}^{\tau_1}$ and $C_2 = {u_2}^{\theta} \cdot {v_2}^{\tau_2}$.

  2. Compute $H = Hash(C_1||C_2)$

  3. Generate $X' = \theta + H \cdot m,\ X'_1 = \tau_1 + H \cdot s_1,\ X'_2 = \tau_2 + H \cdot s_2$

  4. Publish $EL = (H, X', X'_1, X'_2)$

- Verifier wants to check that A and B both hide the same secret. Therefore, Verifier do the algorithm $vSS(EL, u_1, v_1, u_2, v_2, A, B)$ as following;

  – Verify $H = Hash({u_1}^{X'} \cdot {v_1}^{X'_1} \cdot A^{-H}||{u_2}^{X'} \cdot {v_2}^{X'_2} \cdot B^{-H})$

  1. $C_1 = {u_1}^{\theta} \cdot {v_1}^{\tau_1} = e(g_1, t_1)^{\theta} \cdot e(h_1, t_1)^{\tau_1} = e(g_1, t_1)^{\theta + H \cdot m} \cdot e(h_1, t_1)^{\tau_1 + H \cdot s_1} \cdot e(g_1, t_1)^{-H \cdot m} \cdot e(h_1, t_1)^{-H \cdot s_1} = {u_1}^{\theta + H \cdot m} \cdot {v_1}^{\tau_1 + H \cdot s_1} \cdot {u_1}^{-H \cdot m} \cdot {v_1}^{-H \cdot s_1} = {u_1}^{X'} \cdot {v_1}^{X'_1} \cdot A^{-H}$

  2. $C_2 = {u_2}^{\theta} \cdot {v_2}^{\tau_2} = e(g'_1, t'_1)^{\theta} \cdot e(h'_1, t'_1)^{\tau_2} = e(g'_1, t'_1)^{\theta + H \cdot m} \cdot e(h'_1, t'_1)^{\tau_2 + H \cdot s_2} \cdot e(g'_1, t'_1)^{-H \cdot m} \cdot e(h'_1, t'_1)^{-H \cdot s_2} = {u_2}^{\theta + H \cdot m} \cdot {v_2}^{\tau_2 + H \cdot s_2} \cdot {u_2}^{-H \cdot m} \cdot {v_2}^{-H \cdot s_2} = {u_2}^{X'} \cdot {v_2}^{X'_2} \cdot B^{-H}$

  3. $H = Hash({u_1}^{X'} \cdot {v_1}^{X'_1} \cdot A^{-H}||{u_2}^{X'} \cdot {v_2}^{X'_2} \cdot B^{-H}) = Hash(C_1||C_2)$

- If proof can pass, the verifier will be convinced that both A and B hide the same secret.

2.6 The commitment hides the square secret. (SQR Proof)

SQR Proof was proposed by Boudot in 2000 [4], It can be used to validate that the commitment hides the square secret value. Because SQR Proof is a kind of zero-knowledge proof, it means that only prover knows the secret α about the proof. Here, we combine SQR Proof with bilinear group map.

Alice wants to convince Bob that the commitment $E = u^{\alpha^2} \cdot v^{r_1}$ hides the square secret α^2. We denote the algorithm of gernerating SQR proof as $SQR \xleftarrow{R} SQ(\alpha^2, r, u, v, E = u^{\alpha^2} \cdot v^{r_1})$. here $u = e(g_1, t_1), v = e(h_1, t_1)$ where $g_1, t_1 \in \mathbb{G}_1$ We denote the algorithm of verifying SQR proof as $vSQR \xleftarrow{R} vSQ(SQ, u, v, E = u^{\alpha^2} \cdot v^{r_1})$. Prover do the following steps;

- Prover knows secret α and r_1, then Prover wants to convince Verifier that $E = u^{\alpha^2} \cdot v^{r_1}$ hide the square value α^2. Hence, Prover do the algorithm $SQ(\alpha^2, r_1, u, v, E = u^{\alpha^2} \cdot v^{r_1})$ as following;

  1. Choose random number r_2 and compute $F = u^{\alpha} \cdot v^{r_2}$
  2. Compute $r_3 = r_1 - r_2 \cdot \alpha$

3. Compute $E' = F^{\alpha} \cdot v^{r_3}$

4. In order to convincing E' and F both commitment hide the same secret run the $SS(\alpha, r_2, r_3, u, v, F, v, E' = F^{\alpha} \cdot v^{r_3}, F = u^{\alpha} \cdot v^{r_2})$

   (a) Choose random integers θ, τ_1, τ_2 Then compute $C_1 = u^{\theta} \cdot v^{\tau_1}$ and $C_2 = F^{\theta} \cdot v^{\tau_2}$.

   (b) Compute $H = Hash(C_1||C_2)$

   (c) Generate $X' = \theta + H \cdot \alpha,\ X'_1 = \tau_1 + H \cdot r_2,\ X'_2 = \tau_2 + H \cdot r_3$

   (d) Publish $EL = (H, X', X'_1, X'_2, F)$

- Verifier wants to check that E hide the square secret. Therefore, Verifier do the algorithm $vSQ(SQ, u, v, E = u^{\alpha^2} \cdot v_1^r)$ as following;

  – Verify $H = Hash(u^{X'} \cdot v^{X'_1} \cdot F^{-H}||F^{X'} \cdot v^{X'_2} \cdot E^{-H})$

  1. $C_1 = u^{\theta} \cdot v^{\tau_1} = e(g_1, t_1)^{\theta} \cdot e(h_1, t_1)^{\tau_1} = e(g_1, t_1)^{\theta + H \cdot \alpha} \cdot e(h_1, t_1)^{\tau_1 + H \cdot r_2} \cdot e(g_1, t_1)^{-H \cdot \alpha} \cdot e(h_1, t_1)^{-H \cdot r_2} = u^{\theta + H \cdot \alpha} \cdot v^{\tau_1 + H \cdot r_2} \cdot u^{-H \cdot \alpha} \cdot v^{-H \cdot r_2} = u^{X'} \cdot v^{X'_1} \cdot F^{-H}$

  2. $C_2 = F^{\theta} \cdot v^{\tau_2} = (u^{\alpha} \cdot v^{r_2})^{\theta} \cdot v^{\tau_2} = e(g_1, t_1)^{\alpha \cdot \theta} \cdot e(h_1, t_1)^{r_2 \cdot \theta + \tau_2} = e(g_1, t_1)^{\alpha \cdot \theta + H \cdot \alpha^2} \cdot e(h_1, t_1)^{r_2 \cdot \theta \cdot H \cdot \alpha + \tau_2 + H \cdot r_3} \cdot e(g_1, t_1)^{-H \cdot \alpha^2} \cdot e(h_1, t_1)^{-H \cdot \alpha H \cdot r_3} = (u^{\alpha} \cdot v^{r_2})^{X'} \cdot v^{X'_2} \cdot E'^{-H} = F^{X'} \cdot v^{X'_2} \cdot E'^{-H}$

  3. $H = Hash(u^{X'} \cdot v^{X'_1} \cdot F^{-H}||F^{X'} \cdot v^{X'_2} \cdot E^{-H}) = Hash(C_1||C_2)$

- If proof can pass, the verifier will be convinced that commitment E hide the square secret.

2.7 Tsai Non-Interactive ZKRP (NIZKRP) Scheme

Our work is based on Tasi NIZKRP scheme[20]. So here we simply introduce their work. Tsai et al. use the fact that $a \leq m \leq b$ is equivalent to $(m-a+1)\cdot(b-m+1) > 0$, where $a, b, m \in \mathbb{N}$. However, there is high probability to leak the secret m if we directly use the Inequation $(m-a+1)\cdot(b-m+1) > 0$. Therefore, we add in a random number $\omega \in \mathbb{Z} \setminus \{0\}$ to prevent the above problem. Hence, the original problem becomes to the $\omega^2 \cdot (m-a+1)\cdot(b-m+1) > 0$. So the zero-knowledge range proof is reduced to the problem showing that $\omega^2 \cdot (m-a+1)\cdot(b-m+1)$ is greater than zero.

First, Tasi et al. use Fujisaki-Okamoto commitment scheme[10] to commit the secret. Then, they split $\omega^2 \cdot (m-a+1)\cdot(b-m+1)$ into two numbers M and R, where M is a square number and R is a positive number. With the help of the equivalent proof (EL proof) and the square proof (SQR proof) proposed by Boudot [4], they can commit that M is a square number and R is simply delivered in plaintext.

The formal definition of NIZKRP is given as follows.

Definition 4 (Non-Interactive Zero-Knowledge Range Proof)**.** *A non-interactive zero-*

knowledge range proof (NIZKRP) is a tuple of algorithms NIZKRP = (**Setup**, **Prove**, **Verify**) *specified as follows.*

1. **Setup**$(\lambda) \rightarrow p$ *: The probabilistic algorithm* **Setup** *which takes the security parameter* λ *as input, and outputs system parameter* p*.*

2. **Prove**$(p, m, a, b) \rightarrow \pi$ *: The probabilistic algorithm* **Prove** *which takes* p*, secret value* m *and the range value* (a, b) *as input, and outputs the proof* π*.*

3. **Verify**$(p, \pi) \rightarrow d$ *: The probabilistic algorithm* **Verify** *which takes* p *and* π *as input, it returns a decision bit* $d \in \{0, 1\}$ *according to* π*. When* $d = 1$*, it means verifier accepts the proof* π *that* m *is in* (a, b)*. Otherwise, it rejects the proof.*

NIZKRP should satisfy three properties, **correctness**, **soundness**, and **zero-knowledge**. The correctness states that every honest prover can follow the protocol and can generate a convincing range proof if the secret is in the range (a, b), where $a < b$. The soundness states that the prover can only generate valid proof with a negligible probability. The zero-knowledge states that a probabilistic polynomial time (PPT) adversary learns nothing about m from the proof.

The formal definitions of these properties are given as follows.

Definition 5 (Correctness)**.** *Let* $\Pi =$ (**Setup**, **Prove**, **Verify**) *be a non-interactive zero-knowledge range proof scheme. We say that* Π *satisfies correctness property if for all*

security parameter λ, secure value m within the given range (a, b), the following probability is satisfied.

$$Pr\left[\begin{array}{c}\textbf{Verify}(p,\pi)=1: \\ \hline \textbf{Setup}(\lambda)\rightarrow p; \\ \textbf{Prove}(p,m,a,b)\rightarrow\pi\end{array}\right]\geq 1-neg(\lambda)$$

Definition 6 (Soundness)**.** *Let* Π = *(***Setup**,**Prove**,**Verify***) be a non-interactive zero-knowledge range proof scheme. We say that* Π *satisfies soundness if there exists a PPT simulator* S *and for all* λ *such that the following probability is negligible.*

$$Pr\left[\begin{array}{c}\textbf{Verify}(p,\pi)=1\wedge m\notin(a,b) \\ \hline \textbf{Setup}(\lambda)\rightarrow p; \\ S(p,m)\rightarrow\pi\end{array}\right]\leq neg(\lambda)$$

Definition 7 (Zero-Knowledge)**.** *Let* Π = *(***Setup**,**Prove**, **Verify***) be a non-interactive zero-knowledge range proof scheme. We say that* Π *satisfies Zero-knowledge if there exists a PPT simulator* S *and for all security parameter* λ*, adversaries* $\mathcal{A}$*, such that the following*

probability is negligible.

$$|Pr\left[\begin{array}{c} w = w^{'} : \\ \hline \textbf{Setup}(\lambda) \rightarrow p; \\ w \xleftarrow{R} \{0,1\}; \\ \mathcal{A}(p) \rightarrow m_1; \\ m_0, m_1 \in [a,b] \\ \textbf{Prove}(p, m_0, a, b) \rightarrow \pi_0; \\ S(p, m_1) \rightarrow \pi_1; \\ \mathcal{A}(p, \pi_w) \rightarrow w' \end{array}\right] - \frac{1}{2}| \leq neg(\lambda)$$

The main problem of the Tsai NIZKRP scheme is the soundness property. Although the soundness property guarantees that if m is not in the range, the prover cannot generate a computationally valid proof. However, (a, b) is input by the prover and the dishonest prover can pick a valid number m' to generate the proof. According to the correctness property, the generated proof can pass the verification. With the zero-knowledge property, the verifier cannot learn anything about the secret so it cannot challenge the dishonest prover.

Chapter 3 Authenticated Zero-Knowledge Range Proof

In this Chapter, we introduce a new scheme called Authenticated Zero-Knowledge Range Proof (AZKRP). We will introduce our idea first and then define AZKRP. The construction is in section 3.3.

3.1 Our Idea

The main problem of ZKRP is that the range information is public. Unlike ZKP where the secret is generally an answer to a computationally hard problem, ZKRP requires the secret lies in an explicit interval. Therefore, it is easy to pick a valid number from the range. That is, a zero-knowledge range proof becomes meaningless since everyone can generate valid proof even its secret is not in the range. The main cause is that the secret

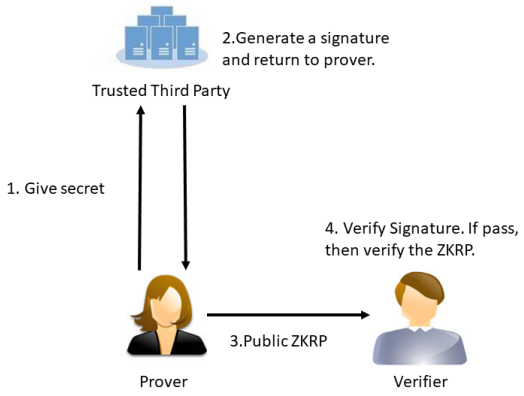

Figure 3.1: AZKRP Architecture

is not authenticated. To solve this problem, we enhance a ZKRP scheme by integrating a signature scheme. We import a trusted third party in our scheme for secret authentication. We make the secret be authenticated before generating a zero-knowledge range proof. The architecture is shown in Fig. 3.1.

There are three entities here, a prover, a verifier, and a TTP. First, the prover asks the TTP to sign its secret. Here we assume that the TTP has a way to verify if the prover's secret is valid. For example, the TTP knows if the given date is the prover's birthday. After validation, the TTP signs the secret from the prover. Unlike other signature schemes where others can verify the signature directly, our signature is composed of some elements that will be used in the proof generation. The prover cannot generate these elements since it does not have the TTP's secret key.

When the prover wants to show that its secret is in the given range, the prover uses the signature obtained from the TTP and the secret about the prover to generating the zero-knowledge range proof. Note that the TTP signature generation is before the range

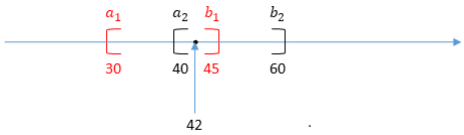

Figure 3.2: Signatures intersection

information. That is, TTP can be offline in the proof generation and verification phase. After receiving the proof, the verifier first verifies the TTP signature part. If the verification fails, the verifier drops the proof. Otherwise, the verifier checks if the secret is in the range.

However, if a signature uses multiple times, that is, a secret is used to generate multiple zero-knowledge range proofs, outsiders may use the signature to link different proofs since they have the same signature. Here the outsider is the one who does not have the TTP's public key to verify the signature. We believe that this scenario is real since in most cases, range proof verification is only performed in limited entities. Given a group that contains those valid provers and verifiers. The members in this group can obtain the TTP's public key. **That is, only those people in the group can obtain the TTP's public key. The Outsider cannot obtain the TTP's public key, since the outsider does not belong to the group.**

Once outsiders link different zero-knowledge range proofs according to their signatures, outsiders know that the secret lies in the intersection of different ranges, which means the range is shortened. For example, as Fig. 3.2, the secret is **42**, and the inter-

section of range $(a_1, b_1) = (30, 45)$ and $(a_2, b_2) = (40, 60)$ is $(\mathbf{40}, \mathbf{45})$. It makes outsider that has more probability to obtain the secret of **42**, because of the range is shortened to $(\mathbf{a_2}, \mathbf{b_1}) = (\mathbf{40}, \mathbf{45})$. To avoid this problem, we develop the randomization feature in our scheme. We apply the randomizable signature scheme to change the signature every time. As for the commitment, we make the commitment composed of two orthogonal subgroup elements. So the signature is generated from one subgroup while the prover can change the other subgroup in each proof generation. Therefore, the commitment can be different in each zero-knowledge range proof.

3.2 Assumptions

We assume that our randomizable signature scheme is secure. In addition, we assume that provers in the same group do not make collusion. That is the probability that multiple provers link lots of commitments and get the TTP's secret key is negligible.

3.3 Definition

The formal definition of AZKRP is as follows.

Definition 8 (Authenticated Zero-Knowledge Range Proof)**.** *An authenticated zero-knowledge range proof (AZKRP) is a tuple of algorithms AZKRP =* (**Setup**, **Sign**, **Prove**, **Verify**)

Table 3.1: AZKRP roles with their algorithms.

Party members	Algorithm
TTP	**Setup**, **Sign**
Prover	**Prove**
Verifier	**Verify**

specified as follows.

1. **Setup**$(\lambda) \rightarrow (\mathcal{E}, \mathcal{F}, \mathcal{K})$ *: The algorithm takes the security parameter* λ *as the input, and outputs system-wise parameter* $\mathcal{E}$*, public information for specific group* $\mathcal{F}$ *and secret* $\mathcal{K}$*.*

2. **Sign**$(\mathcal{E}, \mathcal{F}, \mathcal{K}, m) \rightarrow (C^*, S)$ *: The algorithm takes a message* m *as the input. With* $\mathcal{E}, \mathcal{F}, \mathcal{K}$*, the system computes a commitment* C^* *about* m*. Then the algorithm signs a variance of* C^**. The algorithm outputs the commitment and the signature.*

3. **Prove**$(\mathcal{E}, \mathcal{F}, a, b, m, C^*, S) \rightarrow \pi$ *: The algorithm takes* $\mathcal{E}$*,*$\mathcal{F}$*, range value* (a, b)*, secret value* m *and* C^* *as input, and outputs the proof* π*. The signature* S *is also included in* π*.*

4. **Verify**$(\mathcal{E}, \mathcal{F}, \pi) \rightarrow w$ *: The algorithm takes* $\mathcal{E}$*,*$\mathcal{F}$*, and* π *as input, it returns a decision bit* $w \in \{0, 1\}$ *according to* π*. When* $w = 1$*, it means the verifier accepts the proof* π*. Otherwise, it rejects the proof and the output is 0.*

Since there are three different roles in AZKRP, we use table 3.1 to indicate the relationship between the role and the algorithms.

Just like ZKRP, AZKRP satisfies three security properties: **correctness**, **soundness**, and **zero-knowledge**. The property definitions are similar with ZKRP, except **soundness**. In a ZKRP scheme, the soundness property is that the prover cannot generate valid proof if the secret is not in the range. We have shown that an adversary can easily get a number in the range and can derive valid proof. So here we redefine the soundness property. In our AZKRP scheme, the soundness property is that the prover cannot generate valid proof if the secret is **not authenticated** or is not in the range. This additional requirement ensures that the prover's secret must be authenticated before proof generation. The formal definitions are as follows.

Definition 9 (Correctness)**.** *Let* $\Pi' = ($**Setup**, **Sign**, **Prove**, **Verify**$)$ *be an authenticated zero-knowledge range proof. We say that* Π' *satisfies correctness if for all security parameter* λ*, secure value* m *within the given range* (a, b)*, the following probability is satisfied.*

$$Pr\left[\begin{array}{c} \textbf{Verify}(\mathcal{E}, \mathcal{F}, \pi) \rightarrow w = 1 : \\ \hline \textbf{Setup}(\lambda) \rightarrow (\mathcal{E}, \mathcal{F}, \mathcal{K}); \\ \textbf{Sign}(\mathcal{E}, \mathcal{F}, \mathcal{K}, m) \rightarrow (C^*, S); \\ \textbf{Prove}(\mathcal{E}, \mathcal{F}, a, b, m, C^*, S) \rightarrow \pi; \end{array}\right] \geq 1 - neg(\lambda)$$

Definition 10 (Soundness)**.** *Let* $\Pi' = ($**Setup**, **Sign**, **Prove**, **Verify**$)$ *be an authenticated zero-knowledge range proof. We say that* Π' *satisfies soundness if there exists a PPT*

*simulator **S** and for all λ such that the following two probability is negligible.*

$$Pr\left[\begin{array}{c}\textbf{Verify}(\mathcal{E},\mathcal{F},\pi)=1:\\ \hline \textbf{Setup}(\lambda)\rightarrow(\mathcal{E},\mathcal{F},\mathcal{K});\\ \textbf{S}(\mathcal{E},m)\rightarrow(\pi)\wedge m\in(a,b);\end{array}\right]\leq neg(\lambda)$$

and

$$Pr\left[\begin{array}{c}\textbf{Verify}(\mathcal{E},\mathcal{F},\pi)=1\wedge m\notin(a,b):\\ \hline \textbf{Setup}(\lambda)\rightarrow(\mathcal{E},\mathcal{F},\mathcal{K});\\ \textbf{Sign}(\mathcal{E},\mathcal{F},\mathcal{K},m)\rightarrow(C^*,S);\\ \textbf{S}(\mathcal{E},m,C^*,S)\rightarrow\pi;\end{array}\right]\leq neg(\lambda).$$

The first probability is that the unauthenticated message, which is even in the range, cannot be used to generate valid proof. The second probability is that the authenticated message cannot be used to generate valid proof if the message is not in the range.

Definition 11 (Zero-Knowledge)**.** *Let $\Pi' =$ (***Setup***,* **Sign***,* **Prove***,* **Verify***) be an authenticated zero-knowledge range proof. We say that Π' satisfies Zero-Knowledge if there exists a PPT simulator **S** and for all λ, , adversaries $\mathcal{A}$, such that the following probability is*

negligible.

$$|Pr\left[\begin{array}{c} w = w' : \\ \hline \mathbf{Setup}(\lambda) \rightarrow (\mathcal{E}, \mathcal{F}, \mathcal{K}); \\ \mathbf{Sign}(\mathcal{E}, \mathcal{F}, \mathcal{K}, m_0) \rightarrow (C^*, S); \\ \mathcal{A}(\mathcal{E}) \rightarrow m_1; \\ m_0, m_1 \in [a, b]\ ;\ w \xleftarrow{R} \{0,1\}; \\ \mathbf{Prove}(\mathcal{E}, \mathcal{F}, a, b, m_0, C^*, S) \rightarrow \pi_0; \\ \mathbf{S}(\mathcal{E}, m_1) \rightarrow \pi_1; \\ \mathcal{A}(\mathcal{E}, \pi_w) \rightarrow w' \end{array}\right] - \frac{1}{2}| \leq neg(\lambda)$$

3.4 Construction

In this section, we describe how to construct our AZKRP in detail.

- **Setup**$(1^\lambda) \rightarrow (\mathcal{E}, \mathcal{F}, \mathcal{K})$: Given a security parameter λ, the algorithm generates bilinear group $\mathbb{G}$ of order $N = p_1 p_2$, where p_1, p_2 are distinct primes with bilinear map function $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$, where $\mathbb{G}_T$ is also order N. We let $\mathbb{G}_1, \mathbb{G}_2$ denote two orthogonal subgroups in $\mathbb{G}$ of order p_1, p_2, respectively.

  The algorithm picks up a generator $g \in \mathbb{G}$, where g can be denoted as $g = g_1 \cdot g_2$, and $g_1 \in \mathbb{G}_1$, $g_2 \in \mathbb{G}_2$. The algorithm randomly picks $\alpha, \beta, x \in \mathbb{Z}_N$, and computes

$h = g^{\alpha}$, $t = g^{\beta}$. Note that h can be presented as $h = h_1 \cdot h_2$ and t can be presented as $t = t_1 \cdot t_2$, where $h_1, t_1 \in \mathbb{G}_1$ and $h_2, t_2 \in \mathbb{G}_2$. The algorithm computes $y = t_2^x$. Note that y is a public key for verifying the authenticated zero-knowledge range proof and x is a private key for generating the signature about the commitment which contain the prover's secret. The algorithm then setups a randomizable signature Σ. The output public information $\mathcal{E}$, for the specific group that contain the valid provers and verifiers information $\mathcal{F}$ and secret value $\mathcal{K}$ are as follows:

$$\mathcal{E} = \{\mathbb{G}, e, N, g, t_1, \Sigma\}.$$

$$\mathcal{F} = \{h_1, y\}.$$

$$\mathcal{K} = \{\alpha, \beta, x\}.$$

Note that Σ in $\mathcal{E}$ does not include the signature private key.

- **Sign**$(\mathcal{E}, \mathcal{F}, \mathcal{K}, m) \rightarrow (C^*, S)$: The algorithm first randomly picks $r^* \in \mathbb{Z}_N$. The algorithm computes a commitment $C^* = g^m \cdot h^{r^*}$. Instead of signing C^* directly, the algorithm computes $M = e(C^*, y)$ and signs M with Σ to get the signature S. The algorithm outputs a signature (C^*, S). Note that r^* is included in C^*.

- **Prove**$(\mathcal{E}, \mathcal{F}, a, b, m, C^*, S) \rightarrow \pi$: The algorithm uses the following steps to generate the proof π;

  1. The algorithm first randomizes signature S.

2. The algorithm randomizes commitment C^* to C as follows. the algorithm randomly picks $r_{temp} \in \mathbb{Z}_N$ and computes $C = C^* \cdot h_1^{r_{temp}}$. So $C = g^m \cdot h_1^{r^*+r_{temp}} \cdot h_2^{r^*}$. Note that C can be treated as a commitment generated from r, where $r \mod p_1 \equiv r^* + r_{temp}$ and $r \mod p_2 \equiv r^*$. The algorithm then calculates

$$c = e(C, t_1) = e(g, t_1)^m \cdot e(h, t_1)^r = e(g_1, t_1)^m \cdot e(h_1, t_1)^r = u^m \cdot v^r.$$

For simplicity, we denote $e(g_1, t_1)$ as u and $e(h_1, t_1)$ as v. Note that $e(g, t_1) = e(g_1, t_1)$ because of the canceling property. We use the suffix to show the element's subgroup. The algorithm generates

$$c_1 = c \cdot u^{-a+1} = u^{m-a+1} \cdot v^r,$$

$$c_2 = c^{-1} \cdot u^{b+1} = u^{b-m+1} \cdot v^{-r},$$

$$c_{EL} = c_1^{b-m+1} \cdot v^{r_{EL}} \ , \ r_{EL} \xleftarrow{R} \mathbb{Z}_N.$$

3. To convince that c_2 and c_{EL} commits the same secret $b - m + 1$, the prover produces an EL proof.

$$EL = SS(b - m + 1, -r, r_{EL}, u, v, c_1, v, c_2, c_{EL}).$$

4. The prover commits a square number ω^2 as follows.

$$c_{SQR_1} = (c_{EL})^{\omega^2} \cdot v^{r_{SQR_1}} \ , \ r_{SQR_1} \xleftarrow{R} \mathbb{Z}_N.$$

5. To convince that c_{SQR_1} commits a square number ω^2, the algorithm produces a SQR_1 proof as follows.

$$SQR_1 = SQ(\omega^2, r_{SQR_1}, c_{EL}, v, c_{SQR_1}).$$

6. The algorithm calculates $X = \delta^2, \ \delta \xleftarrow{R} \mathbb{Z}_N$.

   If $X \geq \omega^2 \cdot (m - a + 1) \cdot (b - m + 1)$, repeat this step.

7. The algorithm calculates

$$Y = \omega^2 \cdot (m - a + 1) \cdot (b - m + 1) - X.$$

8. The algorithm randomly picks $r_X \in \mathbb{Z}_N$ and computes r_Y as follows.

$$r_Y = \omega^2((b - m + 1) \cdot r + r_{EL}) + r_{SQR_1} - r_X.$$

9. The algorithm commits X, Y with r_X, r_Y.

$$c_X = u^X \cdot v^{r_X},$$

$$c_Y = u^Y \cdot v^{r_Y}.$$

10. To convince that c_X commits a square number $X = \delta^2$, the algorithm produces a SQR_2 proof.

$$SQR_2 = SQ(X, r_X, u, v, c_X).$$

11. Finally, the algorithm outputs the proof π as follows.

$$\pi = \left\{ \begin{array}{l} a, b, C, S, c_{EL}, c_{SQR_1}, c_X, c_Y, \\ \\ EL, SQR_1, Y, SQR_2 \end{array} \right\}.$$

- **Verify**$(\mathcal{E}, \mathcal{F}, \pi) \rightarrow w$: The algorithm uses the following steps to verify the proof π;

1. The algorithm calculates $M' = e(C, y)$. Then the algorithm verifies if S is a valid signature of M' with Σ. If the signature is invalid, the algorithm outputs 0, which means that the proof is rejected.

2. The algorithm calculates c, c_1, c_2 as follows.

$$c = e(C, t_1),$$

$$c_1 = c \cdot u^{-a+1} = u^{m-a+1} \cdot v^r,$$

$$c_2 = c^{-1} \cdot u^{b+1} = u^{b-m+1} \cdot v^{-r}.$$

3. The algorithm checks if c_2 and c_{EL} are from the same values with EL by the algorithm vEL. If EL is an invalid proof, the algorithm outputs 0.

4. The algorithm checks if c_{SQR_1} is a commitment from the square number with SQR_1 by the algorithm $vSQR$. If SQR_1 is an invalid proof, the algorithm outputs 0.

5. The algorithm checks if c_{SQR_1} is equal to $c_X c_Y$. If the equivalence check fails, the algorithm outputs 0.

6. The algorithm checks if c_X is a commitment from the square number with SQR_2 by the algorithm $vSQR$. If SQR_2 is an invalid proof, the algorithm outputs 0.

7. The algorithm checks if Y is greater than 0. If the check fails, the algorithm outputs 0. Otherwise, the algorithm outputs 1.

In the **Prove** algorithm, we make the signature change every time through the randomizable signature scheme. As for the commitment, since we provide h_1, the prover can change the commitment C in each proof generation.

Note that the commitment is generated by the TTP with **Sign** algorithm only once and the prover can use it to generate different proofs for different ranges. So we can claim that the TTP authentication is offline. The following two tables; Table 3.2 and Table 3.3, are the summary of the procedure about **Prove** algorithm and the **Verify** algorithm.

Table 3.2: Produce AZKRP proof protocol

Prover	TTP
	$C^* = g^m \cdot h^{r^*}$, where $r^* \xleftarrow{R} \in \mathbb{Z}_N$ $M = e(C^*, y)$, $Sign(M, \Sigma) \rightarrow S$
Randomize signature S $C = C^* \cdot h_1^{r_{temp}} = g^m \cdot h^r$ $c = e(C, t_1) = e(g_1, t_1)^m \cdot e(h_1, t_1)^r = u^m \cdot v^r$ $c_1 = c \cdot u^{-a+1} = u^{m-a+1} \cdot v^r$ $c_2 = c^{-1} \cdot u^{b+1} = u^{b-m+1} \cdot v^{-r}$ $c_{EL} = c_1^{b-m+1} \cdot v^{r_{EL}}$, $r_{EL} \xleftarrow{R} \mathbb{Z}_N$ $EL \xleftarrow{R} SS(b-m+1, -r, r_{EL}, u, v, c_1, v, c_2, c_{EL})$ $c_{SQR_1} = (c_{EL})^{\omega^2} \cdot v^{r_{SQR_1}}$, $r_{SQR_1} \xleftarrow{R} \mathbb{Z}_N$ $SQR_1 \xleftarrow{R} SQ(\omega^2, r_{SQR_1}, c_{EL}, v, c_{SQR_1})$ $X = \delta^2$, $\delta \xleftarrow{R} \mathbb{Z}_N$ $Y = \omega^2 \cdot (m-a+1) \cdot (b-m+1) - X$ $r_Y = \omega^2((b-m+1) \cdot r + r_{EL}) + r_{SQR_1} - r_X$, $r_X \in \mathbb{Z}_N$ $c_X = u^X \cdot v^{r_X}$ $c_Y = u^Y \cdot v^{r_Y}$ $SQR_2 \xleftarrow{R} SQ(X, r_X, u, v, c_X)$ $\pi = \{a, b, C, S, c_{EL}, c_{SQR_1}, c_X, c_Y, EL, SQR_1, Y, SQR_2\}$	

Table 3.3: Verify AZKRP proof protocol

Input $\pi = \{a, b, C, S, c_{EL}, c_{SQR_1}, c_X, c_Y, EL, SQR_1, Y, SQR_2\}$
1. Calculate $M' = e(C, y)$
2. Verify S with (M', Σ)
3. Calculate $c = e(C, t_1)$
4. Calculate $c_1 = c \cdot u^{-a+1} = u^{m-a+1} \cdot v^r$
5. Calculate $c_2 = c^{-1} \cdot u^{b+1} = u^{b-m+1} \cdot v^{-r}$
6. Verify $vSS(EL, u, v, c_1, v, c_2, c_{EL})$
7. Verify $vSQ(SQR_1, c_{EL}, v, c_{SQR_1})$
8. Verify $c_{SQR_1} = c_X \cdot c_Y$
9. Verify $vSQ(SQR_2, u, v, c_X)$
10. Verify $Y > 0$

Chapter 4 Security Proof

In this chapter, because we integrate randomizable signature scheme with ZKRP we have to prove all ZKRP again.

4.1 Correctness

In this section, we will show the correctness of our construction. We show that if the secret value m lies in the range (a, b) and is authenticated by the TTP, the proof π can be accepted by the verifier. We check all **Verify** steps in our protocol as follow.

1. In the step 1 of **Verify**, the algorithm computes M' as follows.

$$\begin{aligned} M' &= e(C, y) \\ &= e(g^m \cdot h^r, t_2^x) \\ &= e(g_1^m \cdot g_2^m \cdot h_1^{r^* + r_{temp}} \cdot h_2^{r^*}, t_2^x) \\ &= e(g_2^m \cdot h_2^{r^*}, t_2^x) \end{aligned}$$

As for M which is signed by the TTP, its value can be calculate as follows.

$$\begin{aligned} M &= e(C^*, y) \\ &= e(g^m \cdot h^{r^*}, t_2^x) \\ &= e(g_1^m \cdot g_2^m \cdot h_1^{r^*} \cdot h_2^{r^*}, t_2^x) \\ &= e(g_2^m \cdot h_2^{r^*}, t_2^x) \end{aligned}$$

The verifier takes the signature public key y and the commitment secret information value M' to verify the signature S. If $M' = M$, then the signature should be valid for M'.

2. Since c_1, c_2 are derived from the same equations with the prover, the verifier can have the same c_1, c_2 with the prover.

3. If Boudot scheme stands, EL can show the secret equivalence of two commitments c_2 and c_{EL} where the secret is $b - m + 1$.

4. If Boudot scheme stands, SQR_1 can show that the commitments c_{SQR_1} comes from a square number ω^2.

5. Now we show that c_{SQR_1} is equal to $c_X \cdot c_Y$ as follows.

$$
\begin{aligned}
c_{SQR_1} &= (c_{EL})^{\omega^2} \cdot v_1^{r_{SQR_1}} \\
&= (c_1^{(b-m+1)} \cdot v_1^{r_{EL}})^{\omega^2} \cdot v_1^{r_{SQR_1}} \\
&= ((u_1^{(m-a+1)} \cdot v_1^{r})^{(b-m+1)} \cdot v_1^{r_{EL}})^{\omega^2} \cdot v_1^{r_{SQR_1}} \\
&= u_1^{\omega^2 \cdot (m-a+1) \cdot (b-m+1)} \cdot v_1^{\omega^2 \cdot ((b-m+1) \cdot r + r_{EL}) + r_{SQR_1}} \\
&= u_1^{X+Y} \cdot v_1^{r_X + r_Y} \\
&= c_X \cdot c_Y
\end{aligned}
$$

6. If Boudot scheme stands, SQR_2 can show that the commitments c_X comes from a square number ω^2.

7. Since $c_{SQR_1} = c_X \cdot c_Y$, it implies that $X+Y$ is equal to $\omega^2 \cdot (m-a+1) \cdot (b-m+1)$. Therefore, if $Y > 0$, we can know that $\omega^2 \cdot (m-a+1)(b-m+1) - X > 0$. Since ω^2 and $X = \delta^2$ are both greater than zero, $(m-a+1)(b-m+1)$ is also greater than 0. In other word, $a \leq m \leq b$.

4.2 Soundness

Generally speaking, in a ZKP scheme, soundness is a property that it is computationally impossible for someone to generate a valid proof if it has no valid secret. As described above, range check is not a hard problem since it is trivial to pick a value which is in the

range. So to prove the soundness of our scheme, we focus on the problem that the prover cannot generate a valid signature for the forged secret and the prover cannot generate a valid signature with the secret that is not in the range value.

In our scheme, we do not sign the secret directly since the signature verification requires the message itself, and we do not want to release the message. We do not sign the commitment directly either since we want to provide commitment randomization feature. Instead, we map the commitment to $\mathbb{G}_T$ as M and sign M.

Theorem 1. *If the probability that the prover can use a forged secret without authentication to pass the verification where its commitment can be mapped to M and the probability that the prover can use a secret that is not in the range to pass the verification are negligible, then we can sat that our scheme has the soundness property.*

Lemma 1. *Suppose the signature scheme is secure, the prover cannot forge a valid signature. The probability that the prover can use a forged secret to pass the verification where its commitment can be mapped to M is negligible.*

Proof. Because of the assumption that the signature is secure, the probability that the malicious prover can make a valid signature to generate a commitment is negligible. Therefore, the only method that the malicious prover wants to forge a valid signature is generating a same value M by the different secret value and random value.

Suppose the prover can find (m_1, r_1) and (m_2, r_2), where m_1, m_2 are messages and r_1, r_2 are random numbers for the commitments, that can be mapped to the same M. That is,

$$\begin{aligned} M &= e(C, y) = e(C', y) \\ &= e(g_2^{m_1} h_2^{r_1}, t_2^x) = e(g_2^{m_2} h_2^{r_2}, t_2^x) \\ \Rightarrow \quad & e(g_2^{m_1+\alpha r_1}, t_2^x) = e(g_2^{m_2+\alpha r_2}, t_2^x) \\ \Rightarrow \quad & m_1 + \alpha r_1 = m_2 + \alpha r_2 \\ \Rightarrow \quad & \alpha = \frac{m_2 - m_1}{r_1 - r_2}. \end{aligned}$$

It means that the prover can derive the system wise secret α. However, α is kept secret by TTP. Thus, the probability that the prover can use a forged secret to pass the verification is negligible.

□

Lemma 2. *Suppose the signature scheme is secure, the prover cannot forge a valid signature. The probability that the prover can use a secret that is not in the range to pass the verification is negligible.*

Proof. If the prover uses the secret value m that is not in the range value $[a, b]$, where $(a, b, m \in \mathbb{N})$. There are two conditions as following;

1. $m < a < b \Leftrightarrow m - a < 0\ ,\ b - m > 0 \Leftrightarrow m - a + 1 < 1\ ,\ b - m + 1 > 0.$

   (a) $m - a + 1 = 0 \Leftrightarrow (m - a + 1) \cdot (b - m + 1) = 0$

   According to the **Prove** algorithm Step7,

   $$Y = \omega^2 \cdot (m - a + 1) \cdot (b - m + 1) - X$$

   $(m - a + 1) \cdot (b - m + 1) = 0 \Leftrightarrow Y = -X$

   $\because X = \delta^2,\ \ i.e.\ X > 0\ \ \therefore Y < 0.$

   (b) $m - a + 1 < 0 \Leftrightarrow (m - a + 1) \cdot (b - m + 1) < 0$

   According to the **Prove** algorithm Step7,

   $$Y = \omega^2 \cdot (m - a + 1) \cdot (b - m + 1) - X$$

   $(m - a + 1) \cdot (b - m + 1) < 0 \Leftrightarrow Y < 0$

2. $a < b < m \Leftrightarrow m - a > 0\ ,\ b - m < 0 \Leftrightarrow m - a + 1 > 0\ ,\ b - m + 1 < 1.$

   (a) $b - m + 1 = 0 \Leftrightarrow (m - a + 1) \cdot (b - m + 1) = 0$

   According to the **Prove** algorithm Step7,

   $$Y = \omega^2 \cdot (m - a + 1) \cdot (b - m + 1) - X$$

$$(m-a+1)\cdot(b-m+1)=0 \Leftrightarrow Y=-X$$

$$\because X=\delta^2,\ \ i.e.\ X>0\ \ \therefore Y<0.$$

(b) $b-m+1<0 \Leftrightarrow (m-a+1)\cdot(b-m+1)<0$

According to the **Prove** algorithm Step7,

$$Y=\omega^2\cdot(m-a+1)\cdot(b-m+1)-X$$

$$(m-a+1)\cdot(b-m+1)<0 \Leftrightarrow Y<0$$

Because of $Y<0$, the **Verify** algorithm Step7 will be fail. Therefore, the probability that the prover can use a secret that is not in the range value to pass the verification is negligible.

□

According to the Proofs about Lemma 1. and Lemma 2. , we ensure that our scheme satisfied the soundness property.

4.3 Zero-Knowledge

In our scheme, we sign the commitment instead of the prover's secret and we assume that our randomizable signature protocol is secure. Furthermore, the commitment $C^* =$

$g^m \cdot h^{r^*}$ generated by TTP is a kind of Fujisaki-Okamoto commitment scheme. Since Fujisaki-Okamoto commitment scheme is a kind of zero-knowledge proof. If we can say that our scheme still satisfied the Fujisaki-Okamoto commitment scheme, then we have the zero-knowledge property.

Theorem 2. *Suppose the signature scheme is secure, the prover cannot forge a valid signature. If the commitment c is a kind of Fujisaki-Okamoto commitment scheme, and no one can distinguish the EL proof, and the SQR proof, then our scheme has the zero-knowledge property.*

Lemma 3. *The commitment c used to produce zero-knowledge range proof is a kind of Fujisaki-Okamoto commitment scheme and satisfies the zero-knowledge property.*

Proof. The commitment $c = e(C, t_1)$ is satisfied the Fujisaki-Okamoto commitment binding property refer to the Appendix A. The commitment $c = e(C, t_1)$ releases no information about the message m since for a commitment generated from (m, r), with the secret α, anyone can derive another pair (m', r') that can generate the same commitment as follows.

$$c = e(C, t_1) = e({g_1}^m \cdot {h_1}^r, t_1) = e({g_1}^m, t_1) \cdot e({h_1}^r, t_1) = e(g_1, t_1)^m \cdot e(h_1, t_1)^r$$

$$\exists\, m', r' \; s.t.$$

$$c = e({g_1}^m, t_1) \cdot e({h_1}^r, t_1) = e({g_1}^{m'}, t_1) \cdot e({h_1}^{r'}, t_1)$$

$$\Rightarrow e(g_1, t_1)^m \cdot e(h_1, t_1)^r = e(g_1, t_1)^{m'} \cdot e(h_1, t_1)^{r'}$$

$$\Rightarrow e(g_1, t_1)^{m-m'} = e(h_1, t_1)^{r'-r}$$

$$\Rightarrow e(g_1, t_1)^{m-m'} = e({g_1}^{\alpha}, t_1)^{r'-r}$$

$$\Rightarrow e(g_1, t_1)^{m-m'} = e(g_1, t_1)^{\alpha \cdot (r'-r)}$$

$$\Rightarrow m - m' = \alpha \cdot (r' - r)$$

$$\Rightarrow r' = r + \frac{m - m'}{\alpha}$$

The commitment c can be any messages so there is no information about m to be released. Therefore, the commitment c is satisfied the zero-knowledge property. □

Lemma 4. *The EL proof is a kind of zero-knowledge proof, and the commitments in the EL proof are satisfied the Fujisaki-Okamoto commitment. The EL proof releases no information about the secret m. anyone can derive another EL' proof such that the commitments of the two proofs are same.*

Proof. The EL proof where $EL = SS(m, r_1, r_2, e(g_1, t_1), e(h_1, t_1), e(g_1, t_1), e(h_1, t_1), A, B)$ is a kind of zero-knowledge proof. The EL proof releases no information about the secret

m since for the commitment generated from (m, r), with the secret α, anyone can derive another pairs (m', r'_1), (m', r'_2) that can generate the same commitments as follows.

$$A = e(g_1, t_1)^m \cdot e(h_1, t_1)^{r_1},\ B = e(g_1, t_1)^m \cdot e(h_1, t_1)^{r_2}$$

$$EL = SS(m, r_1, r_2, e(g_1, t_1), e(h_1, t_1), e(g_1, t_1), e(h_1, t_1), A, B)$$

$$\exists\, m', r'_1, r'_2\ s.t.$$

$$A = e(g_1, t_1)^m \cdot e(h_1, t_1)^{r_1} = e(g_1, t_1)^{m'} \cdot e(h_1, t_1)^{r'_1},$$

$$B = e(g_1, t_1)^m \cdot e(h_1, t_1)^{r_2} = e(g_1, t_1)^{m'} \cdot e(h_1, t_1)^{r'_2}$$

$$EL' = SS(m', r'_1, r'_2, e(g_1, t_1), e(h_1, t_1), e(g_1, t_1), e(h_1, t_1), A, B)$$

The commitments A and B are kind of Fujisaki-Okamoto scheme, hence both of them have the zero-knowledge property. The verifier do the vEL algorithm to check the proof is valid or not.

$$vEL \xleftarrow{R} vSS(EL, e(g_1, t_1), e(h_1, t_1), e(g_1, t_1), e(h_1, t_1), A, B)$$

$$vEL \xleftarrow{R} vSS(EL', e(g_1, t_1), e(h_1, t_1), e(g_1, t_1), e(h_1, t_1), A, B)$$

Both verification will be passed if Boudot scheme stands. Thus, no one can distinguish the EL proof and the EL' proof with the same commitments A, B. The EL proof does not reveal any information about secret m. Therefore, the EL proof is satisfied zero-knowledge property. □

Lemma 5. *The SQR proof is a kind of zero-knowledge proof, and the commitment in the SQR proof is satisfied the Fujisaki-Okamoto commitment. The SQR proof releases no information about the secret m. anyone can derive another SQR' proof such that the commitment of the two proofs is same.*

Proof. The SQR proof where $SQR = SQ(m^2, r, e(g_1, t_1), e(h_1, t_1), A)$ is a kind of zero-knowledge proof. The SQR proof releases no information about the secret m since for the commitment generated from (m, r), with the secret α, anyone can derive another pairs (m', r') that can generate the same commitment as follows.

$$A = e(g_1, t_1)^{m^2} \cdot e(h_1, t_1)^r$$

$$SQR = SQ(m^2, r, e(g_1, t_1), e(h_1, t_1), A)$$

$$\exists\ (m')^2, r'\ s.t.$$

$$A = e(g_1, t_1)^{m^2} \cdot e(h_1, t_1)^r = e(g_1, t_1)^{(m')^2} \cdot e(h_1, t_1)^{r'}$$

$$SQR' = SQ((m')^2, r', e(g_1, t_1), e(h_1, t_1), A)$$

The commitment A is kind of Fujisaki-Okamoto scheme, hence A has the zero-knowledge property. The verifier do the $vSQR$ algorithm to check the proof is valid or not.

$$vSQR \overset{R}{\leftarrow} vSQ(SQR, e(g_1, t_1), e(h_1, t_1), A)$$

$$vSQR \overset{R}{\leftarrow} vSQ(SQR', e(g_1, t_1), e(h_1, t_1), A)$$

Both verification will be passed if Boudot scheme stands. Thus, no one can distinguish the SQR proof and the SQR' proof with the same commitment A. The SQR proof does not reveal any information about secret m. Therefore, the SQR proof is satisfied zero-knowledge property. □

According to the Lemma 3. , Lemma 4. , and Lemma 5. we ensure that our scheme satisfied the zero-knowledge property.

Chapter 5 Efficiency Analysis

In this chapter, we evaluate our scheme efficiency and compare with others range proof scheme. Compare to [20], Although we have a new algorithm **Sign** generated by TTP, we only increase a little cost to generate our scheme about **Sign** algorithm and produce the commitment about **Prove** and verify the signature about **Verify**. However our scheme is more secure and meaningful than [20], since it does not guarantee the prover's secret is real value about the prover.

Compare to [5, 6], our scheme's bounded range value is an arbitrary constant rather than a logarithmic size range as the following Table 5.1 (where n is the bit length of the range) and we still keep the zero-knowledge range proof security. Although our scheme efficiency is less than [5, 6], We have more flexible than [5, 6]. Conclusively, our scheme is more secure and meaningful nothing but increase a little cost in our algorithm.

Table 5.1: Efficiency of our range proof

	Range Form	Prove Size
our scheme	Arbitary	26n
[20]	Arbitary	21n
[5, 6]	$[0, 2^{n-1}]$	$2 \cdot log_2(n)$

Chapter 6 Conclusion

In this paper, we propose an authenticated zero-knowledge range proof scheme. It solves the generic range proof problem that the range checking is simple. We introduce a TTP in our scheme and make TTP to authenticate the message before generating a range proof. We also make the proof can be randomized so that no outsiders can tell what kind of messages are used to generate proofs. We believe that our AZKRP scheme is the first range proof scheme that can be applied to the real scenario.

6.1 Future Work

In the future, we will try to make the multiple attribute verification range proof. That is, Alice wants to convince verifier that her's age, height, and weight are in the given range values but she does not want to leak any information about her's age, height, and weight. Generally, there are lots of requirement about Model. Not only age, but also weight, height. Then we can use the multiple attribute verificaion range proof.

The other future work is that the prover generates a zero-knowledge range proof without any range value in the **Prove** algorithm only in the **Verify** algorithm. On the other words, the prover generates a zero-knowledge range proof with the prover's secret without any range value, and verifiers can verify if the proof is in the individual given range values. That is, if Alice wants to participate lots of organizations competition, then she can generate a zero-knowledge range proof, and public to those organizations to verify whether she has qualifications or not.

References

[1] M. Blum, P. Feldman, and S. Micali. Non-interactive zero-knowledge and its applications. In Providing Sound Foundations for Cryptography: On the Work of Shafi Goldwasser and Silvio Micali, pages 329–349. 2019.

[2] D. Boneh, E. Boyle, H. Corrigan-Gibbs, N. Gilboa, and Y. Ishai. Zero-knowledge proofs on secret-shared data via fully linear pcps. In A. Boldyreva and D. Micciancio, editors, Advances in Cryptology – CRYPTO 2019, pages 67–97, Cham, 2019. Springer International Publishing.

[3] D. Boneh, E.-J. Goh, and K. Nissim. Evaluating 2-dnf formulas on ciphertexts. In TCC, pages 325–341, 2005.

[4] F. Boudot. Efficient proofs that a committed number lies in an interval. In B. Preneel, editor, Advances in Cryptology — EUROCRYPT 2000, pages 431–444. Springer Berlin Heidelberg, 2000.

[5] B. Bünz, J. Bootle, D. Boneh, A. Poelstra, P. Wuille, and G. Maxwell. Bulletproofs:

Short proofs for confidential transactions and more. Cryptology ePrint Archive, Report 2017/1066, 2017. https://eprint.iacr.org/2017/1066.

[6] B. Bünz, J. Bootle, D. Boneh, A. Poelstra, P. Wuille, and G. Maxwell. Bulletproofs: Short proofs for confidential transactions and more. In 2018 IEEE Symposium on Security and Privacy (SP), pages 315–334, May 2018.

[7] J. Camenisch and A. Lysyanskaya. Signature schemes and anonymous credentials from bilinear maps. In M. Franklin, editor, Advances in Cryptology – CRYPTO 2004, pages 56–72, Berlin, Heidelberg, 2004. Springer Berlin Heidelberg.

[8] R. Chaabouni, H. Lipmaa, and B. Zhang. A non-interactive range proof with constant communication. In A. D. Keromytis, editor, Financial Cryptography and Data Security, pages 179–199, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.

[9] A. De Santis, S. Micali, and G. Persiano. Non-interactive zero-knowledge proof systems. In C. Pomerance, editor, Advances in Cryptology — CRYPTO '87, pages 52–72, Berlin, Heidelberg, 1988. Springer Berlin Heidelberg.

[10] E. Fujisaki and T. Okamoto. Statistical zero knowledge protocols to prove modular polynomial relations. In B. S. Kaliski, editor, Advances in Cryptology — CRYPTO '97, pages 16–30, Berlin, Heidelberg, 1997. Springer Berlin Heidelberg.

[11] O. Goldreich. Foundations of Cryptography: Basic Tools. Cambridge University Press, USA, 2000.

[12] S. Goldwasser, S. Micali, and C. Rackoff. The knowledge complexity of interactive proof-systems. In Proceedings of the Seventeenth Annual ACM Symposium on Theory of Computing, STOC '85, page 291–304, New York, NY, USA, 1985. Association for Computing Machinery.

[13] S. Goldwasser, S. Micali, and R. L. Rivest. A digital signature scheme secure against adaptive chosen-message attacks. SIAM Journal on Computing, 17(2):281–308, 1988.

[14] T. Koens and C. Ramaekers. Efficient zero-knowledge range proofs in ethereum. 2017.

[15] A. Kosba, A. Miller, E. Shi, Z. Wen, and C. Papamanthou. Hawk: The blockchain model of cryptography and privacy-preserving smart contracts. In 2016 IEEE Symposium on Security and Privacy (SP), pages 839–858, May 2016.

[16] P. McCorry, S. Shahandashti, and F. Hao. A smart contract for boardroom voting with maximum voter privacy. 01 2017.

[17] D. Pointcheval and O. Sanders. Short randomizable signatures. In K. Sako, edi-

tor, Topics in Cryptology - CT-RSA 2016, pages 111–126, Cham, 2016. Springer International Publishing.

[18] D. Pointcheval and O. Sanders. Reassessing security of randomizable signatures. In N. P. Smart, editor, Topics in Cryptology – CT-RSA 2018, pages 319–338, Cham, 2018. Springer International Publishing.

[19] Y. Tao, X. Wang, and R. Zhang. Short zero-knowledge proof of knowledge for lattice-based commitment. In J. Ding and J.-P. Tillich, editors, Post-Quantum Cryptography, pages 268–283, Cham, 2020. Springer International Publishing.

[20] Y. C. Tsai, R. Tso, Z. Liu, and K. Chen. An improved non-interactive zero-knowledge range proof for decentralized applications. In 2019 IEEE International Conference on Decentralized Applications and Infrastructures (DAPPCON), pages 129–134, April 2019.

[21] Y. Wang and A. Kogan. Designing confidentiality-preserving blockchain-based transaction processing systems. International Journal of Accounting Information Systems, 30:1 – 18, 2018. 2017 Research Symposium on Information Integrity & Information Systems Assurance.

[22] L. Xu, N. Shah, L. Chen, N. Diallo, Z. Gao, Y. Lu, and W. Shi. Enabling the sharing economy: Privacy respecting contract based on public blockchain. In Proceedings of

the ACM Workshop on Blockchain, Cryptocurrencies and Contracts, pages 15–21, 2017.

Appendix A — Fujisaki-Okamoto Commitment

The primary Fujisaki-Okamoto Commitment was based on the discrete-logarithm problem and integer factorization problem. It allows everyone to commit a chosen value while keeping it secrets to others. and the prover has the ability to reveal the committed value later [11]. However, in our scheme, we integrate bilinear group map function. Therefore, we give a proof that our scheme is still satisfying the binding property of Fujisaki-Okamoto Commitment.

Lemma 6 (Fujisaki-Okamoto Commitment with bilinear group map function satify binding property)**.** *It is hard to find* $\theta_1, \theta_2, \nu_1, \nu_2 \in \mathbb{Z}_N$ *and satisfy* $\eta = u^{\theta_1} \cdot v^{\nu_1}$ *and* $\eta = u^{\theta_2} \cdot v^{\nu_2}$*, where* $u = e(g_1, t_1), v = e(h_1, t_1)$ *and* $(\theta_1, \nu_1) \neq (\theta_2, \nu_2)$

Proof. Assume $\theta_1, \theta_2, \nu_1, \nu_2 \in \mathbb{Z}_N$, $\eta = u^{\theta_1} \cdot v^{\nu_1}$ and $\eta = u^{\theta_2} \cdot v^{\nu_2}$, where $(\theta_1, \nu_1) \neq (\theta_2, \nu_2)$, then

$$e(g_1, t_1)^{\theta_1} \cdot e(h_1, t_1)^{\nu_1} = e(g_1, t_1)^{\theta_2} \cdot e(h_1, t_1)^{\nu_2}$$

$$e(g_1, t_1)^{\theta_1 - \theta_2} = e(h_1, t_1)^{\nu_2 - \nu_1}$$

$$e(g_1, t_1)^{\theta_1 - \theta_2} = e({g_1}^{\alpha}, t_1)^{\nu_2 - \nu_1}$$

$$e(g_1, t_1)^{\theta_1 - \theta_2} = e(g_1, t_1)^{\alpha \cdot (\nu_2 - \nu_1)}$$

$$(\theta_1 - \theta_2) = \alpha \cdot (\nu_2 - \nu_1)$$

$$\frac{(\theta_1 - \theta_2)}{(\nu_2 - \nu_1)} = \alpha$$

Therefore, If we can find $\theta_1, \theta_2, \nu_1, \nu_2$ s.t. $\frac{(\theta_1 - \theta_2)}{(\nu_2 - \nu_1)} = \alpha$. It means that we can solve the discrete logarithm problem.

□

Appendix B — Inner Forge and translation problem

Since the TTP's public key only for the prover that are in the same group, outsider forge can not achieve. However, those provers in the same group want to forge other prover's commitment secret information value M still can not achieve. Even if they can forge the value $M' = M \cdot e(g^{m'} \cdot h^{r'}, y)$, the signature S signed by the TTP is generated by M, and no one can get the TTP's secret key x to forge a new signature S' which is signed by the M'. Therefore the inner forge can not achieve.

If the verifier uses the other range value $[a', b']$ to verify the ZKRP which is generated by the secret m and range value $[a, b]$. when the verifier generate the commitment c, c_1, c_2 the value of $m - a + 1$ and $b - m + 1$ will change. Hence, $(m - a + 1) \cdot (b - m + 1)$ may less than zero. the algorithm **verify** may fail. In addition, if the prover translate the secret value m and range value $[a, b]$ (for example; $a + 10 < m + 10 < b + 10$) to forge a new range proof, but the range is public, and the **Sign** algorithm does not use range value $[a, b]$

as input. Therefore, the original signature is still the secret value m. The verifier check the signature in the step 1 will be failed.