

Some Low Round Zero-Knowledge Protocols

Hongda Li^{1,2,3}, Peifang Ni^{1,2,3}, and Dongxue Pan^{1,2,3}

1. State Key Laboratory of Information Security, Institute of Information Engineering, CAS
2. School of Cyber Security, University of Chinese Academy of Sciences
3. Data Assurance and Communication Security Research Center, CAS

lihongda@iie.ac.cn nipeifang@iie.ac.cn pandongxue@iie.ac.cn

Abstract. In this paper, we focus on zero-knowledge protocols for NP with low round complexity under the augmented black-box simulation technique, in which the simulator has access to the verifier’s secret information, and obtain positive results on 3-round zero-knowledge proofs and 2-round zero-knowledge arguments for NP and 2-round zero-knowledge proofs for QNR. More precisely, our contributions are five-fold: (i) we propose the notion of generalized claw-free function and the notion of trapdoor generalized claw-free function, and then we show a construction of trapdoor generalized claw-free function under the discrete logarithm assumption and the knowledge of exponent assumption, (ii) we propose the notion of completely extractable bit-commitment and give a construction of it from trapdoor generalized claw-free functions, (iii) we present a 3-round zero-knowledge proof for NP based on the completely extractable bit-commitment schemes and Yao’s garbling circuit technique, (iv) we show a 2-round zero-knowledge argument for NP based on indistinguishable obfuscator, (v) we transform the basic 2-round honest verifier zero-knowledge proof protocol for quadratic non-residue into a 2-round zero-knowledge proof protocol.

Keywords: zero-knowledge proof (argument), augmented black-box simulation, generalized claw-free function, completely extractable commitment scheme, garbled circuits

1 Introduction

Zero-knowledge (ZK) protocol, an interactive proof by the prover P and verifier V , is formally introduced by Goldwasser, Micali and Rackoff [34]. An interactive proof is considered zero-knowledge if P can convince V of the correctness of the statement, while the verifier V learns nothing beyond the fact that the statement is true. ZK protocol can be formalized in two ways, zero-knowledge *proof* (denoted by ZKP) and zero-knowledge *argument* (denoted by ZKA), depending on whether the power of the prover is restricted. ZK protocol has been the subject of intensive study since it was introduced. Specially, Goldreich, Micali and Wigderson proved that every language in NP has a zero knowledge proof if there exist one-way functions [33]. There have been a lot of positive results for constructing ZKP or ZKA satisfying some additional properties, such as constant-round ZK protocol [24, 28, 9, 45, 11] and concurrent or resettable ZK protocol [16, 17, 44, 23, 19, 18].

Zero-knowledge requires in essence whatever the verifier V can compute while interacting with the prover P it can compute by itself without interacting with P . It is formally defined in [34] by requiring that, for any (malicious) verifier V^* , there exists a simulator (PPT algorithm),

which receives only the common input of both the prover and the verifier, can output a random distribution that is indistinguishable from the view of V^* (the real conversation). Initially, all known zero-knowledge protocols used black-box simulator, an universal algorithm which uses V^* as a black-box, to simulate V^* 's view (so called black-box zero-knowledge, BBZK). It is known that 4-round BBZK arguments for NP exist under one-way functions [24, 9], while constant-round BBZKP for NP are presented in [28, 45]. The original definition of ZK in [34] is not closed under sequential composition [30]. Glodreich and Oren presented a more robust definition, called auxiliary-input zero-knowledge, which permits simulator obtain the verifier's auxiliary and is closed under sequential composition [32].

The round complexity of ZK protocols is the main measure of the efficiency. From the practical and theoretical viewpoint, it is desirable to minimize the round complexity of ZK protocols. Glodreich and Oren first proved that there does not exist 2-round auxiliary-input ZK protocols system for a language outside of BPP [32]. The black-box simulator needs to use rewinding technique, so black-box ZK protocols need more rounds. It is known that black-box ZK protocols with at least 4-round exist for NP [24, 9, 28, 45], while Glodreich and Krawczyk [30] showed that 3-round black-box ZK protocols (proofs or arguments) do not exist for the language outside of BPP. They also proved that constant-round public-coin ZK protocol with negligible error probability does not exist for nontrivial language. In addition, Canetti et al. showed there is no constant-round black-box concurrent zero-knowledge protocol for NP [17]. Katz proved that NP-complete languages do not have 4-round black-box zero-knowledge proofs assuming the polynomial hierarchy does not collapse [38].

Although the definitions in [34, 32] have less restriction on PPT simulator, it was not until 2001 that Barak presented a kind of non-black-box simulation method, where the simulator utilizes V^* 's strategy description, and then showed a constant-round bounded concurrent zero-knowledge argument [1]. It has been known from the result of [17] that such protocol cannot be zero-knowledge under black-box simulation. Since then, a number of non-black-box zero knowledge (NBBZK) protocols have been proposed [6, 43, 7, 35, 44, 23, 19, 20, 18]. Non-black-box simulation technique does not use rewinding technique, and so can reduce the round complexity of ZK protocols. Specially, Bitansky and Paneth presented a 4-round resettably-sound ZK protocol for NP [14], and Barak, Lindell and Vadhan even showed a 2-round ZKP system for a problem outside of BPP under "Knowledge of Exponent Assumption" (KEA) [7]. However, there are still some negative results. Barak et al. proved that there does not exist a 2-round ZKP with perfect completeness for any NP-complete language under a plausible assumption [7]. This leads to the question: do there exist 2-round ZK protocols for NP under some standard assumptions?

Another kind of non-black-box ZK protocol is based on special non-standard knowledge assumptions. Bellare and Palacio showed a 3-round non-black-box ZKP under KEA [11] and Lepinski also showed one under "Proof of Knowledge Assumption" (POKA) [40]. Abstracting from such specific knowledge assumptions, Canetti et al. proposed the notion of extractable one-way function and constructed a 3-round ZK protocol from it [15]. Bitansky et al. constructed a 3-round ZKA for NP based on extractable one-way functions [4]. Such extractability for one-way function seems to contradict the existence of indistinguishable obfuscator. Bitansky et al. proved that the existence of indistinguishability obfuscators for a certain class of circuit means that there do not exist extractable one-way functions for non-uniform adversary [3]. Recently, under certain assumptions on program obfuscation (including the existence of (sub-)exponentially secure indistinguishability obfuscation), Kalai et al. proved that there does

not exist a public-coin constant round ZKP even under non-black-box simulation [39] and Fleischhacker et al. gave negative result that 3-round non-black-box ZKP does not exist for languages outside of BPP [26]. Therefore, the existence of 3-message ZK protocols under non-black-box simulation is still unsolved.

The ZK is formally defined by requiring there exists a simulator to output an indistinguishable conversation. The stronger simulator means the weaker zero knowledge property, so enhancing simulator or restricting either prover or verifier help us to achieve more efficient protocols. Dwork and Naor et al. proposed the notion of weak ZK [21] which relaxes ZK by allowing the simulator to depend on the distinguisher to distinguish the simulated conversation from the real one, and Bitansky and Paneth gave a construction of 3-round weak ZKA based on point obfuscation and Yao’s garbled circuit technique [13]. Pass weakens ZK by permitting simulator to run in super-polynomial time and obtains 3-round ZKA [42]. While Bitansky et al. constructed a 3-round ZKA in the restricted adversary models, where either the prover or the verifier is assumed to be uniform [3, 2]. Dwork and Stockmeyer even got 2-round zero-knowledge protocols in the model where the prover has bounded resources [22].

Recently, Li et al. presented augmented black-box simulation technique (the corresponding simulator is called ABB simulator) which allows the black-box simulator to access verifier’s secret information [41]. To simulate the real interaction, the augmented black-box simulator runs the verifier’s strategy V . Specifically, the ABB simulator not only receives V ’s output but also gets its secret information used to compute the output of V . The ABB simulator, being allowed to receive V ’s secret information, is in fact a mental algorithm and so the ABB simulation is just a mental game. However, the verifier can play this mental game by himself, and so the simulated conversation outputted by the ABB simulator can be computed by the verifier. Therefore, the fact that the output of the ABB simulator is indistinguishable from the real conversation means the verifier, with the ABB simulator, can obtain all that it can get from interacting with the prover, that is, interacting with prover will never improve the verifier’s computational power except for a negligible probability.

The original meaning of zero knowledge is in essence that interacting with the prover does not improve the verifier’s computational power. So, using ABB simulator does not weaken ZK property. For the above motivations, we continue to work on the existence of 3-round and 2-round ZK protocols for any NP language under augmented black-box simulation.

1.1 Our Contributions

Generalized Claw-Free Functions and Completely Extractable Bit-Commitment. Our 3-round protocol adopts a new interactive proof model which depends on a special bit-commitment scheme, which is called as completely extractable bit-commitment. A bit commitment scheme is completely extractable if, after the commitment phase is completed, the receiver can correctly reveal the received commitment as 0 or 1. Motivated by the need to construct completely extractable bit-commitment schemes, we propose the notion of generalized claw-free functions (GCFF) and give an instance of construction. Based on GCFF, we finally succeed in constructing a completely extractable bit-commitment scheme, which is perfectly hiding and computationally binding.

In our 3-round ZKP for NP, the verifier uses a completely extractable bit-commitment scheme to commit to its challenge and the prover is a receiver.

3-Round ZKP for NP. We first provide a public-coin interactive proof for any $L \in NP$, which is 3-round and takes the classic form of “commit-challenge-reply”, by means of Yao’s

garbling circuit scheme. For any $L \in NP$, the corresponding NP-relation is R_L . Proving $x \in L$ is equivalent to proving there exists a w such that $R_L(x, w) = 1$. Assume $Gb(\cdot)$ is Yao’s garbling algorithm, and $C_{L,x} : \{0, 1\}^\ell \rightarrow \{0, 1\}$ is a circuit to compute $R_L(x, \cdot)$. The basic 3-round protocol is as follows: The prover first generates a garble circuit $(\widehat{C}_{L,x}, e, d) \leftarrow Gb(C_{L,x}, 1^n)$, and use $\widehat{C}_{L,x}$ as the commitment to R_L and keeps the encoding key e secret. After receiving $\widehat{C}_{L,x}$, the verifier selects and sends its challenge $\sigma \in \{0, 1\}$. Finally, the prover opens the encoding of witness w if $\sigma = 0$, and otherwise the prover opens the encoding key e . Obviously, the above protocol is ZK and has at most 1/2 error probability. Unfortunately, it is known that this protocol is not ZK under polynomial times parallel repetitions [30].

To obtain a 3-round ZKP by paralleling a basic protocol, our basic 3-round protocol uses a new model, simplified as “challenge-commit and reply”. In short, the new model changes the order of “prover-commit” and “verifier-challenge”, such that “verifier-challenge” is independent of “prover-commit”. This results in that “prover-commit” and “prover-reply” are completed in the same round. To ensure the soundness of the protocol, the verifier’s challenge must be hiding. Concretely, the first two round of the protocol is 2-round completely extractable (perfectly hiding and computationally binding) bit-commitment scheme, in which the verifier commits to its challenge. In the third round, the prover sends the commitment to the proving instance $(\widehat{C}_{L,x})$ and the reply to the verifier’s challenge. The soundness of the protocol is directly based on the perfectly hiding property of the completely extractable bit-commitment scheme, while the zero-knowledge under augmented black-box simulation comes from the privacy of Yao’s garbling scheme and the computationally binding property of the completely extractable bit-commitment scheme. Furthermore, the ABB simulator does not need to rewind the verifier’s algorithm, and so we can obtain a 3-round ZKP for $x \in L$ with a negligible error probability under ABB simulation by paralleling the basic 3-round protocol.

2-Round ZKA for NP. We obtain a 2-round ZKA for NP under augmented black-box simulation assuming indistinguishable obfuscator \mathcal{O} exists.

2-round interactive proof protocols for $x \in L$ require that the verifier first generates questions which the prover can answer correctly if and only if $x \in L$. The existence of non-interactive instance-dependent commitment scheme, which is perfectly binding when $x \in L$ and computationally (or statistically) hiding when $x \notin L$, implies the existence of 2-round honest verifier ZK protocols.

Let R_L be NP-relation of $L \in NP$. For any instance x , $R_{L,x}^1(u) = R_L(x, u)$ is a boolean function, and $R_{L,x}^1(u) \equiv 0$ if and only if $x \notin L$. In order to construct an instance-dependent commitment scheme, assume $R_{L,x}^0(u) \equiv 0$ and $|R_{L,x}^0(u)| = |R_{L,x}^1(u)|$. Using indistinguishable obfuscator \mathcal{O} , we can obtain an instance-dependent commitment scheme as follows: To commit to b , the sender sends $C = \mathcal{O}(R_{L,x}^b)$ to the receiver. When $x \notin L$, the computationally hiding property holds since $\mathcal{O}(R_{L,x}^0)$ and $\mathcal{O}(R_{L,x}^1)$ are indistinguishable. When $x \in L$, the scheme is perfectly binding since there exists u such that $\mathcal{O}(R_{L,x}^0)(u) \neq \mathcal{O}(R_{L,x}^1)(u)$. From it, 2-round honest verifier ZKA for $x \in L$ is as follows:

- The verifier V selects $\sigma \in \{0, 1\}$ randomly and sends $C = \mathcal{O}(R_{L,x}^\sigma)$ to the prover.
- The prover P computes $\sigma' = C(w_x)$, where w_x is a witness for $x \in L$, and sends σ' to V .
- The verifier accepts if and only if $\sigma' = \sigma$.

V only obtains what he already knows when it follows the protocol, but this does not hold for a malicious V^* . So the protocol is only honest-verifier ZK. If V does not honestly

generate C , $\sigma' = C(w_x)$ computed by P may be not uniformly distributed on $\{0, 1\}$ and cannot be simulated. In this case, the augmented black-box simulator will fail since it may not get $\sigma' = C(w_x)$ from C and V^* 's private information (random coins). That is, the protocol is not ZK even under ABB simulation.

A way to prevent V from cheating is to ask V 's message to be verifiable by P when $x \in L$. We parallel the above protocol n (security parameter) times: V sends (C_1, \dots, C_n) and P computes $\sigma'_i = C_i(w_x), i = 1, \dots, n$, and require that (C_1, \dots, C_n) must be verifiable by P when $x \in L$. To this end, V is asked to select $\sigma = \sigma_1 \dots \sigma_n, \delta = \delta_1 \dots \delta_n \in \{0, 1\}^n$, and then computes $C_i = \mathcal{O}(R_{L,x}^{\sigma_i}; r_i^{\delta_i})$, where $r_1 \dots r_n \leftarrow G(\sigma), r_i = (r_i^0, r_i^1)$, and G is a pseudorandom generator. Thus, the prover P can verify whether all (C_1, \dots, C_n) is correct after obtaining $\sigma_i = C_i(x, w) \in \{0, 1\}, i = 1, \dots, n$.

Our main idea, to transform the paralleled protocol into ZKA without adding the more interactions, is to have P , instead of opening $(\sigma'_1, \dots, \sigma'_n)$, sends a ‘‘random string’’ which satisfies the following two conditions: 1) V who generates (C_1, \dots, C_n) honestly can verifies it and 2) V^* who does not generate (C_1, \dots, C_n) honestly cannot distinguish it from a real random string. To this end, we need pseudorandom functions (PRF) $\{H_s\}_{s \in \mathcal{I}}$. If (C_1, \dots, C_n) passes the verification (P must get σ and δ), P randomly selects r and sends $(r, H_s(\delta))$ to V , where s is determined by r and σ . Obviously, $(r, H_s(\delta))$ satisfies the above two conditions.

2-Round ZKP for QNR. Glodreich and Oren first proved that there does not exist a 2-round auxiliary-input ZK protocol system for a language outside of BPP [32]. Barak et al. proved that there does not exist a 2-round ZKP with perfect completeness for any NP-complete language under a plausible assumption, but also showed a 2-round ZKP system for a problem outside of BPP under KEA [7]. We present a 2-round ZKP system for QNR by Yao’s garbling circuit scheme.

Recall the classic protocol of proving $x \in QNR$: V sends $w = r^2 x^b$ to the prover for randomly selected $r \in \mathbb{Z}_N, b \in \{0, 1\}$, and then P gets b from w and returns b to V . It is known that the protocol is honest-verifier ZK. A method to transform the classic protocol into a zero knowledge argument is to add an interactive process for the verifier to prove that it was honest. We use a new model to construct 2-round zero knowledge argument, specifically, P sends b in ‘‘encryption’’ way such that 1) V following the protocol can verifies whether the message sent by P is correct and 2) V^* not following the protocol cannot distinguish the message sent by P from a real random string. In that case, V either accepts $x \in QNR$ (when V is honest), or gets nothing (when V is cheating) since P 's message is indistinguishable from a random string. As a result, this 2-round protocol is ZK.

To this end, we first modify the classic protocol as follows: V sends $w = r^2 x^b$ and a point-function I_α for randomly selected α . After receiving $w = r^2 x^b$ and I_α , P computes b, r from w and generates a garbled circuit \bar{C} computing $I(u) = I_\alpha(u \oplus r)$, and finally sends \bar{C} with the encoding of α (denoted by $\hat{\alpha}$) to V . Here, we use Yao’s garbling circuit scheme but with some changes (see subsection 2.4 and 6.2 for details) to prevent V from cheating. The encoding of $\alpha \oplus r$ can be derived from the encoding of α and r , so V can verify if $\bar{C}(\alpha \oplus r) = I_\alpha(\alpha) = 1$ when it generates w honestly. However, any malicious V^* who does not know r satisfying $w = r^2 x^b$ can not verify whether \bar{C} is correct. In short, only an honest verifier can verify whether \bar{C} is correct. Furthermore, if V is honest, the augmented black-box simulator can obtain r and b from V , and then can generate correct \bar{C} . But for malicious V^* , the augmented black-box simulator without r satisfying $w = r^2 x^b$ can randomly select r and

generate a indistinguishable garbled circuit \overline{C} . That is, the protocol is augmented black-box ZK.

1.2 Related Works

While 4-round zero-knowledge arguments for NP are known based on one-way functions [24, 9], the existence of 3-round zero-knowledge (with negligible soundness error) has been a long standing open problem.

The initial constructions of ZKP require the polynomial number of rounds and some works achieve the constant round ZK protocols for any NP language [24, 25, 28, 9, 45, 20, 18, 37]. Previously, [25, 9] showed the 4-round ZKA based on one way functions, Goldreich and Kahan presented 5-round ZKP for NP based on 2-round statistically-hiding commitments, while 4-round black-box zero-knowledge proofs for NP are believed to be impossible [38]. Recently, [10] gives the 4-round ZKP based on keyless multi-collision-resistant hash functions. A few works focus on the existence of 3-round ZK protocols under non standard assumptions. Lepinski showed a 3-round ZKP under “Proof of Knowledge Assumption” [40], Bellare and Palacio fixed the “Knowledge of Exponent Assumption” of [36] and presented 3-round [11], while Bitansky et al. constructed a 3-round ZKA for NP based on extractable one-way functions[4]. These knowledge assumptions ask that there exists an efficient extractor algorithm, by which simulator can have access to the secret coins of the verifier. Bitansky proved that these knowledge assumptions contradict to the existence of indistinguishable obfuscator [3]. Recently, Fleischhacker et al. proved 3-round non-black-box ZKPs do not exist for languages outside of BPP under certain assumptions on program obfuscation (including the existence of sub-exponentially secure indistinguishability obfuscation) [26].

Bitansky and Paneth proposed the notion of weak ZK and gave a construction of 3-round ZKA based on point obfuscation and Yao’s garbled circuit technique [12, 13]. Jawurek et al. provided a constant-round ZKA using Yao’s garbled circuit scheme [37]. Following [37], Ganesh et al. constructed a 3-round ZKP based on RE-OT (receiver equivocal OT protocol) in CRS model [31]. In recent, Bitansky, Kalai, and Paneth introduced a new notion of multi-collision resistance for keyless hash functions and constructed a general 3-round ZKA based on it [10].

Goldreich and Oren first proved that there does not exist 2-round auxiliary-input ZK protocols system for a language outside of BPP [32]. Barak et al. proved that there does not exist a 2-round ZKP with perfect completeness for any NP-complete language under a plausible assumption, but also showed a 2-round ZKP system for a problem outside of BPP under KEA [7]. [41] presented a 2-round ZKA for Exact Cover problem under the Decision Multilinear No-Exact-Cover Assumption.

1.3 About Our 3-Round ZKP

We stress that the impossibility result presented in [26] does not refute our positive result, because ABB simulation is in fact ruled out in [26].

Let 3-round interactive proof system for $L \notin BPP$ be as follows: the prover P first sends $\alpha = P_1(x, w; r)$, the verifier V responds with $\beta = V_1(x, \alpha; s)$ and then P answers with $\gamma = P_2(x, w, \beta; r)$. To argue their negative result, [26] compresses 3-round interactive proof system into 2-round interactive argument, and then shows that the ZK simulator of the 3-round protocol can be used by a malicious prover \overline{P}^* of the 2-round protocol to break the soundness.

In this 2-round protocol, the honest verifier \bar{V} is asked to send an obfuscated version of the verifier's next message function (denoted by $iO[\bar{V}]$), which contains a private key k of the pseudorandom function PRF_k used to compute the random coins of the verifier, to the prover, and the prover computes $\alpha = P_1(x, w; r)$, $\gamma = P_2(x, w, \beta = iO[\bar{V}](\alpha); r)$, and then sends α, γ . By the security assumption on program obfuscation, any PPT prover cannot extract k and thus the random coins of the verifier from the received obfuscated algorithm.

In order to utilize the ZK simulator of 3-round protocol, \bar{P}^* needs to interpret $iO[\bar{V}]$ as the description of a verifier's strategy of 3-round protocol and provide it (as auxiliary input aux) to the ZK simulator. For this purpose, [26] consider the following malicious verifier V^* of 3-round protocol: First, V^* receives as auxiliary input a circuit aux . Second, after receiving the first message α , V^* sends $\beta = aux(\alpha)$ to the prover which will return γ , and finally output (α, β, γ) . Thus, $iO[\bar{V}]$ is in fact the description of V^* 's strategy, and then \bar{P}^* can run the ZK simulator Sim for V^* . Notice that Sim cannot get $s = PRF_k(\alpha)$ by rewinding or interpreting $iO[V]$ and \bar{P}^* can not provide Sim with k (and thus $s = PRF_k(\alpha)$) when \bar{P}^* runs Sim with the received obfuscated algorithm $iO[V]$. [26] requires that Sim must simulate the real conversation without learning V^* 's random coins.

In addition, in [26] \bar{P}^* breaks the soundness as follows: executes Sim with $aux = iO[V]$, $(\alpha, \beta, \gamma) = Sim(x, aux)$, and then sends (α, γ) to the verifier. This means that \bar{P}^* can succeed only if \bar{V} , which knows k and s , cannot distinguish $Sim(x, aux)$ from the real conversation. Therefore, [26] requires that the random coins of the honest verifier is available to the distinguisher.

In [26], the non-uniform distinguisher is allowed to get the verifier's random coins, but the ZK simulator cannot. So, the simulation in [26] is not ABB simulation, which allows the distinguisher to get the verifier's random coins if and only if the ABB simulator can, and thus the conclusion of [26] does not contradict our result.

1.4 Organization of the Paper

The remainder of this paper are organized as follows. In section 2, we give the preliminaries used through the paper. The new proposed notion of generalized claw-free function is presented in section 3. In section 4, we present extractable perfectly-binding bit-commitment and a construction based on generalized claw-free function. Our constructions of 3-round ZKP for any NP language is in section 5. While 2-round ZKA for NP and 2-round ZKP for QNR are showed in section 6.

2 Preliminaries

2.1 Notations and assumptions.

Throughout the paper, n is the security parameter. For any PPT (probabilistic polynomial time) algorithm $A(\cdot)$, $A(x)$ is the result of executing A with input x and the uniformly chosen randomness. We use $y = A(x)$ or $y \leftarrow A(x)$ to denote the output of $A(x)$. For a set S , $y \leftarrow_R S$ denotes that y is uniformly chosen from S . A function $negl(\cdot)$ is negligible if, for any large enough input, its output is smaller than the inverse of any polynomial function $poly(\cdot)$.

2.2 Zero Knowledge

An interactive proof protocol for language $L \in NP$ is an interactive protocol between two parties, the prover P and the verifier V , where P convinces V that the common input x belongs to language L , such that $x \in L$.

Definition 1. (*Interactive Proof*) A 2-party protocol between an unbounded prover P and a polynomial-time verifier V (denoted as $\langle P, V \rangle$) is an interactive proof protocol for language $L \in NP$ if the following two conditions hold:

- *Completeness:* For every $x \in L$, there exists a negligible function $c(\cdot)$ such that,

$$\Pr[\langle P, V \rangle(x) = 1] > 1 - c(|x|)$$

- *Soundness:* For any $x \notin L$, there exists a negligible function $s(\cdot)$ such that,

$$\Pr[\langle P, V \rangle(x) = 1] < s(|x|)$$

where $c(\cdot)$ is the completeness error and $s(\cdot)$ is the soundness error. If the soundness is only required to hold relative to polynomial-time provers, it is called *interactive argument*.

Let $View_{V(z)}^P(x)$ denote the view of V with auxiliary input z and common input x in the real protocol execution with P . The zero knowledge requires that for any PPT verifier V^* , there is a simulator \mathcal{S} with some advantage against prover P , such that the output of \mathcal{S} is indistinguishable from $View_{V^*(z)}^P(x)$.

Definition 2. (*Zero Knowledge*) An interactive protocol $\langle \mathcal{P}, \mathcal{V} \rangle$ for language L is said to be zero knowledge if for every PPT malicious verifier V^* there exists a PPT algorithm (called simulator) \mathcal{S} such that $\{View_{V^*(z)}^P(x)\}_{x \in L, z \in \{0,1\}^*}$ and $\{\mathcal{S}(z, x)\}_{x \in L, z \in \{0,1\}^*}$ are computational indistinguishable.

If $\{View_{V^*(z)}^P(x)\}_{x \in L, z \in \{0,1\}^*}$ and $\{\mathcal{S}(z, x)\}_{x \in L, z \in \{0,1\}^*}$ are statistical indistinguishable, then the interactive protocol is called *statistical zero knowledge*.

Obviously, the ZK simulator \mathcal{S} must be closely related to V^* 's algorithm. If there exists a uniform simulator \mathcal{S} that uses V^* 's strategy in a black-box manner, it is known as the black-box simulator (BB simulator).

Definition 3. (*Black-Box Zero-Knowledge*) Let $\langle P, V \rangle$ be an interactive protocol for a language L . $\langle P, V \rangle$ is black-box zero-knowledge (BBZK) if there exists a probabilistic polynomial-time algorithm \mathcal{S} such that for every probabilistic polynomial-time machine V^* $\{View_{V^*}^P(x)\}_{x \in L}$ and $\{\mathcal{S}^{V^*}(x)\}_{x \in L}$ are computationally indistinguishable.

2.3 Augmented Black-Box ZK

Let $\langle P, V \rangle$ be an interactive protocol for language $L \in NP$. The interactive strategy of V is defined as a next message function, denoted by $V(x, \dots)$. Now we consider an enhanced verifier V which is allowed to obtain auxiliary inputs adaptively. Concretely, V first obtains a common input $x \in L$ and an auxiliary input aux from an oracle \mathcal{O}_{aux} , and then executes the protocol with P . In each verifier-round where V is asked to send message, V may receive

a current auxiliary input z from \mathcal{O}_{aux} (if this verifier-round is the first round, $z = aux$). After that, V computes the next message function $\alpha = V(x, aux, r_V; \beta, z)$, where r_V is V 's random tape and β is P 's message, and then sends α to P (via the communication tape). For convenience we assume that β contains all messages sent by P . Use $\alpha = \perp$ (or $\beta = \perp$) to denote the verifier V (or the prover P) aborts. If V is honest, there is no any oracle \mathcal{O}_{aux} and the next message function ignores z (that is $z = \emptyset$). For a malicious verifier V^* , the next message function $V^*(\dots)$ is unknown and may be any computable function. In addition, we stress that V after obtaining current auxiliary input z may try to extract the information it wants from z , since V is always trying to maximize its attack capabilities. For example, V is asked to compute the commitment to random bit b , $C = Com(b; r)$, but it directly receives as auxiliary input $z = C$. And then, V will try to find (b, r) or some information about (b, r) satisfying $C = Com(b; r)$. For simplicity, we can think that current auxiliary input z itself contains all the information V can get. Let *state* consist of z (if exists) and the random coins (determined by r_V) used by V in the current round.

To introduce augmented black-box simulation, we first imagine that V is given an extra private output tape to record *state*. So, the next message function of V with an extra private output tape can be written as

$$(\alpha, state) = V(x, aux, r_V; \beta, z)$$

In each verifier-round, V sends α to P (via the communication tape) and also writes *state* on the extra private output tape. In fact, V with an extra private output tape is an augmented verifier. We stress that adding an extra private output tape to V does not change the interaction between P and V , since P is completely unaware of the existence of this extra private output tape. So, without loss of generality, we always assume that V is an augmented verifier and still use $\langle P, V \rangle$ to denote the interaction between P and V .

Note that the next message function of augmented verifier V is related to current auxiliary inputs. So, BB simulator for augmented verifiers V is different from that for normal verifier, it not only needs to provide V with P 's message β but also current auxiliary input z . In real interaction, V receives P 's message β and then queries \mathcal{O}_{aux} with β to get current auxiliary input z . To simulate the real conversation, the BB simulator is also allowed to query \mathcal{O}_{aux} which returns the current auxiliary input z . In other words, in addition to *aux* ABB simulator can also get current auxiliary z . Specifically, BB simulators, after specifying the random input r_V (and receiving *aux*), can run V 's next message function in a black-box manner: BB simulators generate P 's message β and obtain the current auxiliary input z , then run $V(x, aux, r_V; \beta, z)$ and receive α from the communication tape.

It is known that the reasons that BB simulator can succeed are that 1) it can rewind V (as black-box algorithm) to extract the verifier's random coins (contained in *state*) and 2) the protocol in questioned itself makes it possible for BB simulators to extract the verifier's secret information by rewinding $V(x, z, r_V; \beta, \cdot)$.

Consider the following 2-round interactive proof for $x \in QNR$.

- V randomly selects $b \in \{0, 1\}$, $r \in Z_n$ and sends $w = r^2 x^b$ to P .
- P determines b' from w and sends b' to V .
- V accepts if and only if $b' = b$.

It is known that the protocol is honest-verifier BBZK. In fact, for semi-honest verifier V' who compute $w = r^2 x^b$ honestly but b is not uniformly distributed on $\{0, 1\}$, V' can only get b

selected by himself from the interacting with P , so the protocol is also ZK for semi-honest verifier V' . However, the BB simulator is unlikely to succeed since the protocol does not provide a way to extract b by rewinding V' , unless it is easy to get b from w . Therefore, the above protocol is not BBZK for semi-honest verifier V' .

Augmented black-box simulator (ABB simulator) are the same as BB simulator, except that it is allowed to access to the extra private output tape of V . Specifically, ABB simulators with x first set the random tape r_V and then execute $V(x, aux, r_V; \beta, z)$ by generating P 's message β (since auxiliary aux and z are available to ABB simulators, so we can always omit aux and z in simulation) round by round until the end of the protocol. Finally, ABB simulator outputs r_V and the simulated conversation (not containing all *state*'s).

To be allowed to access to the extra private output tape, ABB simulators can simulate the real interaction without rewinding V and so have better simulation ability than BB simulators in private coins protocols. For example, in the context of the semi-honest verifier V' , the above interactive proof can be simulated by an ABB simulator \mathcal{S} . In fact, \mathcal{S} only needs to select r_V , get $(w, (r, b)) = V'(x, z, r_V; (r, b))$ from their communication tape and the private output tape of V' , and then output (x, r_V, w, b) . However, in the context of public-coin interactive protocol, ABB simulators have no more advantages than BB simulators.

Zero knowledge is in essence that interacting with prover does not improve verifier's power and is formalized by requiring that there exists an efficient simulator that can simulate verifier's view in a true interaction. The augmented verifier is an imaginary verifier, and so the ABB simulation is a mental experiment. V provided with ABB simulator, however, can complete this mental experiment by himself, since it is in fact the interaction between the ABB simulator and (augmented verifier) V . So, in any case the ABB simulator can only do what the verifier can do alone, although the verifier's private output tape is accessible. Therefore, the fact that the real interaction $\langle P, V \rangle$ is computationally indistinguishable from the output of the ABB simulator means that interacting with P must not improve the V 's computational power except for a negligible probability. This shows that using ABB simulator to formalize zero knowledge meets the requirement of zero knowledge and is reasonable.

Definition 4. (*Augmented Black-Box ZK*) An interactive proof system for language L is called augmented black-box ZK if for every PPT verifier V^* , there exists an ABB simulator \mathcal{S} , such that for any auxiliary input aux and $x \in L$, $\{View_{V^*}^P(x)\}_{x \in L, aux \in \{0,1\}^*}$ and $\{\mathcal{S}^{V^*}(x, aux)\}_{x \in L, aux \in \{0,1\}^*}$ are computationally indistinguishable.

The ABB simulation gives ZK simulator more capabilities but the computing power of ZK simulator does not exceed that of the verifier. In other words, ABB simulator can only output information that the verifier can obtain by alone. So, using the ABB simulation to formalize zero knowledge does not weaken zero knowledge in essence.

2.4 Garbled Circuits

Garbled circuit was first presented by Yao [47], and has been formalized by Bellare et al. [8]. According to the formal language of [8], a garbling circuit scheme *Garble* is defined by a tuple algorithms, $Garble = (Gb, En, Ev, De, Ve)$.

Definition 5. (*Garbled Circuits*) A garbling circuit scheme consists of a tuple of polynomial algorithms $Garble = (Gb, En, Ev, De, Ve)$.

- $Gb(1^n, C)$ takes the security parameter n and a circuit $C : \{0, 1\}^\ell \rightarrow \{0, 1\}^k$ as input, outputs a garbled circuit \widehat{C} of C and a pair of keys (e, d) , where e is an encoding key and d is decoding list.
- $En(e, x)$ takes $x \in \{0, 1\}^\ell$ and encoding key e as input, outputs the garbled encoding \widehat{x} of $x \in \{0, 1\}^\ell$.
- $Ev(\widehat{C}, \widehat{x})$ evaluates garbled circuit \widehat{C} on garbled encoding \widehat{x} , and outputs a garbled output \widehat{y} .
- $De(d, \widehat{y})$ outputs the decoding of \widehat{y} .
- $Ve(C, \widehat{C}, e)$ outputs 1 if \widehat{C} is a valid garbling circuit of C , output 0 otherwise.

A garbling scheme is correct if for any x , it holds that $De(d, Ev(\widehat{C}, En(e, x))) = C(x)$. Except for correctness, a garbling scheme may satisfy privacy and authenticity.

Privacy. Privacy is to protect the privacy of encoding inputs. Bellare et al. presented the formal definition for privacy based on indistinguishability game[8].

Assume that C_0 and C_1 are two circuits with the same side-information and (x_0, x_1) satisfies $C_0(x_0) = C_1(x_1)$. The privacy definition requires that, for any adversary \mathcal{A} , it holds that

$$\Pr \left[b = b' \mid b \leftarrow_R \{0, 1\}, (\widehat{C}, e, d) \leftarrow Gb(1^n, C_b), \widehat{x} \leftarrow En(e, x_b), b' \leftarrow \mathcal{A}(\widehat{C}, \widehat{x}, d) \right] = \frac{1}{2} + \text{negl}(n)$$

Authenticity. Authenticity is that adversary given with \widehat{C} and $\widehat{x} = En(e, x)$ can only learn $Ev(\widehat{C}, \widehat{x})$. Specifically, authenticity asks, for any adversary \mathcal{A} , it holds that

$$\Pr \left[De(d, \widehat{y}) \neq \perp, \widehat{y} \neq Ev(\widehat{C}, \widehat{x}) \mid (\widehat{C}, e, d) \leftarrow Gb(1^n, C), \widehat{x} \leftarrow En(e, x), \widehat{y} \leftarrow \mathcal{A}(\widehat{C}, \widehat{x}, d) \right] = \text{negl}(n)$$

for given C and x .

In this paper, we use Yao’s garbled circuits scheme [47], which uses private key encryption scheme $\Pi = (G, E, D)$ with the following properties: 1) an encryption under one key will not fall in the range of an encryption under another key except for a negligible probability; 2) Given the key K , it is easy to decide whether a given ciphertext falls into the range of encryptions under K .

Let $C : \{0, 1\}^\ell \rightarrow \{0, 1\}^k$ be a acyclic circuit that has $m + k$ gates. So C has $t = \ell + m + k$ wires, denoted as w_1, \dots, w_t , where w_1, \dots, w_ℓ are the input wires and w_{t-k+1}, \dots, w_t are the output wires of C . To garble C , Gb first selects a pair of key (K_i^0, K_i^1) from key space \mathcal{K} to represent the bit values of 0 or 1 on wire w_i ($i = 1, \dots, t$), and sets

$$e = \{K_i^0, K_i^1\}_{i=1}^\ell, d = \{(0, K_{t-k+1}^0), (1, K_{t-k+1}^1), \dots, (0, K_t^0), (1, K_t^1)\}$$

Once all the keys for the wires in the circuit have been chosen, garble each gate in C as follows: For any gate g with two input wires w_i, w_j and one output wire w_o , compute $c_{a,b} = E_{K_i^a}(E_{K_j^b}(K_o^{g(a,b)}))$, $a, b = 0, 1$. The garbled gate of g is represented by a “garbled computation table” $\widehat{g} = (c_0, c_1, c_2, c_3)$ which is the random order of $(c_{0,0}, c_{0,1}, c_{1,0}, c_{1,1})$. The garbled circuit \widehat{C} of C consists of the garbled gate for each gate, $\widehat{C} = \{\{\widehat{g}\}_{g \in C}\}$.

For any $x = x_1 \dots x_\ell$, $\widehat{x} = En(e, x) = \{K_1^{x_1}, \dots, K_\ell^{x_\ell}\}$ is the encoding of x . $Ev(\widehat{C}, \widehat{x})$ computes each gate according to its “garbled computation table”, gate by gate, and obtains the

output of k output gates, $\hat{y} = Ev(\hat{C}, \hat{x})$. Finally, $De(d, \hat{y})$ decodes \hat{y} , $De(d, \hat{y}) = z_1 \cdots z_k$ if and only if $\hat{y} = (K_{t-k+1}^{z_1}, \dots, K_t^{z_k})$. $Ve(C, \hat{C}, e)$ verifies every $\hat{g} = (c_0, c_1, c_2, c_3)$, gate by gate, by decrypting c_0, c_1, c_2, c_3 . $Ve(C, \hat{C}, e) = 1$ if all the verifications pass, $Ve(C, \hat{C}, e) = 0$ otherwise.

It is easy to see that Yao's garbled circuits scheme is correct. In addition, Yao's garbled circuits scheme satisfies privacy and authenticity [8].

2.5 Indistinguishability Obfuscation

Barak et al. [5] first proposed the notion of virtual black-box (VBB) obfuscation which requires that the obfuscation of one arbitrary function leaks nothing except what can be learnt from a black-box oracle access to the function. Unfortunately, [5] showed that there is a family of circuits that cannot be VBB obfuscated. Therefore, [5] presented a weaker notion of obfuscation called indistinguishable obfuscation (iO), but left the problem of whether or not indistinguishable obfuscation exists. The iO only requires that for any two equivalent circuits C_0 and C_1 of similar poly-size, any probabilistic polynomial-time (PPT) adversarial algorithm can distinguish between the obfuscations of C_0 and C_1 with negligible probability. This security of iO may seem rather weak than VBB obfuscation. Garg et al. proposed the first candidate construction of general-purpose indistinguishability obfuscator [29]. Recently, Sahai and Waters constructed a variety of core cryptographic objects such as deniable encryption scheme [46].

Definition 6. Indistinguishability obfuscation (iO) [29]. A PPT algorithm iO is called an indistinguishability obfuscator for a circuit ensemble $\{C_n\}_{n \in N}$ if the following conditions are satisfied:

- (functionality) For all security parameters $n \in N$, for all $C \in C_n$, and for all input x we have that

$$\Pr[C'(x) = C(x) : C' \leftarrow iO(1^n, C)] = 1.$$

- (security) For any PPT distinguisher D , there exists a negligible function $\text{negl}(\cdot)$ such that the following holds: For all security parameters $n \in N$, for all pairs of same size circuits $C_0, C_1 \in C_n$, we have that if $C_0(x) = C_1(x)$ for all inputs x , then

$$|\Pr[D(1^n, iO(1^n, C_0)) = 1] - \Pr[D(1^n, iO(1^n, C_1)) = 1]| \leq \text{negl}(n)$$

3 Generalized Claw-Free Functions

In this section we generalize claw-free functions to propose the notion of generalized claw-free functions. Let $f_s^0 : D_s^0 \rightarrow R_s$ and $f_s^1 : D_s^1 \rightarrow R_s$. A pair (x_0, x_1) satisfying $f_s^0(x_0) = f_s^1(x_1)$ is called a claw of (f_s^0, f_s^1) . Roughly speaking, claw-free functions are a collection of pairs of functions (f_s^0, f_s^1) , satisfying that it is infeasible to find a claw of (f_s^0, f_s^1) .

Definition 7. (Claw-Free Function [27]) A collection of pairs of functions $\mathcal{F} = \{f_s^\sigma : D_s^\sigma \rightarrow R_s, \sigma = 0, 1\}_{s \in \bar{S}}$ is called claw-free functions if the following conditions hold:

- *Easy to sample:* There exist two PPT sampling algorithms \mathcal{S} and \mathcal{D} , such that $s \leftarrow \mathcal{S}(1^n)$ is distributed over $\bar{S} \cap \{0, 1\}^n$ and $x \leftarrow \mathcal{D}(s, \sigma)$ is distributed over D_s^σ for any $s \leftarrow \mathcal{S}(1^n)$ and $\sigma \in \{0, 1\}$.

- *Easy to compute: There exists a PPT algorithm F , such that*

$$F(s, \sigma, x) = f_s^\sigma(x)$$

for any $s \leftarrow \mathcal{S}(1^n)$, $x \leftarrow \mathcal{D}(s, \sigma)$ and $\sigma \in \{0, 1\}$.

- *Identical range distribution: For every $s \leftarrow \mathcal{S}(1^n)$, the random variables $F(s, 0, \mathcal{D}(s, 0))$ and $F(s, 1, \mathcal{D}(s, 1))$ are identically distributed.*
- *Hard to form claws: for any PPT algorithm \mathcal{A} , it holds that:*

$$\Pr[f_s^0(x_0) = f_s^1(x_1) : s \leftarrow \mathcal{S}(1^n), (x_0, x_1) \leftarrow \mathcal{A}(1^n, s)] < \text{negl}(n)$$

We first present the notion of generalized claw-free functions and then give a detailed construction. Roughly speaking, generalized claw-free functions consist of two pair of functions (f_s^0, t_s^0) and (f_s^1, t_s^1) , where (f_s^0, f_s^1) is defined as in claw-free functions, and t_s^σ is defined over D_s^σ , and satisfy the condition: it is infeasible to find (z_0, z_1) such that $z_0 = t_s^0(x_0)$, $z_1 = t_s^1(x_1)$, $f_s^0(x_0) = f_s^1(x_1)$.

Definition 8. (*Generalized Claw-Free Function*) Let \bar{S} be a infinite index set, D_s^0, D_s^1 be two finite sets for any $s \in \bar{S}$. Let f_s^σ and t_s^σ be two functions defined over D_s^σ with value range R_s for any $s \in \bar{S}$ and $\sigma \in \{0, 1\}$. The collection of two pairs of functions $\mathcal{FT} = \{(f_s^0, t_s^0), (f_s^1, t_s^1)\}_{s \in \bar{S}}$ is called *generalized claw-free functions*, if the following conditions hold:

- *Easy to sample: There exist two PPT sampling algorithms \mathcal{S} and \mathcal{D} , such that $s \leftarrow \mathcal{S}(1^n)$ is distributed over $\bar{S} \cap \{0, 1\}^n$ and $x \leftarrow \mathcal{D}(s, \sigma)$ is distributed over D_s^σ for any $s \in \mathcal{S}(1^n)$, $\sigma \in \{0, 1\}$.*
- *Easy to compute: There exist PPT algorithms F, T , such that*

$$F(s, \sigma, x) = f_s^\sigma(x), T(s, \sigma, x) = t_s^\sigma(x)$$

for any $s \leftarrow \mathcal{S}(1^n)$, $\sigma \in \{0, 1\}$ and $x \leftarrow \mathcal{D}(s, \sigma)$.

- *Identical range distribution: For any $s \leftarrow \mathcal{S}(1^n)$, $F(s, 0, \mathcal{D}(s, 0))$ and $F(s, 1, \mathcal{D}(s, 1))$ are identically distributed. $T(s, 0, \mathcal{D}(s, 0))$ and $T(s, 1, \mathcal{D}(s, 1))$ are identically distributed.*
- *Hard to form generalized claws: For any $s \in \mathcal{S}(1^n)$, let*

$$L_s = \{(z, z_0, z_1) : \exists x_0, x_1, z = f_s^0(x_0) = f_s^1(x_1); z_0 = t_s^0(x_0), z_1 = t_s^1(x_1)\}$$

A triple $(z, z_0, z_1) \in L_s$ is called a *generalized claw for index s* . For any PPT algorithm \mathcal{A} , it holds that

$$\Pr[(z, z_0, z_1) \in L_s : s \leftarrow_R \mathcal{S}(1^n), (z, z_0, z_1) \leftarrow \mathcal{A}(s)] < \text{negl}(n)$$

Generalized claw-free functions are denoted by $GCLFF = (\mathcal{FT}, \mathcal{S}, \mathcal{D}, F, T)$.

It is easy to see that if $\{f_s^0, f_s^1\}_{s \in \bar{S}}$ is a claw-free function and functions t_s^0, t_s^1 are 1-1 and easy to inverse, then $\{(f_s^0, t_s^0), (f_s^1, t_s^1)\}_{s \in \bar{S}}$ must be generalized claw-free function. On the other hand, under the condition that t_s^0, t_s^1 are easy to compute, $\{(f_s^0, t_s^0), (f_s^1, t_s^1)\}_{s \in \bar{S}}$ is a generalized claw-free function implies that $\{(f_s^0, f_s^0)\}_{s \in \bar{S}}$ is a claw-free function. Especially, a claw-free functions is a generalized claw-free function where $t_s^0(x) \equiv x, t_s^1(x) \equiv x$.

Definition 9. (Trapdoor Generalized Claw-Free Function) $\mathcal{FT} = \{(f_s^0, t_s^0), (f_s^1, t_s^1)\}_{s \in \bar{S}}$ defined in Definition 8 is trapdoor generalized claw-free functions. A generalized claw-free functions is called trapdoor generalized claw-free functions, if

- Easy to sample: There exist two PPT sampling algorithms \mathcal{S} and \mathcal{D} . $(s, tr) \leftarrow \mathcal{S}(1^n)$, s is distributed over $\bar{S} \cap \{0, 1\}^n$ and tr is a trapdoor. For any $s \in \bar{S} \cap \{0, 1\}^n, \sigma \in \{0, 1\}$, $x \leftarrow \mathcal{D}(s, \sigma)$ is distributed over D_s^σ .
- Easy to compute: There exist PPT algorithms F, T , such that

$$F(s, \sigma, x) = f_s^\sigma(x), T(s, \sigma, x) = t_s^\sigma(x)$$

for any $s \leftarrow \mathcal{S}(1^n), \sigma \in \{0, 1\}$ and $x \leftarrow \mathcal{D}(s, \sigma)$.

- Identical range distribution: For any $s \leftarrow \mathcal{S}(1^n)$, $F(s, 0, \mathcal{D}(s, 0))$ and $F(s, 1, \mathcal{D}(s, 1))$ are identically distributed. $T(s, 0, \mathcal{D}(s, 0))$ and $T(s, 1, \mathcal{D}(s, 1))$ are identically distributed.
- Hard to form claws: For any $s \in \bar{S}(1^n)$, let

$$L_s = \{(z, z_0, z_1) : \exists x_0, x_1, z = f_s^0(x_0) = f_s^1(x_1); z_0 = t_s^0(x_0), z_1 = t_s^1(x_1)\}$$

A triple $(z, z_0, z_1) \in L_s$ is called a generalized claw for index s . For any PPT algorithm \mathcal{A} , it holds that

$$\Pr[(z, z_0, z_1) \in L_s : s \leftarrow_R \mathcal{S}(1^n), (z, z_0, z_1) \leftarrow \mathcal{A}(s)] < \text{negl}(n)$$

- Easy to form claws with trapdoor. There exists a PPT algorithm Ft , for any $(s, tr) \leftarrow \mathcal{S}(1^n)$ and $\sigma \in \{0, 1\}$, $Ft(s, tr, \sigma, F(s, \sigma, x)) = T(s, \sigma, x)$.

Trapdoor generalized claw-free functions is denoted by $TGCFF = (\mathcal{FT}, \mathcal{S}, \mathcal{D}, F, T, Ft)$.

3.1 A construction of trapdoor generalized claw-free function

In this subsection, we present a trapdoor generalized claw-free function $TGCFF = (\mathcal{FT}, \mathcal{S}, \mathcal{D}, F, T, Ft)$.

Let $q, p = 2q + 1$ be two prime, \mathcal{G} be the order q subgroup of \mathbb{Z}_p^* , g be a generator. Let $\bar{S} = \{(q, g, h) : h = g^r, r \in \mathbb{Z}_q\}$, for any $s = (q, g, h)$, define

$$f_s^0(u, v) = (u, g^u h^v), t_s^0(u, v) = (u, g^v)$$

$$f_s^1(u, v) = (u, g^v h^u), t_s^1(u, v) = (u, h^v)$$

Let $\mathcal{FT} = \{(f_s^0, t_s^0), (f_s^1, t_s^1)\}_{s \in \bar{S}}$, $D_s^0 = D_s^1 = \mathbb{Z}_q^2$, $R_s = \mathbb{Z}_q \times \mathcal{G}$. Next, we only need to define the algorithms \mathcal{S} and Ft .

- $(s, tr) \leftarrow \mathcal{S}(1^n)$: Randomly select an n -bit prime q such that $p = 2q + 1$ is a prime. Let g be a generator of the order q subgroup \mathcal{G} . Randomly select $r \in \mathbb{Z}_q$. Let $s = (q, g, h = g^r), tr = r$.
- $(u, v) \leftarrow \mathcal{D}(s, \sigma)$: $u, v \leftarrow_R \mathbb{Z}_q$.
- $F(s, \sigma, (u, v))$: $s = (q, g, h)$, $F(s, \sigma, (u, v)) = \begin{cases} (u, g^u h^v), & \sigma = 0 \\ (u, g^v h^u), & \sigma = 1 \end{cases}$
- $Ft(s, tr, \sigma, f_s^\sigma(u, v))$: Let $s = (q, g, h), tr = r$, $f_s^\sigma(u, v) = (u, z)$. Define

$$Ft(s, r, \sigma, f_s^\sigma(u, v)) = \begin{cases} (u, (z/g^u)^{\frac{1}{r}}), & \sigma = 0 \\ (u, (z/h^u)^r), & \sigma = 1 \end{cases}$$

Theorem 1. *If the discrete logarithm assumption and KEA hold, then $(\mathcal{FT}, \mathcal{S}, \mathcal{D}, F, Tt)$ defined as above is a trapdoor generalized claw-free function.*

Proof. Let $\mathcal{FT} = \{(f_s^0, t_s^0), (f_s^1, t_s^1)\}_{s \in \bar{\mathcal{S}}}$. Obviously, $t_s^\sigma(u, v)$ is easy to compute from $f_s^\sigma(u, v)$ and r for any $s = (g, g, h = g^r)$, that is, r is a trapdoor. We only need to prove that finding a generalized-claw is hard.

Intuitively, if for some PPT algorithm \mathcal{A} with input (g, h) , can find a generalized-claw

$$z = (u, g^u h^{v_0}) = (u, g^{v_1} h^u), z_0 = (u, g^{v_0}), z_1 = (u, h^{v_1})$$

Then \mathcal{A} can compute $y/g^u = h^{v_0}$ and $y/h^u = g^{v_1}$, where $y = g^u h^{v_0} = g^{v_1} h^u$. Therefore, \mathcal{A} is able to output (g^{v_0}, h^{v_0}) and (g^{v_1}, h^{v_1}) . By KEA, there is an extracting algorithm \mathcal{E} to extract v_0 and v_1 . Using \mathcal{E} , we can construct an algorithm \mathcal{B} to output the discrete logarithm of h from the equation $g^u h^{v_0} = g^{v_1} h^u$. This contradicts the discrete logarithm assumption. ■

Knowledge of Exponent Assumption (KEA)[11] Let $q, p = 2q + 1$ be two prime, \mathcal{G} be the order q subgroup of \mathbb{Z}_p^* , g be a generator. For any adversary A that takes input q, g, g^a and returns (C, Y) satisfying $Y = C^a$, there exists an extraction algorithm \mathcal{E} , which given the same inputs as A returns c such that $C = g^c$.

4 Completely Extractable Bit-Commitment Scheme

4.1 Bit-Commitment Scheme

We focus on bit-commitment schemes which imply general commitment schemes.

An interactive bit-commitment scheme $\langle S, R \rangle$ consists of two phases: commit stage $\langle S_c, R_c \rangle$ (where S_c commits to its input $b \in \{0, 1\}$) and open stage $\langle S_o, R_o \rangle$ (where S_o opens b and R_o checks the correctness of the commitment).

Definition 10. (*Bit-Commitment Scheme*) An interactive bit-commitment scheme $\langle S, R \rangle$ consists of two phases, commit stage and open stage:

- *Commit Stage $\langle S_c, R_c \rangle$: $\langle S_c, R_c \rangle$ is an (non-)interactive commitment protocol that S_c commits to $b \in \{0, 1\}$ with random input $r \in \{0, 1\}^{\text{poly}(n)}$, denoted by $\langle S_c(b; r), R_c \rangle$. Except for interacting with the verifier, $S_c(b; r)$ needs to compute z from b and r , denoted by $z = Z(b, r)$, for the open stage (in general, $z = r$). If $\langle S_c, R_c \rangle$ is non-interactive, $S_c(b)$ computes and sends $c = \text{Com}(b; r)$ to R_c , where Com is a PPT algorithm and r is the random input.*
- *Open Stage $\langle S_o, R_o \rangle$: $\langle S_o(b, z), R_o \rangle$ is an (non-)interactive open protocol, where S opens b and R_o checks the correctness of the commitment. When $\langle S_o, R_o \rangle$ is non-interactive S_o opens the commitment by sending (b, z) to R_o .*

The correctness requirement is simple: the commitment to b will be accepted by R when both of parties are honest. The security of commitment schemes $\langle S, R \rangle$ has two aspects: hiding property (protecting against cheating receivers) and binding property (protecting against cheating senders).

$\langle S, R \rangle$ is a bit commitment scheme and n is security parameter. Let $\varepsilon(n)$ denote the probability that S successfully opens a completed commitment $\langle S_c, R_c \rangle$ as either a commitment to 0 or a commitment to 1.

Definition 11. $\langle S, R \rangle$ is statistically (computationally) binding if for any (PPT) sender S^* , $\varepsilon(n)$ is negligible. Furthermore, if $\varepsilon(n) \equiv 0$, $\langle S, R \rangle$ is perfectly binding.

Let $view_{R_c}^{S_c(b;r)}(1^n)$ be the view of R_c in commit stage and consists of R_c 's random-input and the messages it receives from S_c when S_c commits to b using random input r , $view_{R_c}^{S_c(b)}(1^n)$ be the random distribution of $view_{R_c}^{S_c(b;r)}(1^n)$.

Definition 12. $\langle S, R \rangle$ is statistically (computationally) hiding if for any R^* , $view_{R_c^*}^{S_c(0)}(n)$ is statistically (computationally) indistinguishable from $view_{R_c^*}^{S_c(1)}(1^n)$. Especially, when $view_{R_c^*}^{S_c(0)}(1^n)$ and $view_{R_c^*}^{S_c(1)}(1^n)$ are identically distributed, $\langle S, R \rangle$ is perfectly hiding.

A bit-commitment scheme is statistically hiding (binding) means that it is computationally binding (hiding).

4.2 Completely Extractable Bit-Commitment Scheme

Let $\langle S, R \rangle$ be a bit commitment scheme. To commit to b , $S_c(b; r_b)$ interacts with R_c and computes $z = Z(b, r_b)$ for the open phase $\langle S_o, R_o \rangle$, such that $\langle S_o(b; z), R_o(\cdot) \rangle = \text{accept}$. If $\langle S, R \rangle$ is perfectly hiding means that for $b \in \{0, 1\}$, $r_b \in \{0, 1\}^{\text{poly}(n)}$, there exists $r_{1-b} \in \{0, 1\}^{\text{poly}(n)}$ such that $view_{R_c}^{S_c(b;r_b)}(1^n) = view_{R_c}^{S_c(1-b;r_{1-b})}(1^n)$. $\langle S, R \rangle$ is completely extractable if, at the end of commit stage $\langle S_c(b; r_b), R_c \rangle$, the receiver R (may need some trapdoor) can extract the random inputs z_0, z_1 from its view in the commitment stage, such that $\langle S_o(0; z_0), R_o(\cdot) \rangle$ and $\langle S_o(1; z_1), R_o(\cdot) \rangle$ are accepted and $z_b = Z(b, r_b)$ is computed by S_c .

Definition 13. Let $\langle S, R \rangle$ be a bit-commitment scheme. $\langle S, R \rangle$ is completely extractable if there exists a PPT algorithm Ext , after commit stage $\langle S_c(b; r), R_c \rangle$, the receiver using Ext and a special trapdoor can extract z_0, z_1 from the messages received in commit stage, such that $\langle S_o(0; z_0), R_o \rangle = \langle S_o(1; z_1), R_o \rangle = \text{accept}$ and $z_b = Z(b, r)$, where $Z(b, r)$ is computed by S_c in commit stage.

The complete extractability of bit-commitment schemes as above does not contradict the hiding property. Even if receiver extracts z_0, z_1 , it still cannot get b that sender has committed to. On the other hand, the binding property asks that sender without the trapdoor cannot obtain z_0, z_1 to break the binding property.

Completely extractable bit-commitment schemes can be constructed from trapdoor generalized claw-free functions. Let $TGCF = (\mathcal{FT}, \mathcal{S}, \mathcal{D}, F, T, Ft)$ be trapdoor generalized claw-free functions, where $\mathcal{FT} = \{(f_s^0, t_s^0), (f_s^1, t_s^1)\}$. The bit-commitment scheme based on $TGCF$ is as follows:

Construction 4.1: Bit-commitment scheme

- **Commit Stage** $\langle S_c(b), R_c \rangle$:
 - The receiver runs $(s, tr) \leftarrow \mathcal{S}(1^n)$, and sends s to the sender.
 - To commit to $b \in \{0, 1\}$, the sender computes $(c, z_b) = Com(s, b)$ as follows:
 - * Run $x_b \leftarrow \mathcal{D}(s, b)$.
 - * Compute $c = F(s, b, x_b) = f_s^b(x_b)$, $z_b = T(s, b, x_b) = t_s^b(x_b)$.
- Finally, the sender sends c to the receiver.

– **Open Stage** $\langle S_o(b, z_b), R_o(c, tr) \rangle$:

- The sender reveals b, z_b to the receiver.
- After receiving (b, z_b) from the sender, the receiver verifies $z_b = Ft(s, tr, b, c)$.

The corresponding extraction algorithm $Ext: (z_0, z_1) = Ext(tr, c)$, where $z_0 = Ft(s, tr, 0, c)$, $z_1 = Ft(s, tr, 1, c)$.

Theorem 2. *If $TGCF = (\mathcal{FT}, \mathcal{S}, \mathcal{D}, F, T, Ft)$ is trapdoor generalized claw-free functions, the above commitment scheme (Construction 4.1) is a completely extractable perfectly hiding (computationally binding) bit commitment scheme.*

Proof. Computationally-binding. Let $(s, tr) \leftarrow \mathcal{S}(1^n)$. For any given commitment (s, c) , to open it as a commitment to b , the sender must find $z_b = T(s, b, x_b)$. So, if the scheme is not computationally-binding, the sender can find a generalized claw $(c, z_0, z_1) \in L_s$ for index s . By the assumption that $TGCF$ is a generalized claw-free function, this is infeasible.

Perfectly-hiding. It is implied by the fact that $F(s, 0, \mathcal{D}(s, 0))$ and $F(s, 1, \mathcal{D}(s, 1))$ are identically distributed.

Completely extractable. The receiver holding trapdoor tr can compute $z_b = Ft(s, tr, b, c)$ for $b = 0, 1$. ■

Using the construction of trapdoor generalized claw-free function in section 3.1, the scheme is showed as follow.

– **Commit Stage** $\langle S_c(b), R_c \rangle$:

- The receiver randomly select an n -bit prime q such that $p = 2q + 1$ is a prime. Let g be a generator of the order q subgroup \mathcal{G} of Z_p^* . Randomly select $r \in \mathbb{Z}_q$ and computes $h = g^r$. Then, sends $s = (q, g, h)$ to the sender.
- To commit to $b \in \{0, 1\}$, the sender computes $(c, z_b) = Com(s, b)$ as follows:
 - * The sender selects $u, v \in_R \mathbf{Z}_q$.
 - * If $b = 0$, the sender computes $c = f_s^0(u, v) = (u, g^u h^v)$, $z_0 = (u, g^v)$. If $b = 1$, the sender computes $c = f_s^1(u, v) = (u, g^v h^u)$, $z_1 = (u, h^v)$

Then the sender sends c to the receiver.

– **Open Stage** $\langle S_o(b, z_b), R_o(c, r) \rangle$:

- The sender reveals $(0, z_0 = (u, g^v))$ when $b = 0$, or $(1, z_1 = (u, h^v))$ when $b = 1$.
- Let $c = (u, \varphi)$. After receiving (b, z_b) , the receiver holding r verifies

$$z_b = \begin{cases} \left(u, \left(\frac{\varphi}{g^u} \right)^{1/r} \right), & b = 0 \\ \left(u, \left(\frac{\varphi}{h^u} \right)^r \right), & b = 1 \end{cases}$$

The corresponding extraction algorithm $Ext(tr, c)$ is obvious.

5 3-Round ZKP for NP Using Completely Extractable Commitment

In this section, we present a 3-round zero knowledge proof for any $L \in NP$. Furthermore, with augmented black-box simulation technique, we show that the zero knowledge property of our construction is closed under parallel composition.

5.1 3-Round Protocol for NP

Let $L \in NP$ and R_L be the corresponding NP-relation. For any $x \in L \cap \{0, 1\}^n$, there exists $y \in \{0, 1\}^\ell$ such that $R_L(x, y) = 1$, where $\ell = \text{poly}(n)$. Let $C_{L,x} : \{0, 1\}^\ell \rightarrow \{0, 1\}$ be a circuit to compute $R_L(x, \cdot)$. $C_{L,x}$ has t gates and the output gate of $C_{L,x}$ is denoted by g_t .

Let $\text{Garble} = (Gb, En, Ev, De, Ve)$ be Yao's garbled circuit scheme. Using the traditional model of 3-round interactive proof, "commit-challenge-reply", we consider the following interactive proof protocol for $L \in NP$.

Construction 5.1: ZKP for $L \in NP$

Common input: $x \in L$.

P 's auxiliary input: witness y for $x \in L$.

- Commit: The prover P garbles $C_{L,x}(\cdot)$, that is, $(\widehat{C}_{L,x}, e, d) \leftarrow Gb(C_{L,x}, 1^n)$, where

$$e = \{K_i^0, K_i^1\}_{i=1}^\ell, \quad d = \{(0, K^0), (1, K^1)\}$$

P sends $(\widehat{C}_{L,x}, d)$ to the verifier V .

- Challenge: V randomly picks $\sigma \in \{0, 1\}$ and sends it to P .
- Reply: Let $A_0 = \widehat{y} = En(e, y) = \{K_i^{y_i}\}_{i=1}^\ell$ and $A_1 = e$, where $y = y_1 \cdots y_\ell$ is a witness. P replies to V with A_σ .
- Verify: V verifies A_σ as follows:
 - If $\sigma = 0$, compute $\widehat{z} = Ev(\widehat{C}_{L,x}, A_0)$ and verify $De(d, \widehat{z}) = 1$.
 - If $\sigma = 1$, verify $Ve(C_{L,x}, \widehat{C}_{L,x}, A_1, d) = 1$.

The prover commits to the fact that there exists y satisfying $C_{L,x}(y) = 1$ by $(\widehat{C}_{L,x}, d)$. The verifier verifies $\widehat{C}_{L,x}$ is a garbled circuit of $C_{L,x}$ (when $\sigma = 1$) or there exists y satisfying $\widehat{C}_{L,x}(y) = K^1$ (when $\sigma = 0$). The property of Yao's garbling scheme guarantees that the above protocol is complete and sound with error probability $\frac{1}{2}$, and thus an interactive proof for L . Moreover, the protocol is a (black-box) zero knowledge proof with soundness error probability $\frac{1}{2}$.

For any V^* , define a black-box simulator Sim that processes as follows:

- Randomly select r_V used as the random input of V^* .
- Select $\delta \in \{0, 1\}$ randomly.
 - If $\delta = 1$, Let $C'_{L,x} = C_{L,x}$. Sim garbles $C'_{L,x}$ as an honest prover, i.e. $(\widehat{C}'_{L,x}, e, d) \leftarrow Gb(C'_{L,x}, 1^n)$, where

$$e = \{(K_i^0, K_i^1)\}_{i=1}^\ell, \quad d = \{(0, K^0), (1, K^1)\}$$

And then set $A_1 = e$.

- If $\delta = 0$, let $C'_{L,x}$ be the same as $C_{L,x}$ except that its output gate g'_t is set as $g'_t(a, b) \equiv 1$. Sim garbles $C'_{L,x}$ honestly: $(\widehat{C}'_{L,x}, e, d) \leftarrow Gb(C'_{L,x}, 1^n)$, where

$$e = \{K_i^0, K_i^1\}_{i=1}^\ell, \quad d = \{(0, K^0), (1, K^1)\}$$

Then, randomly select $u = u_1 \cdots u_\ell \in \{0, 1\}^\ell$, and set $A_0 = \{K_i^{u_i}\}_{i=1}^k$.

- Invoke V^* with $(\widehat{C}'_{L,x}, d)$ and receive σ from V^* .
- If $\sigma = \delta$, output $(r, (\widehat{C}'_{L,x}, d), A_\sigma)$. Else, output \perp and abort.

Obviously, $\Pr[\text{Sim}(x) = \perp] = \Pr[\delta \neq \sigma] = \frac{1}{2}$. When $\delta = \sigma = 1$, \mathcal{O}_{V^*} does not fail and $\text{Sim}(x)$ is the same as $\text{View}_{V^*}^P(x)$ since $C'_{L,x} = \widehat{C}_{L,x}$.

When $\delta = \sigma = 0$, $C'_{L,x}$ and $C_{L,x}$ are exactly the same except that the output gates g'_t and g_t are different, so the only difference between $\widehat{C}'_{L,x}$ and $\widehat{C}_{L,x}$ is that \widehat{g}'_t (the “garbled truth table” of the output gate g'_t) is different from \widehat{g}_t . Specifically, $\widehat{g}'_t = (c_0, c_1, c_2, c_3)$ is the random order of $c_{a,b} = E_{K_i^a}(E_{K_j^b}(K_t^1))$, $a, b = 0, 1$, but $\widehat{g}_t = (c_0, c_1, c_2, c_3)$ is the random order of $c_{a,b} = E_{K_i^a}(E_{K_j^b}(K_t^{g_t(a,b)}))$, $a, b = 0, 1$. Therefore, if the private key encryption scheme (G, E, D) is IND-CPA, $\widehat{C}'_{L,x}$ and $\widehat{C}_{L,x}$ are computationally indistinguishable. Moreover, notice that $\{K_i^{u_i}\}_{i=1}^k$ and $\{K_i^{y_i}\}_{i=1}^k$ are randomly selected from \mathcal{K} and have the same distribution. So, $\text{Sim}^{\mathcal{O}_{V^*}}(x)$ and $\text{View}_{V^*}^P(x)$ are computationally indistinguishable under the condition that sim does not fail.

Due to the fact that P needs to send $(\widehat{C}_{L,x}, d)$ before V^* publishing the random challenge σ , BB simulator Sim must guess σ and then generates $(\widehat{C}_{L,x}, d)$ that is consistent with σ . So, the above protocol cannot be proved to be zero knowledge under parallel composition.

To construct a parallel zero knowledge proof for L , we consider a new interactive proof model, simplified as “challenge-commit and reply”, where V first selects and commits to its random challenge σ , and then P computes its commitment to $C_{L,x}$ and responds to V 's challenge (without learning σ). Finally, V verifies the received response.

In this new interactive proof model, V 's challenge must be committed by a completely extractable perfectly-hiding commitment scheme. On the one hand, to ensure that the protocol is sound, V 's commitment must be statistically-hiding. On the other hand, since P must respond to V 's challenge σ without knowing σ , P has to give two answers (one for $\sigma = 0$ and the other for $\sigma = 1$) in “ciphertext”. In order to make sure honest V can obtain a correct answer corresponding σ and no verifier can get the two answers at the same time, we need that V 's commitment is completely extractable. The structure of interactive proof is as follows.

- V and P execute a (completely extractable) statistically-hiding (computationally binding) bit commitment scheme. V commits to a random bit σ and obtains $z = z_1 \cdots z_q \in \{0, 1\}^q$ to be used in the open stage.
- P first extracts $(0, z^0)$ and $(1, z^1)$ from the commitment and runs $(\widehat{C}_{L,x}, e, d) \leftarrow Gb(1^n, C_{L,x})$. Then, P , without knowing σ) responds V 's challenge in a special way, such that honest verifier can get what it wants (completeness), $\widehat{y} = \text{En}(e, y)$ (when $\sigma = 0$) or e (when $\sigma = 1$, and no verifier can get $\widehat{y} = \text{En}(e, y)$ and e at the same time (zero knowledge). To this end, P hides $\widehat{y} = \text{En}(e, y)$ and e by z^0 and z^1 respectively. By the binding property of the bit-commitment scheme, V knows at most one of z^0 and z^1 , and so can only get one of e and \widehat{y} at most. Specifically, Yao's garbling scheme is used to achieve the goal that P hides $\widehat{y} = \text{En}(e, y)$ and e respectively with z^0 and z^1 .

Our aim is to construct a 3-round protocol, so 2-round completely extractable perfectly-hiding commitment scheme is needed. Recall the bit-commitment scheme $\langle S, R \rangle$ with extraction algorithm Ext based on trapdoor generalized claw-free functions $TGCF$ (Construction 4.2). In commit stage,

- The receiver runs $(s, tr) \leftarrow \mathcal{S}(1^n)$, and sends s to the sender.
- The sender computes $(c, z_b) = Com(s, b)$, where $c = F(s, b, x_b) = f_s^b(x_b)$, $z_b = T(s, b, x_b) = t_s^b(x_b)$, and sends c to the receiver.

In open stage, the sender reveals b, z_b to the receiver. Assume $|z_b| = q(n)$, where $q(n) = poly(n)$.

Construction 5.2: 3-Round ZKP for L

Common input: $x \in L$.

P 's auxiliary input: witness $y = y_1 \cdots y_\ell$ for $x \in L$.

- P (as the receiver R) randomly runs $(s, tr) \leftarrow \mathcal{S}(1^n)$ and sends s to V .
- V commits to random challenge, proceeds as follows:
 - Randomly pick $\sigma \in \{0, 1\}$, $r \in \{0, 1\}^{poly(n)}$, and computes $(c, z) = Com(s, \sigma; r)$, where $z = z_1 \cdots z_q \in \{0, 1\}^q$. Assume $q \geq \ell$.
 - Send c to P .
- After receiving c , P first verifies c . If the verification passes, P proceeds as follows:
 - Extract $z^0, z^1 \in \{0, 1\}^q$ with tr from c : $(z^0, z^1) = Ext(tr, c)$.
 - Garble $C_{L,x}$: Run $(\widehat{C}_{L,x}, e, d) \leftarrow Gb(C_{L,x}, 1^n)$, where $z^b = z_1^b \cdots z_q^b$, $b = 0, 1$, and

$$e = ((K_1^0, K_1^1), \dots, (K_\ell^0, K_\ell^1)), d = \{(0, K^0), (1, K^1)\}$$

- Randomly construct function ψ :

$$\psi(z) = ((Ran_1^0, Ran_1^1), \dots, (Ran_\ell^0, Ran_\ell^1)), z \in \{0, 1\}^q, Ran_i^b \leftarrow_R \{0, 1\}^k$$

such that $\psi(z^0)$ and $\psi(z^1)$ satisfy the following conditions respectively:

1) When $z = z^0$, $Ran_i^b = K_i^b$, $b = 0, 1$, $i = 1, \dots, \ell$.

2) When $z = z^1$, $Ran_i^{z_i} = K_i^{y_i}$, $Ran_i^{1-z_i} \leftarrow_R \{0, 1\}^k$, $i = 1, \dots, \ell$.

Let C_ψ be a circuit to compute ψ .

- Garble C_ψ : Run $(\widehat{C}_\psi, e_\psi, d_\psi) \leftarrow Gb(C_\psi, 1^n)$, where e_ψ and d_ψ are encoding key and decoding list respectively, specifically, $e_\psi = ((\Psi_1^0, \Psi_1^1), \dots, (\Psi_q^0, \Psi_q^1))$.
- Let $\tau = z^0 \oplus z^1$. Set $\bar{e}_\psi = ((\bar{\Psi}_1^{\tau_1}, \bar{\Psi}_1^{1-\tau_1}), \dots, (\bar{\Psi}_q^{\tau_q}, \bar{\Psi}_q^{1-\tau_q}))$.
- Finally, send $(\widehat{C}_{L,x}, d)$ and $(\widehat{C}_\psi, \bar{e}_\psi, d_\psi)$ to V .
- Receiving $(\widehat{C}_{L,x}, d)$ and $(\widehat{C}_\psi, \bar{e}_\psi, d_\psi)$, V proceeds as follows:
 - Assume $\bar{e}_\psi = ((\bar{\Psi}_1^0, \bar{\Psi}_1^1), \dots, (\bar{\Psi}_q^0, \bar{\Psi}_q^1))$. Compute $\widehat{K} = \widehat{C}_\psi(\bar{\Psi}_1^{z_1}, \dots, \bar{\Psi}_q^{z_q})$, $\bar{K} = De(d_\psi, \widehat{K})$.
 - Assume $\bar{K} = ((\Gamma_1^0, \Gamma_1^1), \dots, (\Gamma_\ell^0, \Gamma_\ell^1))$. When $\sigma = 0$, verify $\widehat{C}(\Gamma_1^{z_1}, \dots, \Gamma_\ell^{z_\ell}) = K^1$; When $\sigma = 1$, verify $Ve(C_{L,x}, \widehat{C}, \bar{K}, d) = 1$. V accepts if and only if the verification passes.

Theorem 3. $\langle S, R \rangle$ is a two-round completely extractable perfectly-hiding commitment scheme, and Garble = (Gb, En, Ev, De, Ve) is Yao's garbled circuit scheme, then Construction 5.2 is the (augmented black-box) zero knowledge proof system for $L \in NP$.

Proof. Completeness: Assume $x \in L$, then $C_{L,x}(y) = 1$. If V honestly computes $(c, z) = Com(s, \sigma, r)$, then $z = z^\sigma$. So,

$$(\bar{\Psi}_1^{z^1}, \dots, \bar{\Psi}_q^{z^q}) = En(\bar{e}_\psi, z) = \begin{cases} (\Psi_1^{z^0}, \dots, \Psi_q^{z^0}) = En(e_\psi, z^0), \sigma = 1 \\ (\Psi_1^{z^1}, \dots, \Psi_q^{z^1}) = En(e_\psi, z^1), \sigma = 0 \end{cases}$$

V computes

$$\hat{K} = \hat{C}_\psi(\bar{\Psi}_1^{z^1}, \dots, \bar{\Psi}_q^{z^q}) = \begin{cases} \hat{C}_\psi(En(e_\psi, z^0)), \sigma = 1 \\ \hat{C}_\psi(En(e_\psi, z^1)), \sigma = 0 \end{cases}$$

By correctness of Yao's garbling scheme, we have

$$\bar{K} = Dec(d_\psi, \hat{K}) = ((\Gamma_1^0, \Gamma_1^1), \dots, (\Gamma_\ell^0, \Gamma_\ell^1)) = \begin{cases} \psi(z^1), \sigma = 0 \\ \psi(z^0), \sigma = 1 \end{cases}$$

By definition of ψ , $\Gamma_i^{z^0} = K_i^{y_i}$ when $\sigma = 0$, and $\Gamma_i^b = K_i^b$ when $\sigma = 1$, for $i = 1, \dots, \ell$ and $b = 0, 1$. So, we have

$$\hat{C}(\Gamma_1^{z^0}, \dots, \Gamma_\ell^{z^0}) = \hat{C}(K_1^{y_1}, \dots, K_\ell^{y_\ell}) = K_t^1, \sigma = 0$$

$$Ve(C_{L,x}, \hat{C}_{L,x}, \bar{K}, d) = Ve(C_{L,x}, \hat{C}_{L,x}, e, d) = 1, \sigma = 1$$

Therefore, V always accepts.

Soundness: IF $x \notin L$, there does not exist y such that $R_{L,x}(y) = 1$. Since V commits to σ by $\langle S, R \rangle$ that is perfectly hiding, P cannot get V 's challenge σ . So, it holds that either $\hat{C}(\Gamma_1^{z^1}, \dots, \Gamma_\ell^{z^q}) \neq K_t^1$, or $Ve(C_{L,x}, \hat{C}, \bar{K}, d) \neq 1$. In other words, the probability that V accepts the proof is no more than $1/2$.

Zero Knowledge: Intuitively, the assumption that $\langle S, R \rangle$ is computationally-binding ensures that V knows at most one of z^0 and z^1 except negligible probability. This means that the protocol is zero knowledge. To formally explain it, define an (ABB) simulator Sim for any V^* with auxiliary input aux as follows: (Here, V^* is an augmented verifier)

$Sim(x, C_{L,x})$ proceeds as follows:

- Randomly select r_V used as the random input of V^* .
- Run $(s, tr) \leftarrow \mathcal{S}(1^n)$. Invoke V^* with s and receive c . If V^* aborts or c is not correct, outputs (r_V, s) and stop.
- Else, receive $state$ from the private output tape and compute $(z^0, z^1) = Ext(c, tr)$. Search (σ, z) in $state$ (and auxiliary input aux) satisfying $z \in \{z^0, z^1\}$ ($z = z^\sigma$). If no such (σ, z) is found, randomly select $\sigma \in \{0, 1\}$. (Augmented verifier V^* write (σ, z) on the private output tape if knowing).
- Let $C'_{L,x}$ be the same as $C_{L,x}$ except that its output gate g'_i is set as $g'_i(a, b) \equiv 1$ for $a, b \in \{0, 1\}$.
- Garble $C_{L,x}$ or $C'_{L,x}$: Run $(\hat{C}, e, d) \leftarrow Gb(C, 1^n)$, where $C = C'_{L,x}$ if $\sigma = 0$, or $C = C_{L,x}$ otherwise. Specifically, assume

$$e = \{(K_i^0, K_i^1)\}_{i=1}^\ell, d = \{(0, K^0), (1, K^1)\}$$

- Randomly construct function ψ' :

$$\psi'(z) = ((\text{Ran}_1^0, \text{Ran}_1^1), \dots, (\text{Ran}_\ell^0, \text{Ran}_\ell^1)), z \in \{0, 1\}^q, \text{Ran}_i^b \leftarrow_R \{0, 1\}^k$$

such that the following conditions hold:

1) If $\sigma = 0$, $\psi'(z^0) = ((K_1^0, K_1^1), \dots, (K_\ell^0, K_\ell^1))$.

2) If $\sigma = 1$, $\psi'(z^1) = ((\text{Ran}_1^0, \text{Ran}_1^1), \dots, (\text{Ran}_\ell^0, \text{Ran}_\ell^1))$ satisfies $\text{Ran}_i^{z_i^0} = K_i^{u_i}$, $u_i \leftarrow_R \{0, 1\}$, $i = 1, \dots, \ell$. Let $u = u_1 \cdots u_\ell$.

Let $C_{\psi'}$ be a circuit to compute ψ' .

- Garble $C_{\psi'}$: Run $(\widehat{C}_{\psi'}, e_{\psi'}, d_{\psi'}) \leftarrow Gb(C_{\psi'}, 1^n)$, where $e_{\psi'}$ and $d_{\psi'}$ are encoding key and decoding list respectively, specifically, $e_{\psi'} = ((\Psi_1^0, \Psi_1^1), \dots, (\Psi_\ell^0, \Psi_\ell^1))$.
- Let $\tau = z^0 \oplus z^1$. Set $\bar{e}_{\psi'} = ((\Psi_1^{\tau_1}, \Psi_1^{1-\tau_1}), \dots, (\Psi_q^{\tau_q}, \Psi_q^{1-\tau_q}))$.
- Output $(x, C_{L,x}, r_V, s, c, (\widehat{C}, d), (\widehat{C}_{\psi'}, \bar{e}_{\psi'}, d_{\psi'}))$.

First, notice $\Pr[\text{Sim aborts}] = \Pr[P \text{ aborts}]$, so assume *Sim* never aborts.

To show that the distributions $\text{Sim}(x, C_{L,x})$ is indistinguishable from $\text{View}_{V^*}^P(x)$, we introduce hybrid prover P' . P' interacts with V^* and can obtain σ in the same way as *Sim* (reading V 's private output tape or randomly selecting). In $\langle P', V^* \rangle$, P' is the same as P except that it generates ψ' as *sim* does but replaces $\text{Ran}_i^{z_i^0} = K_i^{u_i}$ with $\text{Ran}_i^{z_i^0} = K_i^{y_i}$, $i = 1, \dots, \ell$. The only difference between *Sim* and P' is that *Sim* replaces computing $(\widehat{C}, e, d) \leftarrow Gb(C'_{L,x}, 1^n)$ with computing $(\widehat{C}_{L,x}, e, d) \leftarrow Gb(C_{L,x}, 1^n)$ when $\sigma = 0$. Notice that $C_{L,x}$ and $C'_{L,x}$ have the same structure and satisfy $C_{L,x}(y) = C'(u) = 1$, by the privacy of Yao's garbling scheme, $(\widehat{C}, d, \text{En}(e, u))$ and $(\widehat{C}_{L,x}, d, \text{En}(e, y))$ are indistinguishable. This implies that $(\widehat{C}, d, (\widehat{C}_{\psi'}, d_{\psi'}))$ and $(\widehat{C}_{L,x}, d, (\widehat{C}_{\psi'}, d_{\psi'}))$ are indistinguishable when $\sigma = 0$, since $\psi'(z)$ is independent of $(\widehat{C}_{L,x}, d)$ when $z \neq z^0$. In case of $\sigma = 1$, *Sim* is same as P' . Therefore, we have that $\text{Sim}(x, C_{L,x})$ is indistinguishable from $\text{View}_{V^*}^{P'}(x)$.

The only difference between P' and P is that $\psi'(z^{(1-\sigma)})$ is different from $\psi(z^{(1-\sigma)})$. Specifically, $\psi(z^{(1-\sigma)})$ is related to e and $\psi'(z^{(1-\sigma)})$ is independent of e . By the binding property of the commitment scheme and the authenticity of the garbling scheme, the probability that v^* knows $z^{(1-\sigma)}$ is negligible. Notice that $\psi'(z^{(1-\sigma)})$ and $\psi(z^{(1-\sigma)})$ are randomly selected. So, $\text{View}_{V^*}^{P'}(x)$ and $\text{View}_{V^*}^P(x)$ are computationally indistinguishable.

The above in all, $\text{Sim}(x, C_{L,x})$ is computationally indistinguishable from $\text{View}_{V^*}^P(x)$. ■

Since ABB simulator *Sim* does not rewind V^* , so it can run in parallel to simulate the paralleled protocol. This means that the presented 3-round ZKP is ZK under parallel composition. Therefore, we can obtain 3-round ZKP with negligible error probability by paralleling Construct 5.2.

Theorem 4. *If two-round completely extractable statistically hiding commitment schemes exist and Yao's garbled circuit scheme is secure, then there exist 3-round augmented back-box zero knowledge systems for any $L \in NP$.*

It is easy to see that Construction 5.2 is not BBZK, since the black-box simulator cannot obtain V 's challenge σ by rewinding V . However, if we add one round at the end of Construction

5.2 for the verifier to open the commitment c honestly, i.e. the verifier must reveal c honestly after receiving $(\widehat{C}_{L,x}, d)$ and $(\widehat{C}_\psi, \bar{e}_\psi, d_\psi)$, the protocol is BBZK, since the black-box simulator can reconstruct $View_{V^*}^P(x)$ for any malicious V^* by rewinding V^* . Furthermore, by the black-box simulation technique from [28], the 4-round protocol resulting from Construction 5.2 by adding one round for the verifier to open the commitment c is in fact parallel black-box simulation zero knowledge (so it also is augmented black-box ZK). This in a sense shows that Construction 5.2 is parallel ZK under ABB simulation, since adding one round at the end of protocol for the verifier to open the commitment c does not damage the zero knowledge property.

6 2-Round ZK protocol

In this section we focus on the existence of 2-round ZK protocols for NP, and present a 2-round ZKA for NP and a 2-round ZKP for QNR respectively.

6.1 ZKA for NP

In this subsection, we present a 2-round ZKA for NP. Our tool is indistinguishable obfuscator. Let \mathcal{O} be an indistinguishable obfuscator, $G : \{0, 1\}^n \rightarrow \{0, 1\}^{2n \cdot p(n)}$ be a pseudorandom generator, where n is security parameter, $p(n)$ is the length of random input of \mathcal{O} . Let $\{H_s\}_{s \in I}$ be a family of pseudorandom functions with sampling algorithm $s \leftarrow \text{Sample}(1^n, r)$.

Let $L \in NP$, the corresponding binary relation is denoted by $R_L(\cdot, \cdot)$. Let $R_0(\cdot, \cdot) \equiv 0$ and $|R_0| = |R_L|$. For any instance x , define a function $R_{L,x}^b(u)$ as follows:

$$R_{L,x}^b(u) = \begin{cases} R_L(x, u), & b = 1 \\ R_0(x, u), & b = 0 \end{cases}$$

Notice that $R_{L,x}^1(u) = R_{L,x}^0(u) \equiv 0$ when $x \notin L$, and there exists w such that $R_{L,x}^1(w) = 1$ when $x \in L$. This property implies instance-dependent commitment. For a given instance x , one computes $C = \mathcal{O}(R_{L,x}^b; r)$ and uses it as the commitment to b , where r is randomly selected. When $x \in L$, the commitment is perfect binding and b is revealed by the witness of x . When $x \notin L$, the commitment is computationally hiding.

The structure of our construction is as follows: To prove $x \in L$, the verifier first selects σ randomly and commit to it using the above commitment scheme, and then asks the prover to guess σ . The verifier accepts if and only if the prover's guess is correct. Obviously, this 2-round protocol is only honest verifier zero knowledge. To obtain 2-round zero knowledge protocol, we consider the parallel protocol, where V is asked to compute $C_i = \mathcal{O}(R_{L,x}^{\sigma_i}; r_i)$, $i = 1, \dots, n$ and sends (C_1, \dots, C_n) to P . To prevent V from cheating, we require that $\sigma = \sigma_1 \cdots \sigma_n$ is related to r_1, \dots, r_n . Specifically, V needs to select $\sigma = \sigma_1 \cdots \sigma_n$, $\delta = \delta_1 \cdots \delta_n \in \{0, 1\}^n$, and then computes $C_i = \mathcal{O}(R_{L,x}^{\sigma_i}; r_i^{\delta_i})$, where $r_1 \cdots r_n \leftarrow G(\sigma)$, $r_i = (r_i^0, r_i^1)$.

Construction 6.1: 2-Round ZKA for L

The Common input: $x \in L$. The prover's auxiliary input: witness w for $x \in L$.

- The verifier proceeds as follows:

1. Pick $\sigma = \sigma_1 \cdots \sigma_n \in_R \{0, 1\}^n$ and $\delta = \delta_1 \cdots \delta_n \in_R \{0, 1\}^n$, compute $r_1 \cdots r_n = G(\sigma)$, where $r_i = (r_i^0, r_i^1) \in \{0, 1\}^{2p(n)}$, $i = 1, \dots, n$.
 2. Compute $C_i = \mathcal{O}(R_{L,x}^{\sigma_i}; r_i^{\delta_i})$ and sends $\{C_i\}_{i=1}^n$ to the prover.
- After receiving $\{C_i\}_{i=1}^n$, the prover proceeds as follows:
1. Verify $\{C_i\}_{i=1}^n$ satisfying the following conditions:
 - $\sigma_i = C_i(x, w) \in \{0, 1\}$, $i = 1, \dots, n$. Set $\sigma = \sigma_1 \cdots \sigma_n$.
 - For any i , $\exists \delta_i$, $C_i = \mathcal{O}(R_{L,x}^{\sigma_i}; r_i^{\delta_i})$, where $r_1 \cdots r_n \leftarrow G(\sigma)$, $r_i = (r_i^0, r_i^1)$. Set $\delta = \delta_1 \cdots \delta_n$.
 2. If the verification fails, randomly select $\sigma, \delta \in_R \{0, 1\}^n$.
 3. Randomly select $r \in \{0, 1\}^n$, and compute $s \leftarrow \text{Sample}(1^n; r \oplus \sigma)$.
 4. Send $r, a = H_s(\delta)$ to the verifier.
- Receiving (r, a) , the verifier computes $s \leftarrow \text{Sample}(1^n; r \oplus \sigma)$ and verifies $a = H_s(\delta)$. The verifier accepts iff $a = H_s(\delta)$.

Theorem 5. *Let \mathcal{O} be an indistinguishable obfuscator, G be a pseudorandom generator and $\{H_s\}_s$ be PRF. Construction 6.1 is a zero knowledge argument system for NP.*

Proof. **Completeness:** This is obvious.

Soundness: When $x \notin L$, it holds that $R_{L,x}^1(u) = R_{L,x}^0(u) \equiv 0$ for any u . By the assumption that \mathcal{O} is an indistinguishable obfuscator, $\mathcal{O}(R_{L,x}^0; r_i)$ and $\mathcal{O}(R_{L,x}^1; r_i)$ are indistinguishable. This means that the prover cannot get $\sigma = \sigma_1 \cdots \sigma_n$ and $\delta = \delta_1 \cdots \delta_n$ from $\{C_i\}_{i=1}^n$ except for a negligible probability. So, the probability that the prover find r such that $a = H_s(\delta)$ (where $s \leftarrow \text{Sample}(1^n; r \oplus \sigma)$) must be negligible, that is,

$$\Pr[\langle P, V \rangle(x) = 1] = \text{negl}(n)$$

Zero knowledge: Intuitively, if V^* does not follow the protocol to generate $\{C_i\}_{i=1}^n$, that is, V^* has no idea about σ and δ , then (r, a) that V^* received from P is random (when P 's verification does fail) or indistinguishable from random one (when V^* gets correct $\{C_i\}_{i=1}^n$ from its auxiliary input z). For any V^* with the auxiliary input z , the augmented black-box simulator Sim is as follows:

$Sim(x, z)$ proceeds as follows:

- Select uniformly random input r_V for the verifier V^* .
- Invoke V^* and obtain its return $\{C_i\}_{i=1}^n$.
- Find $\sigma' = \sigma'_1 \cdots \sigma'_n$ in V^* 's private output tape and auxiliary input z , compute $r_1 \cdots r_n = G(\sigma')$, where $r_i = (r_i^0, r_i^1)$. Get δ'_i satisfying $C_i = \mathcal{O}(R_{L,x}^{\sigma'_i}; r_i^{\delta'_i})$. $\sigma' = \sigma'_1 \cdots \sigma'_n$, $\delta' = \delta'_1 \cdots \delta'_n$. If there are no such σ' or δ' , randomly select $\sigma', \delta' \in \{0, 1\}^n$.
- Randomly pick ρ' and compute $s' \leftarrow \text{Sample}(1^n, \rho' \oplus \sigma')$, $a' = H_{s'}(\delta')$.
- Outputs $(r_V, \{C_i\}_{i=1}^n, \rho', a')$.

Note that $Sim(x, z)$ is different from P only when V^* has sent correct $\{C_i\}_{i=1}^n$ but has no idea about the corresponding σ and δ . In such case, P can find the corresponding σ and δ , while $Sim(x, z)$ randomly selects σ' and δ' . Anyhow s' and s are always indistinguishable. Therefore, by the assumption that $\{H_s\}$ is a PRF, $(\text{ran}, \{C_i\}_{i=1}^n, \rho', a')$ is indistinguishable from the V^* 's view $(\text{ran}, \{C_i\}_{i=1}^n, \rho, a)$.

6.2 ZKP for QNR

We show how to transform the basic 2-round proof protocol for quadratic non-residue (QNR) [34], which is only an honest verifier ZKP, into a ZKP protocol without adding the round complexity.

QR (quadratic residue) and QNR are as follows:

$$QR = \{(n, x) : n \in \mathbf{N}, x \in \mathbf{Z}_n^*, \exists u \in \mathbf{Z}_n^*, u^2 = x \pmod{n}\}$$

$$QNR = \{(n, x) : n \in \mathbf{N}, x \in \mathbf{Z}_n^*, \forall u \in \mathbf{Z}_n^*, u^2 \neq x \pmod{n}\}$$

It is well known that $QR \in NP \cap CoNP$, $QNR \in NP \cap CoNP$. When n is a composite (of unknown factorization), it is believed that to determine $(n, x) \in QR$ or $(n, x) \in QNR$ for a given integer x is infeasible. Let $\ell = |n|$ be security parameter. The basic interactive proof for $(n, x) \in QNR$ is as follows [34]:

- V randomly selects $b \in \{0, 1\}$, $r \in \mathbf{Z}_n^*$ and sends $w = r^2 x^b$ to P
- P checks w . Sets $b' = 1$ if $w \in QNR$ and $b' = 0$ otherwise. Finally, sends b' to V

It is known that the above construction can be transformed into a zero knowledge proof via adding an interactive process, by which the verifier proves itself honestly executing the protocol [34]. Consequently, the known ZKP for QNR is at least 4-round.

The verifier is honest if and only if it holds r that satisfies $w = r^2 x^b$. When $x \in QNR$, r is determined by $w = r^2 x^b$ and the prover can extract it after receiving w . To prove $x \in QNR$, the prover can show r (extracted from w) instead of b in the second round. Obviously, this protocol is only honest verifier zero knowledge. If the prover can show that he knows r (implied by w) in a special manner such that only the honest verifier holding r can verify it, that is, a cheating verifier without holding r cannot get any information about r , then the protocol will be zero knowledge. Based on this idea, we construct a 2-round ZKP for QNR .

The main tools in our construction are Yao's garbled circuits scheme and point-functions. For any $\alpha \in \{0, 1\}^\ell$, let $I_\alpha : \{0, 1\}^\ell \rightarrow \{0, 1\}$ be a point-function defined as

$$I_\alpha(u) = \begin{cases} 1, & u = \alpha \\ 0, & u \neq \alpha \end{cases}$$

and $C : \{0, 1\}^\ell \rightarrow \{0, 1\}$ be an acyclic boolean circuit computing I_α . Assume that C has m gates (the output gate is denoted as g_m) and $t = \ell + m$ wires, denoted as w_1, \dots, w_t , where w_1, \dots, w_ℓ are the input wires and w_t is the output wire of C .

Our protocol uses Yao's garbled circuits scheme, formally denoted by

$$Garble = (Gb, En, Ev, De, Ve)$$

with the following changes:

- 1) Let (G, E, D) be the private key encryption scheme used in Yao's garbled circuits scheme. We require (G, E, D) to satisfy the condition that the ciphertext under one key is also a ciphertext under another key. But Yao's garbled circuits scheme requires an encryption under one key will not fall in the range of an encryption under another key except a negligible probability.

- 2) Let w_i, w_j and w_o be two input wires and one output wire of gate g respectively, and the corresponding keys are $(K_i^0, K_i^1), (K_j^0, K_j^1)$ and (K_o^0, K_o^1) . The garbled gate \hat{g} is $(c_{0,0}, c_{0,1}, c_{1,0}, c_{1,1})$ ("garbled computation table"), where

$$c_{a,b} = E_{K_i^a} \left(E_{K_j^b} \left(K_o^{g(a,b)} \right) \right) \quad (a, b = 0, 1)$$

In Yao's garbled circuits scheme, the garbled gate of g is the random ordering of $(c_{0,0}, c_{0,1}, c_{1,0}, c_{1,1})$.

- 3) Because of 1), evaluating garbled circuit needs not only the coding of input x but also x . In addition, the truth table of each gate in the circuit must be known. Let $(\hat{C}, e, d) \leftarrow Gb(C)$. On inputting $x = x_1 \cdots x_\ell$ and $\hat{x} = (K_1^{x_1}, \dots, K_\ell^{x_\ell}) = En(e, x)$, $Ev(\hat{C}, x, \hat{x})$ computes each gate according to its "garbled computation table", gate by gate, and obtains the output of \hat{C} , $\hat{y} = Ev(\hat{C}, x, \hat{x})$.

For any gate g , let w_i, w_j and w_o be two input wires and one output wire of g respectively. According to 2), the "garbled computation table" of \hat{g} is $(c_{0,0}, c_{0,1}, c_{1,0}, c_{1,1})$, where $c_{a,b} = E_{K_i^a} \left(E_{K_j^b} \left(K_o^{g(a,b)} \right) \right)$ $(a, b = 0, 1)$. To evaluate the garbled gate \hat{g} with (a, K_i^a) (the input of w_i and the corresponding key) and (b, K_j^b) (the input of w_j and the corresponding key), we first decrypt $c_{a,b}$ with (K_i^a, K_j^b) to get $K_o^{g(a,b)}$ and then obtain $g(a, b)$ from the truth table of g . Finally, $(g(a, b), K_o^{g(a,b)})$ is used as the output of \hat{g} . If $g = g_m$ is the output gate, $K_o^{g(a,b)}$ is the output of \hat{C} .

- 4) For $(\hat{C}, e, d) \leftarrow Gb(C)$ and any $x, x' \in \{0, 1\}^\ell$, $Ev(\hat{C}, x', \hat{x} = En(e, x))$ can be computed according to the above method whether there is $x' = x$ or not. \hat{x} contains keys used for decryption, while x' decides which ciphertext to be decrypted. Therefore, to compute $Ev(\hat{C}, x', \hat{x})$ with $x' \neq x$ means to decrypt some ciphertext encrypted by key K with key $K' \neq K$. So, the probability that $Ev(\hat{C}, x', \hat{x})$ outputs (b, K_m^1) is negligible when $x' \neq x$ and (G, E, D) is secure. For this reason, we modify the decoding algorithm De by setting $De(d, \hat{y}) = 0$ when $y \neq K_m^1$, where $d = \{(0, K_m^0), (1, K_m^1)\}$.

In addition to, we introduce an algorithm $ReOrder$ which takes \hat{C} and $r = r_1 \cdots r_\ell \in \{0, 1\}^\ell$ as inputs and output a new garbled circuit \bar{C} , where $(\hat{C}, e, d) \leftarrow Gb(C)$. Concretely, $ReOrder$ re-orders the garbled computation table of each gate $\hat{g} = (c_{0,0}, c_{0,1}, c_{1,0}, c_{1,1})$ if necessary, when g has at least one input wire in $\{w_1, \dots, w_\ell\}$.

$ReOrder(\hat{C}, r)$:

- For each garbled gate $\hat{g} \in \hat{C}$, reorder $\hat{g} = (c_{0,0}, c_{0,1}, c_{1,0}, c_{1,1})$ as follows: Let w_i, w_j be two input wires of g , set

$$c'_{0,0} := c_{r_i, r_j}, c'_{0,1} := c_{r_i, 1 \oplus r_j}, c'_{1,0} := c_{1 \oplus r_i, r_j}, c'_{1,1} := c_{1 \oplus r_i, 1 \oplus r_j}$$

where $r_{\ell+1} = \dots = r_{t-1} = 0$. Set $\bar{g} = (c'_{0,0}, c'_{0,1}, c'_{1,0}, c'_{1,1})$. Notice that $\bar{g} = \hat{g}$ when $r_i = r_j = 0$. Finally, set $\bar{C} = \{\bar{g}\}_{\hat{g} \in \hat{C}}$.

- Output \bar{C} .

Since (G, E, D) satisfies the condition that the ciphertext under one key is also a ciphertext under another key, $\hat{C} = ReOrder(\bar{C}, r)$ completely hides r . In addition, assume that C

computes I_α and $(\widehat{C}, e, d) \leftarrow Gb(C)$, where $e = \{(K_i^0, K_i^1)\}_{i=1}^\ell$. For any $r \in \{0, 1\}^\ell$, let $\overline{C} = ReOrder(\widehat{C}, r)$ and $\overline{e} = \{(\overline{K}_i^0, \overline{K}_i^1)\}_{i=1}^\ell$, where $\overline{K}_i^0 = K_i^{r_i}$, $\overline{K}_i^1 = K_i^{1 \oplus r_i}$. Then, $(\overline{C}, \overline{e}, d)$ is in fact a garbled circuit that computes the point-function $\overline{I}(u) = I_\alpha(u \oplus r)$. Furthermore, from $\overline{K}_i^{u_i \oplus r_i} = K_i^{u_i}$, $i = 1, \dots, \ell$, it is known that $En(\overline{e}, u \oplus r) = En(e, u)$. So the following lemma holds.

Lemma 1. Assume $(\widehat{C}, e, d) \leftarrow Gb(C)$, where $e = \{(K_i^0, K_i^1)\}_{i=1}^\ell$. For any $r \in \{0, 1\}^\ell$, let $\overline{C} = ReOrder(\widehat{C}, r)$ and $\overline{e} = \{(\overline{K}_i^0, \overline{K}_i^1)\}_{i=1}^\ell$, where $\overline{K}_i^0 = K_i^{r_i}$, $\overline{K}_i^1 = K_i^{1 \oplus r_i}$. Then it holds that

$$\overline{C}(u \oplus r, En(\overline{e}, u \oplus r)) = \widehat{C}(u, En(e, u)), \quad u \in \{0, 1\}^\ell$$

Let $Gen : \{0, 1\}^k \rightarrow \{0, 1\}^{2\ell k}$ be a pseudorandom generator, where k is the length of the keys of (G, E, D) .

Construction 6.2: 2-round ZKP for QNR

Common inputs $x \in QNR$ and security parameter ℓ .

- The verifier V proceeds as follows:
 - Randomly selects $r \in \mathbf{Z}_n^*$, $b \in \{0, 1\}$ and computes $w = r^2 x^b$. Let $r = r_1 \cdots r_\ell$.
 - Randomly selects $\alpha = \alpha_1 \cdots \alpha_\ell \in \{0, 1\}^\ell$ and constructs a circuit C_α to compute point-function $I_\alpha(\cdot)$
 - Sends $w, (\alpha, C_\alpha)$ to the prover P .
- After receiving $(w, (\alpha, C_\alpha))$, P proceeds as follows:
 - Finds s, b such that $w = s^2 x^b$, where $s = s_1 \cdots s_\ell$. If no such s exists, randomly selects $s \in \mathbf{Z}_n^*$.
 - Randomly selects $K', K \in \{0, 1\}^\ell$, and generates ℓ pairs of keys $\{(K_i^0, K_i^1)\}_{i=1}^\ell$ using Gen , i.e. $Gen(K) = \{(K_i^0, K_i^1)\}_{i=1}^\ell$.
 - Garble C_α : $(\widehat{C}_\alpha, e, d) \leftarrow Gb(C_\alpha)$, such that $e = \{(K_i^0, K_i^1)\}_{i=1}^\ell$, $d = \{(0, K'), (1, K)\}$.
 - $\overline{C} = ReOrder(\widehat{C}_\alpha, s)$, $\overline{e} = \{(K_i^{s_i}, K_i^{1 \oplus s_i})\}_{i=1}^\ell$.
 - Sends \overline{C} and $K_\alpha = (K_1^{\alpha_1}, \dots, K_\ell^{\alpha_\ell})$ to V .
- Receiving $(\overline{C}, K_\alpha = (K_{\alpha,1}, \dots, K_{\alpha,\ell}))$, V verifies P 's response.
 - Computes $\overline{K} = \overline{C}(\alpha \oplus r, K_\alpha)$.
 - Computes $\{(\overline{K}_i^0, \overline{K}_i^1)\}_{i=1}^\ell = Gen(\overline{K})$.
 - $(\widehat{C}, e) = ReOrder(\overline{C}, r)$.
 - Accepts iff 1) $\overline{K}_i^{\alpha_i} = K_{\alpha,i}$ and 2) $\forall e \left(C_\alpha, \widehat{C}, \{(\overline{K}_i^0, \overline{K}_i^1)\}_{i=1}^\ell, d \right) = 1$.

We first informally argue that V , even if it cheats, learns nothing else from P . The only message sent by P is (\overline{C}, K_α) , where \overline{C} is in fact a garbled circuit of point-function $I_{\alpha \oplus s}$ and K_α (a set of randomly selected keys) is the random code of $\alpha \oplus s$. V computes $\overline{K} = \overline{C}(\alpha \oplus s, K_\alpha)$ and then verifies it. If V is honest, (\overline{C}, K_α) (garbled circuit of point-function $I_{\alpha \oplus s}$) reveals nothing since V knows α and $s = r$. When V does not know r satisfying $w = r^2 x^b$, it can only execute $\overline{C}(\beta, K_\alpha)$ with any incorrect input $\beta \neq \alpha \oplus s$, and so only gets a random key except for a negligible probability.

Theorem 6. *Let $Garble = (Gb, En, Ev, De, Ve)$ be Yao's garbled circuits scheme. Construction 6.2 is a zero knowledge proof system for QNR .*

Proof. **Completeness:** When the protocol is executed correctly, $\overline{C}(\alpha \oplus r, K_\alpha) = \widehat{C}(\alpha, K_\alpha) = K$ since $s = r$, so the verifier always accepts.

Soundness: When $x \in QR$, there exist $r' \neq r''$ such that $w = (r')^2 = (r'')^2 x$, P cannot determine which one is chosen by V . So, for any \overline{C} and K_α , at least one of $\overline{K} = \overline{C}(\alpha \oplus r', K_\alpha)$ and $\overline{K} = \overline{C}(\alpha \oplus r'', K_\alpha)$ cannot pass V 's verification, except for a negligible probability. Therefore, the probability that V accepts $x \in QNR$ is no more than $\frac{1}{2}$.

Zero Knowledge: For any V^* , we define an augmented black-box simulator Sim . On inputting $x \in QNR$, $Sim(x)$ proceeds as follows:

- Selects random input r_V for V^* .
- Receives w and C_α, α from V^* , where C_α computes I_α .
- Checks V^* 's private output $state$, and finds s' and b such that $w = (s')^2 x^b$. If no such s' can be found in $state$ (this means V^* does not know s' or such s' does not exist), Sim randomly picks $s' \in \{0, 1\}^\ell$.
- Randomly selects $K', K \in \{0, 1\}^\ell$, and generates ℓ pairs of keys $\{(K_i^0, K_i^1)\}_{i=1}^\ell$ using Gen , i.e. $Gen(K) = \{(K_i^0, K_i^1)\}_{i=1}^\ell$.
- Garble C_α : $(\widehat{C}, e, d) \leftarrow Gb(C_\alpha)$, such that $e = \{(K_i^0, K_i^1)\}_{i=1}^\ell$, $d = (K^0, K^1)$.
- $\overline{C}' = ReOrder(\widehat{C}, s')$, $\overline{e}' = \left\{ \left(K_i^{s'_i}, K_i^{1 \oplus s'_i} \right) \right\}_{i=1}^\ell$.
- Output $(x, R_V, \overline{C}', K'_\alpha = (K_1^{\alpha_1}, \dots, K_\ell^{\alpha_\ell}))$.

Notice that Sim is different from P only when there exists r such that $w = r^2 x^b$ but V^* does not know r . In this case, Sim generates $(\overline{C}', \overline{e}')$ with randomly selected s' while P generates $(\overline{C}, \overline{e}, d)$ with s satisfying $w = s^2 x^b$. It is known that $(\overline{C}', \overline{e}', d)$ is a garbled circuit computing $I_\alpha(u \oplus s')$ and $(\overline{C}, \overline{e}, d)$ is a garbled circuit computing $I_\alpha(u \oplus s)$. By lemma 1, we have that

$$\overline{C}(u, K_\alpha) = \overline{C}(u, En(\overline{e}, u)) = \widehat{C}(u \oplus s, En(e, u \oplus s))$$

$$\overline{C}'(u, K'_\alpha) = \overline{C}'(u, En(\overline{e}', u)) = \widehat{C}(u \oplus s', En(e, u \oplus s'))$$

where $(\widehat{C}, e, d) \leftarrow Gb(C_\alpha)$. Therefore, when $u \neq \alpha \oplus s$ or $u \neq \alpha \oplus s'$, both $\overline{C}(u, K_\alpha)$ and $\overline{C}'(u, K'_\alpha)$ are the encoding of 0. $\widehat{C} = ReOrder(\overline{C}, s)$ and $\widehat{C}' = ReOrder(\overline{C}', s')$ completely hide s and s' respectively. So, by the privacy of $Garble = (Gb, En, Ev, De, Ve)$, any PPT distinguisher without knowing s, s' can not distinguish $(\overline{C}', K'_\alpha = (K_1^{\alpha_1}, \dots, K_\ell^{\alpha_\ell}))$ from $(\overline{C}, K_\alpha = (K_1^{\alpha_1}, \dots, K_\ell^{\alpha_\ell}))$. That is, $Sim(x)$ and $View_{V^*}^P(x)$ are computationally indistinguishable. \blacksquare

ABB simulator does not rewind verifier V^* , so the parallel repetition of Construction 6.2 can preserve augmented black-box ZK. That is, n parallel repetitions of Construction 6.2 results in a zero knowledge protocol with error probability 2^{-n} .

References

1. Barak, B.: How to go beyond the black-box simulation barrier. In FOCS 2001, pp. 106-115 (2001)

2. Bitansky, N., Brakerski, Z., Kalai, Y.T., Paneth, O., Vaikuntanathan, V.: 3-message zero knowledge against human ignorance In TCC 2016, pp. 57-83 (2016)
3. Bitansky, N., Canetti, R., Paneth, O., Rosen, A.: On the existence of extractable one-way functions In STOC 2014, pp. 505-514 (2014)
4. Nir Bitansky, Ran Canetti, Alessandro Chiesa, Shafi Goldwasser, Huijia Lin, Aviad Rubinfeld, and Eran Tromer. The hunting of the SNARK. IACR Cryptology ePrint Archive, <http://eprint.iacr.org/2014/580> (2014)
5. Barak, B., Goldreich, O., Impagliazzo, R., Rudich, S., Sahai, A., Vadhan, S.P., Yang, K.: On the (im)possibility of obfuscating programs. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 1-18. Springer, Heidelberg (2001)
6. Barak, B., Lindell, Y., Strict polynomial-time in simulation and extractor. In 34th ACM Symposium on the Theory of Computing, 2002:484-493.
7. Barak, B., Lindell, Y., Vadhan, V.: Lower bounds for non-black-box zero knowledge. Journal of Computer and System Science, 72(2): 321-391, 2006
8. Bellare M, Hoang V T, Rogaway P. Foundations of garbled circuits. In Proceedings of the 2012 ACM conference on Computer and communications security. 2012.; pp. 784-796 (2012).
9. Bellare, M., Jakobsson, M., Yung, M.: Round-optimal zero-knowledge arguments based on any one-way function. In Advances in Cryptology- EUROCRYPT'97. Lecture Notes in Computer Science, vol. 1233 pp. 280-305 (1997)
10. Bitansky, N., Kalai, Y.T., Paneth, O.: Multi-collision resistance: a paradigm for keyless hash functions. Cryptology ePrint Archive, Report 2017/488 (2017), <http://eprint.iacr.org/2017/48> pp. 671-684 (2018)
11. Bellare, M., Palacio, A.: The knowledge-of-exponent assumptions and 3-round zero-knowledge protocols. Proc of Crypto 3152, 273-289 (2004)
12. Bitansky N, Paneth O. From Point Obfuscation To 3-round Zero-Knowledge. IACR Cryptology ePrint Archive, <https://eprint.iacr.org/2011/493>
13. Bitansky, N., Paneth, O.: Point obfuscation and 3-round zero knowledge. In TCC 2012. LNCS, Volume 7194, pages 189-207, 2012.
14. Bitansky, N., Paneth, O.: On the impossibility of approximate obfuscation and application to resettable cryptography. In STOC 2013, pages 241-250.
15. Canetti, R., Dakdouk, R. R.: Extractable perfectly one-way functions. In ICALP, Track C, 2008, pp. 449-460 (2008)
16. Canetti, R., Goldreich, O., Goldwasser, S., Micali, S., Resettable zero-knowledge. In STOC2000, pp:235-244 (2000).
17. Canetti, R., Kilian, J., Petrank, E., Rosen, A.: Black-box concurrent zero-knowledge requires (almost) logarithmically many rounds. SIAM Journal on Computing, 2002, 32(1):1-47.
18. Chung K M , Lin H , Pass R . Constant-Round Concurrent Zero-Knowledge from Indistinguishability Obfuscation. ePrint Archive, Report 2014/991 (In CRYPTO 2015)
19. Chung, K. M., Ostrovsky, R., Pass, R., Visconti, I. Simultaneous Resettable from One-Way Functions. In Foundations of Computer Science, 2013 (FOCS2013), pp: 251-260 (2013)
20. Chung K M , Ostrovsky R , Pass R , et al. 4-Round Resettable-Sound Zero Knowledge. In TCC 2014, pp. 192-216 (2014)
21. Dwork, C., Naor, M., Reingold, O., Stockmeyer, Larry J. Magic functions, In FOCS 1999, pp: 523-534 (1999)
22. Dwork C, Stockmeyer L J. 2-round zero knowledge and proof auditors. In symposium on the theory of computing, 2002 (STOC2002), pp. 322-331.
23. Deng Y , Goyal V , Sahai A . Resolving the Simultaneous Resettable Conjecture and a New Non-Black-Box Simulation Strategy. In FOCS 2009, October 25-27, 2009, Atlanta, Georgia, USA. pp. 251-260 (2009)
24. Feige, U., Shamir, A.: Zero knowledge proofs of knowledge in two rounds. In Advances in Cryptology - CRYPTO'89 Proceedings pp. 526-544 (1989)

25. Feige, U., Shamir, A.: Witness indistinguishable and witness hiding protocols. pp. 416-426 (1990)
26. Fleischhacker, N., Goyal, V., Jain, A.: On the existence of three round zero-knowledge proofs pp. 3-33 (2018)
27. Goldreich, O.. Foundations of cryptography: Basic tools, Cambridge University Press, 2001.
28. Goldreich, O., Kahan, A.: How to construct constant-round zero-knowledge proof systems for np. *Journal of Cryptology* 9(3), pp. 167-189 (1996)
29. Garg, S., Gentry, C., Halevi, S., Raykova, M., Sahai, A., Waters, B.: Candidate indistinguishability obfuscation and functional encryption for all circuits. In FOCS, pp. 40-49 (2013)
30. Goldreich, O., Krawczyk, H.: On the composition of zero-knowledge proof systems. *SIAM Journal on Computing* 25(1), 169-192 (1996)
31. Ganesh, C., Kondi, Y., Patra, A., Sarkar, P.: Efficient adaptively secure zero-knowledge from garbled circuits. In: Abdalla, M., Dahab, R. (eds.): PKC 2018, LNCS 10770, pp. 499-529, 2018.
32. Goldreich, O., Oren, Y.: Definitions and properties of zero knowledge proof systems. *J. Cryptology*, 7(1), 1-32 (1994).
33. Goldreich, O., Micali, S., Wigderson, A.: Proofs that yield nothing but their validity or all languages in np have zero-knowledge proof systems. *Journal of the ACM* 38(3), 690-728 (1991)
34. Goldwasser, S., Micali, S., Rackoff, C.: The knowledge complexity of interactive proof systems. *SIAM Journal on Computing* 18(1), 186-208 (1989)
35. Vipul Goyal, Amit Sahai. Resolving the simultaneous resettability conjecture and a new non-black-box simulation strategy. eprint.iacr.org/2008/545.
36. Hada, S., Tanaka, T.: On the existence of 3-round zero-knowledge protocols. In *International Cryptology Conference*. Springer Berlin Heidelberg, 1998, pp. 408-423 (1998).
37. Jawurek, M., Kerschbaum, F., Orlandi, C.: Zero-knowledge using garbled circuits: how to prove non-algebraic statements efficiently. In: *CCS 2013*, pp. 955-966 (2013)
38. Katz J. Which languages have 4-round zero-knowledge proofs? In *Theory of Cryptography Conference (TCC2008)*. Springer, Berlin, Heidelberg, pp. 73-88 (2008)
39. Kalai, Y.T., Rothblum, G.N., Rothblum, R.D.: From obfuscation to the security of Fiat-shamir for proofs. In *CRYPTO 2017*, pp. 224-251 (2017)
40. Lepinski, M.: On the existence of 3-round zero-knowledge proofs. Ph.D. thesis, Massachusetts Institute of Technology (2002)
41. Li, Hongda, Dongxue Pan, and Peifang Ni. Augmented Black-Box Simulation and Zero Knowledge Argument for NP. <https://eprint.iacr.org/2017/1243>.
42. Pass R . Simulation in Quasi-Polynomial Time, and Its Application to Protocol Composition. In *Advances in Cryptology - EUROCRYPT 2003*, pp 160-176, 2003.
43. Pass, R., Rosen, A., New and improved constructions of non-malleable cryptographic protocols. In *STOC 2005*, ACM, 2005, pp. 533-542 (2005)
44. Pass R, Venkatasubramanian M. On constant-round concurrent zero-knowledge. In *Theory of Cryptography Conference*. Springer, Berlin, Heidelberg, 2008, pp. 553-570 (2008)
45. Rosen, A.. A note on constant-round zero-knowledge proofs for NP. In *TCC 2004*, pp. 191-202 (2004)
46. Sahai, A., Waters, B.. How to use indistinguishability obfuscation: Deniable encryption, and more. In *Proceedings of the 46th Annual ACM Symposium on Theory of Computing (STOC2014)*, pp. 475-484 (2014)
47. Yao A C. Protocols for secure computations. In *23rd annual symposium on foundations of computer science (SFCS 1982)*. IEEE, 1982: 160-164.