(Grouped) Ciphertext Validity Argument

1 Preliminaries

Basic notation. For two integers $n < m$, we write $[n, m]$ to denote the set $\{n, n+1, \ldots, m\}$. When $n = 1$, we simply write $[m]$ to denote the set $\{1, \ldots, m\}$. For any finite set S, we use $x \leftarrow_{\mathsf{R}} S$ to denote the process of sampling an element $x \in S$ uniformly at random. Unless specified otherwise, we use λ to denote the security parameter. We say that an algorithm is efficient if it runs in probabilistic polynomial time in the length of its input. We say that a function $f : \mathbb{N} \to \mathbb{N}$ is negligible if $f = o(1/n^c)$ for any positive integer $c \in \mathbb{N}$. Throughout the exposition, we use $\mathsf{poly}(\cdot)$ and $\mathsf{negl}(\cdot)$ to denote any polynomial and negligible functions respectively.

1.1 Discrete Log Relation Assumption

The discrete log relation assumption states that given a number of random group elements in $\mathbb{G}$, no efficient adversary can find a non-trivial relation on these elements.

Definition 1.1 (Discrete Log Relation)**.** Let $\mathbb{G} = \mathbb{G}(\lambda)$ be a group of prime order p. Then the *discrete log relation* assumption on $\mathbb{G}$ states that for any efficient adversary $\mathcal{A}$ and $n \geq 2$, there exists a negligible function $\mathsf{negl}(\lambda)$ such that

$$\Pr\left[\mathcal{A}(G_1, \ldots, G_n) \to a_1, \ldots, a_n \in \mathbb{Z}_p : \exists\ a_i \neq 0 \wedge \sum_{i \in [n]} a_i \cdot G = 0\right] = \mathsf{negl}(\lambda),$$

where $G_1, \ldots, G_n \leftarrow_{\mathsf{R}} \mathbb{G}$.

1.2 Rewinding Lemma

To prove security, we make use of the rewinding lemma. For the purpose of this document, we do not require the rewinding lemma in its full generality and therefore, we rely on the following simple variant from the work of Boneh et al. [1].

Lemma 1.2 (Rewinding Lemma)**.** *Let S, R, and T be finite, non-empty sets, and let X, Y, Y', Z, and Z' be mutually independent random variables such that*

- *X takes values in the set S,*
- *Y and Y' are each uniformly distributed over R,*
- *Z and Z' take values in the set T.*

Then for any function $f : S \times R \times T \to \{0, 1\}$, we have

$$\Pr\left[f(X, Y, Z) = 1 \wedge f(X, Y', Z') = 1 \wedge Y \neq Y'\right] \geq \varepsilon^2 - \varepsilon/N,$$

where $\varepsilon = \Pr[f(X, Y, Z) = 1]$ and $N = |R|$.

2 Zero-Knowledge Argument Definitions

In full generality, zero-knowledge argument systems can be defined with respect to any class of decidable languages. However, to simplify the presentation, we define argument systems with respect to CRS-dependent languages. Specifically, let $\mathcal{R} \subset \{0,1\}^* \times \{0,1\}^* \times \{0,1\}^*$ be an efficiently decidable ternary relation. Then a CRS-dependent language for a string $\rho \in \{0,1\}^*$ is defined as

$$\mathcal{L}_\rho = \{u \mid \exists\ w : (\rho, u, w) \in \mathcal{R}\}.$$

We generally refer to ρ as the common reference string, u as the instance of the langauge, and w as the witness for u.

For a class of CRS-dependent languages, an argument system consists of the following algorithms.

Definition 2.1 (Argument System)**.** A non-interactive argument system Π_{AS} for a CRS-dependent relation $\mathcal{R}$ consists of a tuple of efficient algorithms $(\mathsf{Setup}, \mathsf{Prove}, \mathsf{Verify})$ with the following syntax:

- $\mathsf{Setup}(1^\lambda) \to \rho$: On input the security parameter λ, the setup algorithm returns a common reference string ρ.
- $\mathcal{P}(\sigma, u, w)$: The prover $\mathcal{P}$ is an interactive algorithm that takes in as input a common reference string σ, instance u, and witness w. It interacts with the verifier $\mathcal{V}$ according to the specification of the protocol.
- $\mathcal{V}(\sigma, u)$: The verifier $\mathcal{V}$ is an interactive algorithm that takes in as input a common reference string ρ and an instance x. It interacts with the prover $\mathcal{P}$ in the protocol and in the end, it either accepts (returns 1) or rejects (returns 0) the instance x.

We use $\langle \mathcal{P}(\rho, u, w), \mathcal{V}(\rho, u) \rangle = 1$ to denote the event that the verifier $\mathcal{V}$ accepts the instance of the protocol. We use $\langle \mathcal{P}(\rho, u, w), \mathcal{V}(\rho, u) \rangle \to \mathsf{tr}$ to denote the communication transacript between the prover $\mathcal{P}$ and verifier $\mathcal{V}$ during a specific execution of the protocol.

An argument system must satisfy a correctness and two security properties. The correctness property of an argument system is generally referred to as *completeness*. It states that if the prover $\mathcal{P}$ takes in as input a valid instance-witness tuple $(\rho, u, w) \in \mathcal{R}$ and follows the protocol specification, then it must be able to convince the verifier to accept.

Definition 2.2 (Completeness)**.** Let Π_{AS} be a proof system for a relation $\mathcal{R}$. Then we say that Π_{AS} satisfies perfect completeness if for any $(u, w) \in \mathcal{R}$, we have

$$\Pr\left[\langle \mathcal{P}(\rho, u, w), \mathcal{V}(\rho, u) \rangle = 1\right] = 1,$$

where $\rho \leftarrow \mathsf{Setup}(1^\lambda)$.

The first security property that an argument system must satisfy is *soundness*, which can be defined in a number of ways. In this work, we work with *computational witness-extended emulation* as presented in Bulletproofs [2].

Definition 2.3 (Soundness [3, 4, 2])**.** Let Π_{AS} be a proof system for a relation $\mathcal{R}$. Then we say that Π_{AS} satisfies *witness-extended emulation* soundness if for all deterministic polynomial time $\mathcal{P}^*$,

there exists an efficient emulator $\mathcal{E}$ such that for all efficient adversaries $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, there exists a negligible function $\mathsf{negl}(\lambda)$ such that

$$\left| \begin{array}{c} \Pr\left[\mathcal{A}_2(\mathsf{tr}) = 1 \;\middle|\; \begin{array}{c} \rho \leftarrow \mathsf{Setup}(1^\lambda), (u, \mathsf{st}) \leftarrow \mathcal{A}_1(\rho), \\ \mathsf{tr} \leftarrow \langle \mathcal{P}^*(\rho, u, \mathsf{st}), \mathcal{V}(\rho, u) \rangle \end{array}\right] - \\ \Pr\left[\mathcal{A}_2(\mathsf{tr}) = 1 \wedge (\mathsf{tr} \text{ accepting } \Rightarrow (\rho, u, w) \in \mathcal{R}) \;\middle|\; \begin{array}{c} \rho \leftarrow \mathsf{Setup}(1^\lambda), \\ (u, \mathsf{st}) \leftarrow \mathcal{A}_1(\rho), \\ (\mathsf{tr}, w) \leftarrow \mathcal{E}^{\mathcal{O}}(\rho, u) \end{array}\right] \end{array} \right| = \mathsf{negl}(\lambda),$$

where the oracle is defined as $\mathcal{O} = \langle \mathcal{P}^*(\rho, u, \mathsf{st}), \mathcal{V}(\rho, u) \rangle$. The oracle $\mathcal{O}$ allows the emulator $\mathcal{E}$ to rewind the protocol to a specific point and resume the protocol after reprogramming the verifier with fresh randomness.

Traditionally, the soundness condition for an argument system of knowledge requires that there exists an extractor that can use its rewinding capability to extract a valid witness from any accepting transcript of the protocol that is produced by a dishonest prover $\mathcal{P}^*$. The witness-extended emulation strengthens this traditional definition by requiring that the extractor (emulator) not only successfully extracts a valid witness, but also produces (emulates) a valid transcript of the protocol for which the verifier accepts. The value st in the definition above can be viewed as the internal state of $\mathcal{P}^*$, which can also be its randomness.

The second security property that we require from an argument system is the zero-knowledge property. All argument systems that we rely on in the `ZK-Token` program are public coin protocols that we ultimately convert into a non-interactive protocol. Therefore, we rely on the standard zero-knowledge property against honest verifiers.

Definition 2.4 (Zero-Knowledge)**.** Let Π_{AS} be a proof system for a relation $\mathcal{R}$. Then we say that Π_{AS} satisfies *honest verifier* zero-knowledge if there exists an efficient simulator $\mathcal{S}$ such that for all efficient adversaries $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, we have

$$\begin{aligned} &\Pr\left[(\rho, u, w) \in \mathcal{R} \wedge \mathcal{A}_1(\mathsf{tr}) = 1 \;\middle|\; \begin{array}{c} \rho \leftarrow \mathsf{Setup}(1^\lambda), (u, w, \tau) \leftarrow \mathcal{A}_2(\rho), \\ \mathsf{tr} \leftarrow \langle \mathcal{P}(\rho, u, w), \mathcal{V}(\rho, u; \tau) \rangle \end{array}\right] \\ &= \Pr\left[(\rho, u, w) \in \mathcal{R} \wedge \mathcal{A}_1(\mathsf{tr}) = 1 \;\middle|\; \begin{array}{c} \rho \leftarrow \mathsf{Setup}(1^\lambda), \\ (u, w, \tau) \leftarrow \mathcal{A}_2(\rho), \\ \mathsf{tr} \leftarrow \mathcal{S}(u, \tau) \end{array}\right], \end{aligned}$$

where ρ is the public coin randomness used by the verifier.

3 (Grouped) Ciphertext Validity Argument

A ciphertext validity argument is defined with respect to a grouped ElGamal ciphertext that is defined with respect to a set of public key group elements $P_0, \ldots, P_{\ell-1}$ for a fixed positive integer $\ell \in [\ell]$. A ciphertext for a randomness r and message x with respect to a set of public key elements $P_0, \ldots, P_{\ell-1}$ is defined as follows:

$$\mathsf{ct} = \left(C = r \cdot H + x \cdot G,\ \{D_i = r \cdot P_i\}_{i \in [\ell]}\right)$$

At the start of a ciphertext validity argument protocol, the prover and verifier have access to a set of public keys $\{P_i\}_{i \in [\ell]}$ and ciphertext $\mathsf{ct} = (C, \{D_i\}_{i \in [\ell]})$. The prover's goal in the protocol is to

convince the verifier that it knows valid randomness r and message x that guarantee the validity of the ciphertext. Formally, the ciphertext-validity proof captures the following language:

$$\mathcal{L}^{\mathsf{ct\text{-}validity}}_{G,H,\ell} = \left\{ \begin{array}{c} u = (P_1, \ldots, P_\ell, C, D_1, \ldots, D_\ell) \in \mathbb{G}^{1+2\ell}, \\ w = (r, x) \in \mathbb{Z}_p^2 \end{array} \;\middle|\; \begin{array}{c} C = r \cdot H + x \cdot G \\ D_i = r \cdot P_i \quad \forall i \in [\ell] \end{array} \right\}.$$

The formal specification of the protocol is given as follows:

Prover(x, w)		**Verifier**(x)
$y_r \leftarrow_{\mathsf{R}} \mathbb{Z}_p$		
$y_x \leftarrow_{\mathsf{R}} \mathbb{Z}_p$		
$Y \leftarrow y_r \cdot H + y_x \cdot G$		
$X_i \leftarrow y_r \cdot P_i \quad \forall i \in [\ell]$		
	$\xrightarrow{Y, X_0, \ldots, X_{\ell-1}}$	
		$c \leftarrow_{\mathsf{R}} \mathbb{Z}_p$
	$\xleftarrow{c}$	
$z_r \leftarrow c \cdot r + y_r$		
$z_x \leftarrow c \cdot x + y_x$		
	$\xrightarrow{z_r, z_x}$	
		$z_r \cdot H + z_x \cdot G \stackrel{?}{=} c \cdot C + Y_0$
		$z_r \cdot P_i \stackrel{?}{=} c \cdot D_i + X_i \quad \forall i \in [\ell]$

At the start of the protocol, the prover samples random scalar elements y_r, y_r and commits to them by sending $Y = y_r \cdot H + y_x \cdot G$, $X_i = y_r \cdot P_i$ for $i \in [\ell]$. Upon receiving a challenge c, it provides the verifier with the masked randomness and message $z_r = c \cdot r + y_r$ and $z_x = c \cdot x + y_x$. Finally, the verifier tests the relations $z_r \cdot H + z_x \cdot G = c \cdot C + Y_0$ and $z_r \cdot P_i = c \cdot D_i + X_i$ for $i \in [\ell]$.

The ciphertext validity argument above satisfies all the correctness and security properties that are specified in Section 2. We formally state these properties in the following theorems.

Theorem 3.1 (Completeness)**.** *The ciphertext validity argument satisfies completeness 2.2.*

Theorem 3.2 (Soundness)**.** *Suppose that $\mathbb{G}$ is a prime order group for which the discrete log relation assumption (Definition 1.1) holds. Then the ciphertext valdity argument satisfies witness-extended emulation soundness 2.3.*

Theorem 3.3 (Zero-Knowledge)**.** *The ciphertext validity argument satisfies perfect honest verifier zero-knowledge 2.4.*

We provide the formal proofs for these theorems in Section 4.

4 Proofs

4.1 Proof of Theorem 3.1

To prove completeness, let us fix any valid instance $\mathcal{L}_{G,H,\ell}^{\mathsf{ct-validity}}$: $P_1, \ldots P_\ell, C, D_1, \ldots, D_{\ell-1} \in \mathbb{G}^{1+2\ell}$ and witness $r, x \in \mathbb{Z}_p^2$ such that

- $C = r \cdot H + x \cdot G$
- $D_i = r \cdot P_i \quad \forall i \in [\ell]$

Let y_r, y_x, z_r, z_x be any elements in $\mathbb{Z}_p$ and let

- $Y = y_r \cdot H + y_x \cdot G$,
- $X_i = y_r \cdot P_i$ for $i \in [\ell]$
- $z_r = c \cdot r + y_r$,
- $z_x = c \cdot x + y_x$,

in an execution of the protocol. Then we have

$$\begin{aligned} z_r \cdot H + z_x \cdot G &= (c \cdot r + y_r) \cdot H + (c \cdot x + y_x) \cdot G \\ &= c \cdot (r \cdot H + x \cdot G) + (y_r \cdot H + y_x \cdot G) \\ &= c \cdot C + Y \end{aligned}$$

$$\begin{aligned} z_r \cdot P_i &= (c \cdot r + y_r) \cdot P_i \\ &= c \cdot (r \cdot P_i) + y_r \cdot P_i \\ &= c \cdot D_i + y_r \cdot P_1 \\ &= c \cdot D_i + X_i \end{aligned}$$

As all of the algebraic relations that the verifier checks at the end of the protocol hold, the proof is always accepted. Completeness follows.

4.2 Proof of Theorem 3.2

To prove soundness, we construct an emulator $\mathcal{E}$ that has oracle access to any malicious prover $\mathcal{P}^*$ and extracts a valid witness by rewinding $\mathcal{P}^*$ and simulating four executions of the zero-balance protocol with an honest verifier $\mathcal{V}$.

Let $(P, C, D_1, \ldots, D_{\ell-1})$ be an instance of the language $\mathcal{L}_{G,H,\ell}^{\mathsf{ct-validity}}$. We construct an emulator $\mathcal{E}$ that uses $\mathcal{P}^*$ to extract a valid witness as follows:

- The emulator $\mathcal{E}$ first executes $\langle \mathcal{P}^*(\rho, u, \mathsf{st}), \mathcal{V}(\rho, u) \rangle$ to produce a transcript $\mathsf{tr} = (w, Y, X_1, \ldots, X_{\ell-1}, c, z_r, z_x)$.
- Then, it rewinds the protocol to the point where the verifier $\mathcal{V}$ samples a random $c \leftarrow_{\mathsf{R}} \mathbb{Z}_p$. It programs $\mathcal{V}$ with fresh randomness such that $\mathcal{V}$ generates a new $c' \leftarrow \mathbb{Z}_p$ independently of the previous execution of the protocol.
- The emulator completes the second execution of $\langle \mathcal{P}^*(\rho, u, \mathsf{st}), \mathcal{V}(\rho, u) \rangle$, producing a new transcript $\mathsf{tr} = (t, Y, X_1, \ldots, X_{\ell-1}, c', z'_r, z'_x)$.

- If $c - c' = 0$, then the emulator aborts and returns $\perp$. Otherwise, it computes
  - $r \leftarrow (z_r - z'_r)/(c - c')$
  - $x \leftarrow (z_x - z'_x)/(c - c')$

  and returns (r, x).

We first bound the probability that $\mathcal{E}$ does not abort at the end of the two executions of $\langle \mathcal{P}^*(\rho, u, \mathsf{st}), \mathcal{V}(\rho, u) \rangle$. Then, we show that if $\mathcal{E}$ does not abort, then its output (r, x) satisfies

- $C = r \cdot H + x \cdot G$,
- $D_i = r \cdot P_i$ for all $i \in [\ell]$.

Abort probability of the emulator. The emulator $\mathcal{E}$ aborts only when $c = c'$, which is dependent on the probability that $\mathcal{P}^*$ successfully convinces $\mathcal{V}$ at the end of the protocol. Let $\varepsilon_{\mathcal{P}^*}$ be the probability that $\mathcal{P}^*$ successfully convinces $\mathcal{V}$ in $\langle \mathcal{P}^*(\rho, u, \mathsf{st}), \mathcal{V}(\rho, u) \rangle$. We bound the probability that $c = c'$ with $\varepsilon_{\mathcal{P}^*}$ using the rewinding lemma 1.2. Specifically, let us define the following random variables:

- Let X be the elements $(w, Y, X_1, \ldots, X_{\ell-1})$ in the transcript of an execution of the protocol $\langle \mathcal{P}^*(\rho, u, \mathsf{st}), \mathcal{V}(\rho, u) \rangle$.
- Let Y and Y' be the values c and c' respectively in the two executions of $\langle \mathcal{P}^*(\rho, u, \mathsf{st}), \mathcal{V}(\rho, u) \rangle$.
- Let Z and Z' be the values (z_r, z_x) and (z'_r, z'_x) respectively in the two executions of $\langle \mathcal{P}^*(\rho, u, \mathsf{st}), \mathcal{V}(\rho, u) \rangle$.
- Let $f(\mathsf{tr}) \to \{0, 1\}$ be the protocol verification function that returns 1 if tr is an accepting transcript and 0 otherwise.

Then, the rewiding lemma states that

$$\Pr\left[f(X, Y, Z) = 1 \wedge f(X, Y', Z') = 1 \wedge Y \neq Y'\right] \geq \varepsilon^2 - \varepsilon/p.$$

By assumption, we have $1/p = \mathsf{negl}(\lambda)$. Therefore, if $\varepsilon_{\mathcal{P}^*}$ is non-negligible, then the probability that $\mathcal{E}$ aborts at the end of the two executions of $\langle \mathcal{P}^*(\rho, u, \mathsf{st}), \mathcal{V}(\rho, u) \rangle$ is non-negligible.

Output validity of emulator. Now assume that the two executions of $\langle \mathcal{P}(\rho, u, w), \mathcal{V}(\rho, u) \rangle$ returns two accepting transcripts $\mathsf{tr} = (Y, X_1, \ldots, X_{\ell-1}c, z_r, z_x)$, $\mathsf{tr}' = (Y, X_1, \ldots, X_{\ell-1}c', z'_r, z'_x)$, and that $\mathcal{E}$ does not abort and returns

- $r \leftarrow (z_r - z'_r)/(c - c')$
- $x \leftarrow (z_x - z'_x)/(c - c')$

Since tr and tr' are accepting transcripts, we have

$$z_r \cdot H + z_x \cdot G = c \cdot C + Y,$$
$$z'_r \cdot H + z'_x \cdot G = c' \cdot C + Y,$$

This means that $(z_r - z'_r) \cdot H + (z_x - z'_x) \cdot G = (c - c') \cdot C$ and hence, $r \cdot H + x \cdot G = C$. Similarly, we have

$$z_r \cdot P_i = c \cdot D + X_i,$$
$$z'_r \cdot P_i = c' \cdot D + X_i,$$

for $i \in [\ell]$ This means that $(z_r - z'_r) \cdot P_i = (c - c') \cdot D_i$, which means that $r \cdot P_i = D_i$ for $i \in [\ell]$. Soundness follows.

4.3 Proof of Theorem 3.3

Fix any elements $P, C, D_1, \ldots, D_{\ell-1} \in \mathbb{G}$ and $r, x \in \mathbb{Z}_p$ such that the ciphertext validity relation holds. Let $\mathsf{tr}^* = (w, Y^*, X_1^*, \ldots, X_{\ell-1}^*, c^*, z_r^*, z_x^*)$ be any accepting transcript. By the specification of the protocol, the probability that an honest execution of the protocol by the prover and the verifier results in the transcript tr^* is given by

$$\Pr\left[\left\langle \mathcal{P}(\rho, u, w), \mathcal{V}(\rho, u)\right\rangle \to \mathsf{tr} \wedge \mathsf{tr} = \mathsf{tr}^*\right] = 1/p^{\ell+1}.$$

To prove zero-knowledge, we define a simulator $\mathcal{S}$ that produces such distribution without knowledge of a valid witness r and x.

$\mathcal{S}(P, C, D_0, \ldots, D_{\ell-1})$:

1. Sample $c, z_r, z_x, \leftarrow_{\mathsf{R}} \mathbb{Z}_p$
2. Set $Y = z_r \cdot H + z_x \cdot G - c \cdot C$
3. Set $X_i = z_r \cdot P - c \cdot D_i$
4. Return $\mathsf{tr} = (w, Y, X_0, \ldots, X_{\ell-1}, c, z_r, z_x)$

The simulator $\mathcal{S}$ returns a transcript that is uniformly random given that

- $z_r \cdot H + z_x \cdot G = c \cdot C + Y$,
- $z_i \cdot P_i = c \cdot D_i + Y_i$ for $i \in [\ell]$.

As the variables Y, $X_0, \ldots, X_{\ell-1}$ are completely determined by t, c, z_r, z_x, we have

$$\Pr\left[\mathcal{S}(P, C, D_0, \ldots, D_{\ell-1}) \to \mathsf{tr} \wedge \mathsf{tr} = \mathsf{tr}^*\right] = 1/p^{\ell+1},$$

for any fixed transcript tr^*. Zero-knowledge now follows.

References

[1] Boneh, D., Drijvers, M., and Neven, G. Compact multi-signatures for smaller blockchains. In *International Conference on the Theory and Application of Cryptology and Information Security* (2018), Springer, pp. 435–464.

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

[3] Groth, J., and Ishai, Y. Sub-linear zero-knowledge argument for correctness of a shuffle. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques* (2008), Springer, pp. 379–396.

[4] Lindell, Y. Parallel coin-tossing and constant-round secure two-party computation. *Journal of Cryptology 16*, 3 (2003).