Explainable Arguments

Lucjan Hanzlik[1(✉)] and Kamil Kluczniak[1,2]

[1] CISPA Helmholtz Center for Information Security, Saarbrücken, Germany
{lucjan.hanzlik,kamil.kluczniak}@cispa.saarland
[2] Stanford University, Stanford, USA
kamil.kluczniak@stanford.edu

Abstract. We introduce an intriguing new type of argument systems with the additional property of being explainable. Intuitively by explainable, we mean that given any argument under a statement, and any witness, we can produce the random coins for which the Prove algorithm outputs the same bits of the argument.

This work aims at introducing the foundations for the interactive as well as the non-interactive setting. We show how to build explainable arguments from witness encryption and indistinguishability obfuscation. Finally, we show applications of explainable arguments. Notably we construct deniable chosen-ciphertext secure encryption. Previous deniable encryption scheme achieved only chosen plaintext security.

1 Introduction

Deniability, first introduced by Dolev, Dwork, and Naor [30], is a notion that received a considerable amount of attention because of its application to authentication protocols. This property allows the user to argue against a third party that it did not take part in a protocol execution. The usual argument made by the user to the third party is that the server could simulate a valid communication transcript without actually interacting with the user.

A variant of deniability was considered in the case of encryption schemes [15,16,63], where a public Expl algorithm allows anyone to open any ciphertext to any message without the secret key. Since we can publicly open ciphertexts, the random coins cannot serve as proof that a particular message is encrypted.

A similar concept was recently introduced to ring signatures [58] and called unclaimability. The property states that no one can claim to be the signer of a particular ring signature σ. The premise is similar. There exists an Expl algorithm that allows any of the ring members to generate random coins that can be used to receive the same σ.

Deniability and unclaimability are related notions. In the former, we consider the server malicious because it tries to gain an undeniable proof of an interaction. In the latter, the malicious party is a different user that tries to make it impossible for honest users to explain an interaction/signature. Interestingly, the deniability and unclaimability definitions studied in the literature only consider scenarios where the party producing a transcript/signature/ciphertext is honest, but may eventually become corrupt in the future.

I. Eyal and J. Garay (Eds.): FC 2022, LNCS 13411, pp. 59–79, 2022.
https://doi.org/10.1007/978-3-031-18283-9_4

1.1 Contribution

We introduce a new property for argument systems called explainability. Explainability informally resembles deniability and unclaimability. We consider interactive and non-interactive variants of such systems. We show that achieving strong explainability is hard and requires very strong primitives like witness encryption (WE) and indistinguishability obfuscation (iO). Our contribution can be summarized as follows.

New Definitions. We introduce a new property for argument systems that we call explainability, i.e., the ability for anyone with a valid witness wit to compute the random coins coins that "explain" a given argument arg. By "explain," we mean that the witness and coins result in the same argument string $\mathsf{arg} = \mathsf{Prove}(\mathsf{stmt}, \mathsf{wit}; \mathsf{coins})$ or the same transcript of an interaction, given the same instance of the verifier. Thus if one can explain an argument for all witnesses and all coins, then such argument/transcript cannot serve as proof that a particular witness was used. We accounted for certain subtle differences between interactive and non-interactive arguments. In both cases, we consider *malicious prover* explainability, where a prover tries to create a proof that other provers cannot explain with a different but valid witness. In this case, we require the protocol to be unique, in the sense that it is infeasible for a malicious prover to produce two different arguments (or transcripts) that the verifier accepts given the same statement and random coins. For the interactive case, we also consider a *malicious verifier* (similar to deniability) that can abort the protocol execution or send corrupt messages to make it impossible for provers with a different witness to explain the current interaction. Since, in the non-interactive case, there is no interaction with a verifier, we consider a scenario where the common reference string (if used) is maliciously generated. We refer to this case as *malicious setup explainability*. Additionally, we call a (non-)interactive argument system fully explainable, when it is explainable even if both the setup/verifier and the prover are malicious.

Implications. To study the power of explainable arguments we prove several interesting implications of explainable arguments.

- We show that when an argument system is malicious verifier explainable, then it is also witness indistinguishable.
- We show that non-interactive malicious prover explainable arguments and one-way functions imply witness encryption (WE). This result serves us as evidence that constructing such arguments is difficult and requires strong cryptographic primitives.

Constructions of Interactive Explainable Arguments. We introduce new properties for witness encryption that we call *robustness* and *plaintext awareness*. Informally, robustness ensures that decryption is independent of which witness is

used. In other words, there do not exist two valid witnesses for which a ciphertext decrypts to a different message (or $\perp$). Plaintext awareness ensures that an encrypter must know the plaintext it encrypted. We then show how to leverage robust witness encryption to construct interactive explainable arguments. The resulting protocol is round-optimal, predictable, and can be instantiated to yield an optimally laconic argument. Given the witness encryption is plaintext aware, we can show that the protocol is zero-knowledge. Finally, assuming the witness encryption is extractably secure, we can show that our protocol is a proof of knowledge.

Constructions of Non-interactive Explainable Arguments. We show how to construct malicious setup and malicious prover explainable arguments from indistinguishability obfuscation. While malicious prover explainable arguments can trivially be build using techniques from Sahai and Waters [63], the case of malicious setup explainable arguments is more involved and requires us to use dual-mode witness indistinguishable proofs. Furthermore, we show how to build fully explainable arguments, additionally assuming NIZK.

Why Study Explainable Arguments? Argument systems are fundamental primitives in cryptography. While some privacy properties like zero-knowledge already give a strong form of deniability, our notion of explainability is much stronger as it considers the extreme case where the provers' coins are leaked or are chosen maliciously. For example, using our explainable arguments, we can show explainable interactive anonymous authentication schemes, where anonymity is defined similarly as in ring-signature schemes (see full paper [45]). Notably, we can construct CCA-1 secure encryption with deniability as defined by Sahai and Waters [63], from CPA secure deniable encryption and our explainable arguments assuming random oracles. Our deniable encryption is a variant of the Naor-Yung transform [56], but only rely on witness indistinguishability instead of zero-knowledge which allows us to instantiate this transformation using our explainable arguments.

Malicious Verifier/Setup Explainability. We consider adversaries that are substantially more powerful than what is usually studied in the literature, e.g., in deniable authentication schemes or ring-signatures. In particular, in our case, the user can deny an argument even when the adversary asks to reveal the user's random coins used to produce the argument. Immediate real-world examples of such powerful adversaries are rogue nation-state actors that might have the right to confiscate a user's hardware and apply effectual forensics techniques to obtain the random seeds as evidence material against the user. We believe that the threat posed by such potent adversaries may prevent the use of e.g., ring-signatures by whistleblowers, as the anonymity notions provided might be insufficient.

Malicious Prover Explainability. The main application we envision for malicious prover explainability is internet voting. An essential part of a sound and fair

voting scheme is to prevent the selling of votes by malicious voters. We note that the "selling votes" issue isn't limited to actual bribery but, perhaps more critically, addresses the issue of forcing eligible voters to vote on a particular candidate. In this case, an authoritarian forces others to deliver evidence that they voted on a particular option or participate in a specific digital event. An authoritarian here may be an abusive family member, corrupt supervisor, or employer. Our strong unclaimability notion is essential to handle such drastic cases, mainly because users might be coerced or bribed to use specific coins in the protocol.

1.2 Related Work

Explainability of the verifier was used by Bitansky and Choudhuri [8] as a step in proving the existence of deterministic-prover zero-knowledge proofs. In their definition they used the fact that the choices of a verifier can be "explained" by outputting random coins that will lead to the same behaviour. This later can be used to transform the system to be secure even against a malicious verifier. In contrary, we consider the explainability of the prover. While arguments with our type of explainability have not been studied before, there exists some related concepts. Here we give an overview of the related literature.

Deniable Authentication. Dolev, Dwork, and Naor [30] first introduced the concept of deniability. The first formal definition is due to Dwork, Naor, and Sahai [32]. Deniability was studied in numerous works [25,48,55] in the context of authentication protocols. The concept was later generalized to authenticated key exchange and was first formally defined by Di Raimondo, and Genaro [26]. Since then deniable key exchange protocols got much attention from the community [11,24,27,28,46,49,51,65–69]. In such protocols, deniability is informally defined as a party's ability to simulate the transcript of interaction without actually communicating with another party. Since each party can generate a transcript itself, the transcript cannot be used as proof to a third party that the interaction took place. At a high level, deniability is very similar to zero-knowledge, but it is important to mention that Pass [59] showed some subtle differences between both notions.

Deniable Encryption. Deniable encryption was first introduced by Canetti, Dwork, Naor, and Ostrovsky [15]. Here we deal with a "post" compromise situation, where an honest encrypter may be forced to "open" a ciphertext. In other words, given a ciphertext, it should be possible to show a message and randomness that result in the given ciphertext. Deniable encryption was intensively studied [1,7,20–22,41,57,63]. Very recently, Canetti, Park, and Poburinnaya [16] generalize deniable encryption to the case where multiple parties are compromised and show constructions also assuming indistinguishability obfuscation.

Ring Signatures. Early forms of deniability were the main motivation for the work of Rivest, Shamir, and Tauman [61], which introduces the concept of ring

signatures. This early concept took into account a relaxed form of deniability where only the secret key of a user may leak. Very recently [58] extended ring signatures with additional deniability properties. For example, they show a signer deniable ring signature where any signer may generate random coins that, together with its secret key, will result in the given signature. However, they require to assume the prover is honest at the moment of signature generation. In our argument setting, we do not make such assumptions.

We are the first to study arguments with unclaimability and deniability properties that allow denying executing a protocol even when the prover is forced to reveal all its random coins or where the prover chooses its coins maliciously. Previous works mostly address a post-compromise setting, whereas some of our explainability notions take into account malicious prover. We believe that our primitives may find applications in protocols as a means of providing consistency checks or anonymous authentication of the votes. For example, the protocols from [17,62] rely on a trusted party to verify a voter's signature. That party knows the user's vote. Using our explainable arguments, we can build (see full paper) a simple anonymous authentication protocol without degrading receipt freeness of the voting scheme, and in effect, remove the trust assumption in terms of privacy.

Receipt Freeness and Coertion Resistance in Voting Schemes. Some of our definitions and potential application are tightly connected to voting schemes. In particular, our definition of malicious prover explainability poses the same requirements, at a high level, for an argument system as receipt freeness or coercion resistance in voting schemes [6,47,54,64]. Since we focus on a single primitive, our definitions are much simpler in comparison to complex voting systems. For example, the definition from [17] involves numerous oracles, and defines a set of parties, and assumes trusted parties. Our definition for malicious prover explainability is simple and says that it is infeasible to produce two different arguments under the same statement that verify incorrectly.

Outline of the Paper. In Sect. 3 we give definitions of explainable argument systems. In Sect. 4 we construct non-interactive explainable arguments. In Sect. 5 we introduce robust witness encryption, and apply it to build interactive explainable arguments. Finally, in Sect. 6, we show how to apply explainable arguments to construct deniable CCA-secure encryption. In the full paper [45], we recall all definitions for the primitives in the preliminaries section, show an explainable anonymous authentication protocol, and all security proofs.

2 Preliminaries

Notation. We denote execution of an algorithm Alg on input x as $a \leftarrow \mathsf{Alg}(x)$ were the output is assigned to a. Unless said otherwise, we will assume that algorithms are probabilistic and choose some random coins internally. In some cases, however, we will write $\mathsf{Alg}(.; r)$ to denote that Alg proceeds deterministically on

input a seed $r \in \{0,1\}^s$ for some integer s. We denote an execution of a protocol between parties V and P, by $\langle \mathsf{Prove}(.) \rightleftharpoons \mathsf{Verify}(.) \rightarrow x \rangle = \mathsf{trans}$, where x is the output of Verify after completion of the protocol, and trans is the transcript of the protocol. A transcript trans contains all messages send between Prove and Verify and the input of Verify. We write $\mathsf{View}(\mathsf{Prove}(.) \rightleftharpoons \mathsf{Verify}(.))$ to denote the view of Verify. The view contains the transcript, all input to Verify including its random coins and its internal state. W say that a function $\mathsf{negl} : \mathbb{N} \mapsto \mathbb{R}^+$ is negligible if for every constant $c > 0$ there exists a integer $N_c \in \mathbb{N}$ such that for all $\lambda > N_c$ we have $\mathsf{negl}(\lambda) < \lambda^{-c}$.

Standard Definitions. We use a number of standard cryptographic tools throughout the paper, including: pseudorandom generators and Goldreich-Levin hardcore bits [39], existential unforgeable and unique signature schemes [37,42], zero-knowledge (ZK) and witness-indistinguishable (WI) argument systems, non-interactive ZK arguments from non-falsifiable assumptions [35], dual-mode witness-indistinguishable proofs [43], CCA1 secure and publicly deniable encryption [63], witness encryption [36] and extractable witness encryption [40], indistinguishability obfuscation [3], and punctured pseudorandom functions [13,14,50].

3 Explainable Arguments

In this section, we introduce the security notions for explainable arguments.

3.1 Interactive Explainable Arguments

In an interactive argument system, the prover uses a witness wit for statement stmt to convince the verifier that the statement is true. The communication between the prover and the verifier creates a transcript trans that contains all the exchanged messages. An interactive explainable argument system allows a prover with a different witness wit^* to generate random coins coins for which $\mathsf{Prove}(\mathsf{stmt}, \mathsf{wit}^*; \mathsf{coins})$ interacting with the same instance of the verifier (i.e., the verifier uses the same random coins) creates the same transcript trans. In other words, this means that any prover with a valid witness can provide random coins that would explain the interaction in trans. More formally.

Definition 1 (Interactive Explainable Arguments). *An interactive argument system* $\Pi_{\mathcal{R}} = (\mathsf{Prove}, \mathsf{Verify})$ *for language* $L_{\mathcal{R}}$ *is an interactive explainable argument system if there exists an additional* Expl *algorithm:*

- $\mathsf{Expl}(\mathsf{stmt}, \mathsf{wit}, \mathsf{trans})$*: takes as input a statement* stmt*, any valid witness* wit *(i.e.* $\mathcal{R}(\mathsf{stmt}, \mathsf{wit}) = 1$*) and transcript* trans*, and outputs* $\mathsf{coins} \in \mathbf{Coin}_{\mathsf{Prove}}$ *(i.e. coins that are in the space of the randomness used in* Prove*),*

which satisfies the correctness definition below.

Definition 2 (Correctness). *For all security parameter λ, for all statements* $\mathsf{stmt} \in L_{\mathcal{R}}$*, for all* $\mathsf{wit}, \mathsf{wit}^*$ *such that* $\mathcal{R}(\mathsf{stmt}, \mathsf{wit}) = \mathcal{R}(\mathsf{stmt}, \mathsf{wit}^*) = 1$*, we have*

$$\langle \mathsf{Verify}(\mathsf{stmt}) \rightleftharpoons \mathsf{Prove}(\mathsf{stmt}, \mathsf{wit}) \rangle =$$
$$\langle \mathsf{Verify}'(\mathsf{stmt}; \mathsf{trans}) \rightleftharpoons \mathsf{Prove}(\mathsf{stmt}, \mathsf{wit}^*; \mathsf{coins}_E) \rangle = \mathsf{trans},$$

where $\mathsf{coins}_E \leftarrow \mathsf{Expl}(\mathsf{stmt}, \mathsf{wit}^*, \mathsf{trans})$ *and* $\mathsf{coins}_E \in \mathbf{Coin}_{\mathsf{Prove}}$ *and* Verify' *sends its messages as in* trans *as long as* Prove *answers as is* trans*. If the output of* Prove *do not match* trans*, then* Verify' *aborts and outputs* $\perp$*.*

Remark 1. Note that a naive way to implement the Expl algorithm would be to set coins_E and make the Prove algorithm to "replay" the messages. However, this is obviously a scheme that would not be desirable, since an adversary could easily distinguish such coins from honest ones. Therefore we require that $\mathsf{coins}_E \in \mathbf{Coin}_{\mathsf{Prove}}$ to ensure that coins_E can be given as input to an honest Prove algorithm.

The above definition constitutes a correctness definition for explainable arguments and assumes that all parties are honest. Informally, we require that given a witness and a transcript of an interaction between a verifier and a prover (with a possibly different witness), Expl generates coins such that a honest prover returns the same messages given that the verifier send its messages as in trans.

Below we describe explainability of a malicious verifier. Roughly speaking, this property says that a transcript produced during an execution with a malicious verifier, and a honest prover P, should be explainable. The goal of a verifier, is to send such messages to the prover P, that P sends such responses that no other prover (with a different witness) would send. If the adversary succeeds then the transcript (possibly with P's random coins) can be used as a proof to a third party, that P indeed took part in the communication. Remind that P may be forced to reveal its random coins after completing the protocol.

Definition 3 (Malicious Verifier Explainability). *For a security parameter λ, we define the advantage* $\mathsf{Adv}^{\mathsf{MVExpl}}_{\mathcal{A}}(\lambda)$ *of an adversary* $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3)$ *as*

$$1 - \Pr[\langle \mathcal{A}_3(\mathsf{stmt}; \mathsf{coins}_{\mathcal{A}}) \rightleftharpoons \mathsf{Prove}(\mathsf{stmt}, \mathsf{wit}^*; \mathsf{coins}_P) \rangle = \mathsf{trans}], \text{ where}$$

$$\begin{aligned} (\mathsf{stmt}, \mathsf{wit}, \mathsf{wit}^*, \mathsf{st}) &\leftarrow \mathcal{A}_1(\lambda), \\ \mathsf{trans} = \langle \mathsf{coins}_{\mathcal{A}} \leftarrow \mathcal{A}_2(\mathsf{stmt}; \mathsf{st}) &\rightleftharpoons \mathsf{Prove}(\mathsf{stmt}, \mathsf{wit}) \rangle, \\ \mathsf{coins}_P &\leftarrow \mathsf{Expl}(\mathsf{stmt}, \mathsf{wit}^*, \mathsf{trans}), \\ \mathsf{wit} \neq \mathsf{wit}^*, \quad \mathcal{R}(\mathsf{stmt}, \mathsf{wit}) &= \mathcal{R}(\mathsf{stmt}, \mathsf{wit}^*) = 1, \end{aligned}$$

where the probability is taken over the random coins of Prove*. Furthermore, $\mathcal{A}_3$ sends the same messages to* Prove *as in* trans *as long as the responses from the prover are as in* trans*.*

We say that an interactive argument system is malicious verifier explainable *if for all adversaries $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3)$ such that $\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3$ are PPT algorithms*

there exists a negligible function $\mathsf{negl}(.)$ *such that* $\mathsf{Adv}^{\mathsf{MVExpl}}_{\mathcal{A}}(\lambda) \leq \mathsf{negl}(\lambda)$. *We say that the argument system is* malicious verifier statistically explainable *if the above holds for an unbounded adversary* $\mathcal{A}$.

Let us now consider a scenario where proving ownership of an argument is beneficial to the prover, but at the same time, the system requires the proof to be explainable. A malicious prover tries to prove the statement in a way that makes it impossible for others to "claim" the generated proof. For this property, it is easy to imagine a malicious prover that sends such messages to the verifier, that the verifier accepts, and no other honest prover would ever send such messages. In practice, we may imagine that an adversary runs a different implementation of the prover, for which the distribution of the sent messages deviate from the distribution of the original implementation. Later to "claim" the transcript that adversary may prove that the transcript is indeed the result of the different algorithm, not the honest one. Note that such a "claim" is sound if an honest prover would never produce such messages. To prevent such attacks, we require that there is only one (computationally feasible to find) valid way a prover can respond to the messages from an honest verifier.

Definition 4 (Uniqueness/Malicious Prover Explainability). *We define the advantage* $\mathsf{Adv}^{\mathsf{MPExpl}}_{\mathcal{A}}(\lambda)$ *of an adversary* $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3)$ *as*

$$1 - \Pr\left[\begin{array}{c} \langle 1 = \mathsf{Verify}(\mathsf{stmt}; \mathsf{coins}_V) \rightleftharpoons \mathcal{A}_2(\mathsf{st}_1) \rightarrow \mathsf{st}_2 \rangle \\ \neq \langle 1 = \mathsf{Verify}(\mathsf{stmt}; \mathsf{coins}_V) \rightleftharpoons \mathcal{A}_3(\mathsf{st}_2) \rangle \end{array}\right],$$

where $\mathsf{st}_1, \mathsf{stmt} \leftarrow \mathcal{A}_1(\lambda)$ *and the probability is taken over the coins* coins_V.

We say that an interactive argument system is malicious prover explainable *if for all PPT adversaries* $\mathcal{A}$ *there exists a negligible function* $\mathsf{negl}(.)$ *such that* $\mathsf{Adv}^{\mathsf{MPExpl}}_{\mathcal{A}}(\lambda) \leq \mathsf{negl}(\lambda)$. *We say that the system is* malicious prover statistically explainable *if the above holds for an unbounded* $\mathcal{A}$.

Theorem 1. *If* $(\mathsf{Prove}, \mathsf{Verify}, \mathsf{Expl})$ *is a malicious verifier (statistical) explainable argument system then it is also (statistical) witness indistinguishable.*

Definition 5. *We say that an interactive argument system is fully explainable if it is malicious prover explainable and malicious verifier explainable.*

3.2 Non-interactive Explainable Arguments

Here we present definitions for non-interactive explainable arguments. Similar to the interactive case, we begin by defining what it means that a system is explainable.

Definition 6 (Non-Interactive Explainable Arguments). *A non-interactive argument system* $\Pi_{\mathcal{R}} = (\mathsf{Setup}, \mathsf{Prove}, \mathsf{Verify})$ *for language* $L_{\mathcal{R}}$ *is a non-interactive explainable argument system if there exists an additional* Expl *algorithm:*

– $\mathsf{Expl}(\mathsf{crs}, \mathsf{stmt}, \mathsf{wit}, \arg)$*: takes as input a statement* stmt*, any valid witness* wit *and an argument* $\arg$*, and outputs random coins* coins

which satisfies the correctness definition below.

Definition 7 (Correctness). *For all security parameter* λ*, for all statements* $\mathsf{stmt} \in L_{\mathcal{R}}$*, for all* $\mathsf{wit}, \mathsf{wit}^*$ *such that* $\mathcal{R}(\mathsf{stmt}, \mathsf{wit}) = \mathcal{R}(\mathsf{stmt}, \mathsf{wit}^*) = 1$*, for all random coins* $\mathsf{coins}_P \in \mathbf{Coin}_{\mathsf{Prove}}$*, we have*

$$\mathsf{Prove}(\mathsf{crs}, \mathsf{stmt}, \mathsf{wit}; \mathsf{coins}_P) = \mathsf{Prove}(\mathsf{crs}, \mathsf{stmt}, \mathsf{wit}^*; \mathsf{coins}_E)$$

where $\mathsf{coins}_E \leftarrow \mathsf{Expl}(\mathsf{crs}, \mathsf{stmt}, \mathsf{wit}^*, \arg)$*,* $\mathsf{coins}_E \in \mathbf{Coin}_{\mathsf{Prove}}$ *and* $\mathsf{crs} \leftarrow \mathsf{Setup}(\lambda)$*.*

Now we define malicious setup explainability. Note that a malicious verifier cannot influence the explainability of an argument because there is no interaction with the prover. Hence, the malicious verifier from the interactive setting is replaced with an untrusted setup. An adversary might generate parameters that result in the Expl algorithm to output coins yielding a different argument or even failing on certain witnesses. In some sense, we can think of the adversary as wanting to subvert the common reference string against deniability of certain "targeted" witnesses.

Definition 8 (Malicious Setup Explainability). *We define the advantage* $\mathsf{Adv}^{\mathsf{MSExpl}}_{\mathcal{A}}(\lambda)$ *of an adversary* $\mathcal{A}$ *by the following probability*

$$1 - \Pr\left[\arg^* = \arg : \begin{array}{c} (\mathsf{stmt}, \mathsf{wit}, \mathsf{wit}^*, \mathsf{crs}) \leftarrow \mathcal{A}(\lambda) \\ \mathsf{wit} \neq \mathsf{wit}^* \\ \mathcal{R}(\mathsf{stmt}, \mathsf{wit}) = \mathcal{R}(\mathsf{stmt}, \mathsf{wit}^*) = 1 \\ \arg \leftarrow \mathsf{Prove}(\mathsf{crs}, \mathsf{stmt}, \mathsf{wit}); \\ \mathsf{coins}_P \leftarrow \mathsf{Expl}(\mathsf{crs}, \mathsf{stmt}, \mathsf{wit}^*, \arg); \\ \arg^* \leftarrow \mathsf{Prove}(\mathsf{crs}, \mathsf{stmt}, \mathsf{wit}^*; \mathsf{coins}_P) \end{array}\right],$$

where the probability is taken over the random coins of the prover Prove*. We say that a non-interactive argument is* malicious setup explainable *if for all PPT adversaries* $\mathcal{A}$ *there exists a negligible function* $\mathsf{negl}(.)$ *such that* $\mathsf{Adv}^{\mathsf{MSExpl}}_{\mathcal{A}}(\lambda) \leq \mathsf{negl}(\lambda)$*. We say the that a non-interactive argument is* malicious setup statistically explainable *if the above holds for an unbounded adversary* $\mathcal{A}$*. Moreover, we say that a non-interactive argument is perfectly malicious setup explainable if* $\mathsf{Adv}^{\mathsf{MSExpl}}_{\mathcal{A}}(\lambda) = 0$*.*

Theorem 2. *If there exists a malicious setup explainable non-interactive argument, then there exists a two-move witness-indistinguishable argument, where the verifier's message is reusable. In other words, given a malicious setup explainable non-interactive argument, we can build a private-coin ZAP.*

Malicious prover explainability is defined similarly as in the case of interactive arguments. For the non-interactive setting, it is simpler to formalize the definition, as we simply require the adversary to return two arguments that verify correctly, but their canonical representation is different.

Definition 9 (Uniqueness/Malicious Prover Explainability). *We define the advantage of an adversary $\mathcal{A}$ against malicious prover explainability of* ExArg *as* $\mathsf{Adv}^{\mathsf{MPExpl}}_{\mathcal{A}}(\lambda) = \Pr[\arg_1 \neq \arg_2]$ *where* $\mathsf{crs} \leftarrow \mathsf{Setup}(\lambda)$ *and* $(\mathsf{stmt}, \arg_1, \arg_2) \leftarrow \mathcal{A}(\lambda)$ *are such that* $\mathsf{Verify}(\mathsf{crs}, \mathsf{stmt}, \arg_1) = \mathsf{Verify}(\mathsf{crs}, \mathsf{stmt}, \arg_2)$*, and the probability is over the random coins of* Setup*. We say that a non-interactive argument is* malicious prover explainable *if for all PPT adversaries $\mathcal{A}$ there exists a negligible function* $\mathsf{negl}(.)$ *such that* $\mathsf{Adv}^{\mathsf{MPExpl}}_{\mathcal{A}}(\lambda) \leq \mathsf{negl}(\lambda)$*. We say that a non-interactive argument is* malicious prover statistically explainable *if the above holds for an unbounded adversary $\mathcal{A}$. Moreover, we say that an argument system is a perfectly malicious prover explainable if* $\mathsf{Adv}^{\mathsf{MPExpl}}_{\mathcal{A}}(\lambda) = 0$.

For full explainability, we combine both malicious prover and malicious verifier explainability.

Definition 10 (Full Explainability). *We define the advantage of an adversary $\mathcal{A}$ against full explainability of* ExArg *by the following probability*

$$\mathsf{Adv}^{\mathsf{FExpl}}_{\mathcal{A}}(\lambda) = \Pr[\arg_1 \neq \arg_2]$$

where $(\mathsf{stmt}, \mathsf{crs}, \arg_1, \arg_2) \leftarrow \mathcal{A}(\lambda)$ *is such that* $\mathsf{Verify}(\mathsf{crs}, \mathsf{stmt}, \arg_1) = \mathsf{Verify}(\mathsf{crs}, \mathsf{stmt}, \arg_2)$*. We say that a non-interactive argument is* full explainable *if for all PPT adversaries $\mathcal{A}$, there exists a negligible function* $\mathsf{negl}(.)$ *such that* $\mathsf{Adv}^{\mathsf{FExpl}}_{\mathcal{A}}(\lambda) \leq \mathsf{negl}(\lambda)$*. We say that the non-interactive argument is* full statistically explainable *if the above holds for an unbound adversary $\mathcal{A}$. Moreover, we say that an argument system is perfectly full explainable if* $\mathsf{Adv}^{\mathsf{FExpl}}_{\mathcal{A}}(\lambda) = 0$.

Theorem 3. *If* ExArg *is a fully explainable argument, then* ExArg *is a malicious setup and malicious prover explainable argument.*

Theorem 4. *Given that one-way functions and malicious prover selectively sound non-interactive (resp. two-move) arguments for* NP *exist, then there exists a witness encryption scheme for* NP*.*

4 Non-interactive Explainable Arguments

In this section, we show that it is possible to construct malicious setup explainable non-interactive argument systems from falsifiable assumptions. We also show a fully explainable argument assuming non-interactive zero-knowledge. As both schemes are nearly identical and differ only in several lines, we will denote the lines or specific algorithms with $\circ$ for the malicious setup explainable argument, and with $\dagger$, we denote the code specific for the fully explainable argument.

Scheme 1 (Non-interactive Explainable Argument). Let $\nabla = \circ$ for the malicious setup explainable argument, and $\nabla = \dagger$ for the fully explainable argument. Let DMWI be a dual-mode proof, NIWI be a non-interactive witness indistinguishable proof, Com be an equivocal commitment scheme, Sig be a unique signature scheme, and PRF be a punctured pseudorandom function. We construct the non-interactive argument system $\mathsf{ExArg}^{\nabla} = (\mathsf{Setup}, \mathsf{Prove}, \mathsf{Verify})$ as follows.

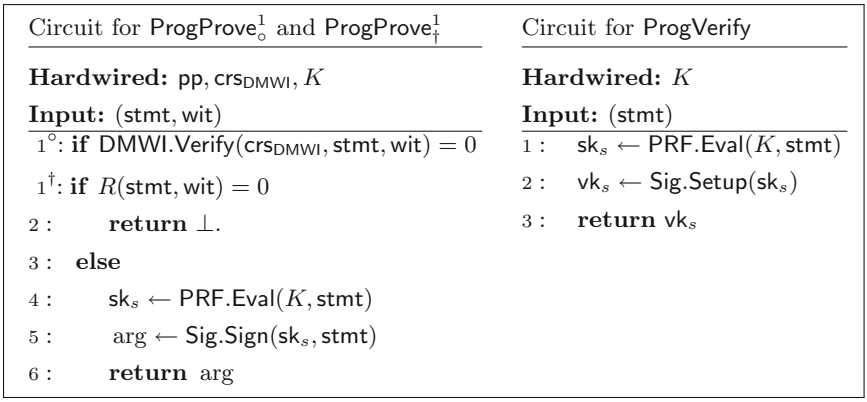

Fig. 1. Circuits for $\mathsf{ProgProve}^1_\circ$, $\mathsf{ProgProve}^1_\dagger$ and ProgVerify. Note that ProgProve differ only in line 1.

$\mathsf{Setup}(\lambda, \mathrm{Ł}_{\mathcal{R}})$:

1. Choose $K \leftarrow \mathsf{PRF.Setup}(\lambda)$ and $\mathsf{crs}_{\mathsf{DMWI}} \leftarrow \mathsf{DMWI.Setup}(\lambda, \mathsf{modeSound}; \mathsf{coins}_S)$, where coins_S are random coins.
2. $O_{\mathsf{Prove}} \leftarrow \mathsf{Obf}(\lambda, \mathsf{ProgProve}^1_\nabla[\mathsf{pp}, \mathsf{crs}_{\mathsf{DMWI}}, K]; \mathsf{coins}_P)$, where $\mathsf{ProgProve}^1_\nabla$ is given by Fig. 1 and coins_P are random coins.

3°. Define statement $\mathsf{stmt}^\circ_{\mathsf{Setup}}$ as

$$\left\{ \begin{array}{c} \exists_{i\in[2],K,\mathsf{coins}_P} O_{\mathsf{Prove}} \leftarrow \mathsf{Obf}(\lambda, \mathsf{ProgProve}^i_\circ[\mathsf{pp}, \mathsf{crs}_{\mathsf{DMWI}}, K]; \mathsf{coins}_P) \ \vee \\ \exists_{\mathsf{mode},\mathsf{coins}_S} \mathsf{crs}_{\mathsf{DMWI}} \leftarrow \mathsf{DMWI.Setup}(\lambda, \mathsf{mode}; \mathsf{coins}_S) \wedge \mathsf{mode} = \mathsf{modeWI} \end{array} \right\}.$$

$3^\dagger$. Define statement $\mathsf{stmt}^\dagger_{\mathsf{Setup}}$ as

$$\{\exists_{K,\mathsf{coins}_P} O_{\mathsf{Prove}} \leftarrow \mathsf{Obf}(\lambda, \mathsf{ProgProve}^1_\dagger[\mathsf{pp}, \mathsf{crs}_{\mathsf{DMWI}}, K]; \mathsf{coins}_P)\}.$$

4. Set $\mathsf{wit}_{\mathsf{Setup}} = (1, K, \mathsf{coins}_P)$.

5°. $\pi \leftarrow \mathsf{NIWI.Prove}(\mathsf{stmt}^\circ_{\mathsf{Setup}}, \mathsf{wit}_{\mathsf{Setup}})$.

$5^\dagger$. $\pi \leftarrow \mathsf{NIZK.Prove}(\mathsf{stmt}^\dagger_{\mathsf{Setup}}, \mathsf{wit}_{\mathsf{Setup}})$.

6. Compute $O_{\mathsf{Verify}} \leftarrow \mathsf{Obf}(\lambda, \mathsf{ProgVerify}[K])$ and output $\mathsf{crs} = (O_{\mathsf{Prove}}, O_{\mathsf{Verify}}, \mathsf{pp}, \mathsf{etd}, \mathsf{crs}_{\mathsf{DMWI}}, \pi)$.

$\mathsf{Prove}(\mathsf{crs}, \mathsf{stmt}, \mathsf{wit}; r)$:

1°. Set $\mathsf{stmt}^\circ_{\mathsf{Setup}}$ as in the setup algorithm.

$1^\dagger$. Set $\mathsf{stmt}^\dagger_{\mathsf{Setup}}$ as in the setup algorithm.

2°. If $\mathsf{NIWI.Verify}(\mathsf{stmt}^\circ_{\mathsf{Setup}}, \pi) = 0$ return $\bot$.

$2^\dagger$. If $\mathsf{NIZK.Verify}(\mathsf{stmt}^\dagger_{\mathsf{Setup}}, \pi) = 0$ return $\bot$.

3°. Run $\mathsf{wit}' \leftarrow \mathsf{DMWI.Prove}(\mathsf{crs}_{\mathsf{DMWI}}, \mathsf{stmt}, \mathsf{wit}; r)$ and $\mathrm{arg} \leftarrow O_{\mathsf{Prove}}(\mathsf{stmt}, \mathsf{wit}')$.

$3^\dagger$. Run $\mathrm{arg} \leftarrow O_{\mathsf{Prove}}(\mathsf{stmt}, \mathsf{wit})$.

4. Run $\mathsf{vk}_s \leftarrow O_{\mathsf{Verify}}(\mathsf{stmt})$.

5. If $\mathsf{Sig.Verify}(\mathsf{vk}_s, \mathrm{arg}, \mathsf{stmt}) \neq 1$ return $\perp$.
6. Otherwise, return arg.

$\mathsf{Verify}(\mathsf{crs}, \mathsf{stmt}, \mathrm{arg})$:
1. Run $\mathsf{vk}_s \leftarrow O_{\mathsf{Verify}}(\mathsf{stmt})$.
2. Output $\mathsf{Sig.Verify}(\mathsf{vk}_s, \mathsf{sig}, \mathsf{msg})$

$\mathsf{Expl}(\mathsf{crs}, \mathsf{stmt}, \mathsf{wit}, \mathrm{arg})$:
1. Output 0.

Circuit for $\mathsf{ProgProve}^2_\circ$ and $\mathsf{ProgProve}^2_\dagger$

Hardwired: $\mathsf{crs}_{\mathsf{DMWI}}, \mathsf{pp}$

$K_{\mathsf{stmt}^*} = \mathsf{PRF.Puncture}(K, \mathsf{stmt}^*)$

Input: $(\mathsf{stmt}, \mathsf{wit}, r)$

1°: **if** $\mathsf{DMWI.Verify}(\mathsf{crs}_{\mathsf{DMWI}}, \mathsf{stmt}, \mathsf{wit}) = 0$

$1^\dagger$: **if** $R(\mathsf{stmt}, \mathsf{wit}) = 0$

2 : **return** $\perp$.

3 : **else**

4 : $\mathsf{sk}_s \leftarrow \mathsf{PRF.Eval}(K_{\mathsf{stmt}^*}, \mathsf{stmt})$

5 : $\mathrm{arg} \leftarrow \mathsf{Sig.Sign}(\mathsf{sk}_s, \mathsf{stmt})$

6 : **return** arg

Circuit for $\mathsf{ProgVerify}^*$

Hardwired: $\mathsf{stmt}^*, \mathsf{vk}^*_s,$

$K_{\mathsf{stmt}^*} = \mathsf{PRF.Puncture}(K, \mathsf{stmt}^*)$

Input: (stmt)

1 : **if** $\mathsf{stmt} = \mathsf{stmt}^*$

2 : **return** vk^*_s

3 : **else**

4 : $\mathsf{sk}_s \leftarrow \mathsf{PRF.Eval}(K_{\mathsf{stmt}^*}, \mathsf{stmt})$

5 : $\mathsf{vk}_s \leftarrow \mathsf{Sig.Setup}(\mathsf{sk}_s)$

6 : **return** vk_s

Fig. 2. Circuits for $\mathsf{ProgProve}^2_\circ$, $\mathsf{ProgProve}^2_\dagger$ and $\mathsf{ProgVerify}^*$ used in the soundness proof of the non-interactive argument.

Theorem 5. *Let* ExArg° *be the system given by Scheme 1. The system* ExArg° *is computationally sound (in the selective setting) assuming indistinguishability obfuscation of* Obf*, pseudorandomness in punctured points of* PRF*, mode indistinguishability of the* DMWI *scheme, and unforgeability of the signature scheme (Fig. 2).*

Theorem 6. *Given that the signature scheme* Sig *is unique,* NIWI *is perfectly sound,* DMWI *is a dual-mode proof, and all primitives are perfectly correct, the argument system* ExArg° *is malicious setup explainable.*

Theorem 7. *Let* $\mathsf{ExArg}^\dagger$ *be the system given by Scheme 1. The system* $\mathsf{ExArg}^\dagger$ *is computationally sound (in the selective setting), assuming indistinguishability obfuscation of* Obf*, pseudorandomness in punctured points of* PRF*, zero-knowledge of the* NIZK *scheme and unforgeability of the signature scheme.*

Theorem 8. *Given that the signature scheme* Sig *is unique,* NIZK *is sound, and all primitives are perfectly correct, argument system* $\mathsf{ExArg}^\dagger$ *is fully explainable.*

Corollary 1. *The scheme is witness indistinguishable against a malicious setup.*

Proof. Witness indistinguishability follows from explainability of the argument system and Theorem 2.

Theorem 9. *Let* ExArg^∇ *be the system given by Scheme 1 for* $\nabla = \circ$ *or* $\nabla = \dagger$. ExArg^∇ *is zero-knowledge in the common reference string model.*

5 Robust-Witness Encryption and Interactive Explainable Arguments

We introduce robust witness encryption and show a generic transformation from any standard witness encryption scheme to a robust witness encryption scheme.

Definition 11 (Robust Witness Encryption). *We call a witness encryption scheme* $\mathsf{WE} = (\mathsf{Enc}, \mathsf{Dec})$ *a robust witness encryption scheme if it is correct, secure and robust as defined below:*

Robustness: *A witness encryption scheme* $(\mathsf{Enc}, \mathsf{Dec})$ *is robust if for all PPT adversaries* $\mathcal{A}$ *there exists a negligible function* $\mathsf{negl}(.)$ *such that*

$$\Pr\left[m_0 \neq m_1 : \begin{array}{c} \mathcal{R}(\mathsf{stmt}, \mathsf{wit}_0) = \mathcal{R}(\mathsf{stmt}, \mathsf{wit}_1) = 1 \wedge \\ (\mathsf{stmt}, \mathsf{ct}, \mathsf{wit}_0, \mathsf{wit}_1) \leftarrow \mathcal{A}(\lambda); \\ m_0 \leftarrow \mathsf{Dec}(\mathsf{stmt}, \mathsf{wit}_0, \mathsf{ct}) \\ m_1 \leftarrow \mathsf{Dec}(\mathsf{stmt}, \mathsf{wit}_1, \mathsf{ct}) \end{array}\right] \leq \mathsf{negl}(\lambda),$$

We call the scheme perfectly robust if the above probability is always zero.

Below we define plaintext awareness [5], but tailored to the case of witness encryption.

Definition 12 (Plaintext Aware Witness Encryption). *Let* $\mathsf{WE} = (\mathsf{Enc}, \mathsf{Dec})$ *be a witness encryption scheme. We extend the scheme with an algorithm* Verify *that on input a ciphertext* ct *and a statement* stmt *outputs a bit indicating whether the ciphertext is in the ciphertext space or not. Additionally we define an algorithm* Setup *that on input the security parameter* λ *outputs a common reference string* crs*, and an algorithm* Setup^* *that additionally outputs* τ*. We say that the witness encryption scheme for a language* $L \in \mathbf{NP}$ *is plaintext aware if for all PPT adversaries* $\mathcal{A}$*, there exists a negligible function* $\mathsf{negl}(.)$ *such that*

$$\begin{aligned} &|\Pr[\mathcal{A}(\mathsf{crs}) = 1 : \mathsf{crs} \leftarrow \mathsf{Setup}(\lambda)] \\ &- \Pr[\mathcal{A}(\mathsf{crs}) = 0 : (\mathsf{crs}, \tau) \leftarrow \mathsf{Setup}^*(\lambda)]| \leq \mathsf{negl}(\lambda), \end{aligned}$$

and there exists a PPT extractor Ext *such that*

$$\Pr\left[\mathsf{msg} \leftarrow \mathsf{Ext}(\mathsf{stmt}, \mathsf{ct}, \tau) : \begin{array}{c} (\mathsf{crs}, \tau) \leftarrow \mathsf{Setup}^*(\lambda); \\ (\mathsf{ct}, \mathsf{stmt}) \leftarrow \mathcal{A}(\mathsf{crs}); \\ \mathsf{Verify}(\mathsf{stmt}, \mathsf{ct}) = 1 \end{array}\right] \leq 1 - \mathsf{negl}(\lambda)$$

where for all witnesses wit *such that* $\mathcal{R}(\mathsf{stmt}, \mathsf{wit}) = 1$ *we have* $\mathsf{msg} = \mathsf{Dec}(\mathsf{ct}, \mathsf{wit})$*, and the probability is taken over the random coins of* Setup *and* Setup^**.*

Scheme 2 (Generic Transformation). Let $\mathsf{WE} = (\mathsf{Enc}, \mathsf{Dec})$ be a witness encryption scheme and $\mathsf{NIZK} = (\mathsf{NIZK.Prove}, \mathsf{NIZK.Verify})$ be a proof system. We construct a robust witness encryption scheme WE_{rob} as follows.

$\mathsf{Enc}_{rob}(\lambda, \mathsf{stmt}, \mathsf{msg})$:

1. Compute $\mathsf{ct}_{\mathsf{msg}} \leftarrow \mathsf{WE.Enc}(\lambda, \mathsf{stmt}, \mathsf{msg})$
2. Let $\mathsf{stmt}_{\mathsf{NIZK}}$ be defined as $\{\exists_{\mathsf{msg}} \ \mathsf{ct}_{\mathsf{msg}} \leftarrow \mathsf{WE.Enc}(\lambda, \mathsf{stmt}, \mathsf{msg})\}$
3. Compute $\pi \leftarrow \mathsf{NIZK.Prove}(\mathsf{stmt}_{\mathsf{NIZK}}, \mathsf{wit})$ using witness $\mathsf{wit} = (\mathsf{msg})$
4. Return $\mathsf{ct} = (\mathsf{ct}_{\mathsf{msg}}, \pi)$.

$\mathsf{Dec}_{rob}(\mathsf{stmt}, \mathsf{wit}, \mathsf{ct})$:

1. Set the statement $\mathsf{stmt}_{\mathsf{NIZK}}$ as $\{\exists_{\mathsf{msg}} \ \mathsf{ct}_{\mathsf{msg}} \leftarrow \mathsf{WE.Enc}(\lambda, \mathsf{stmt}, \mathsf{msg})\}$
2. If $\mathsf{NIZK.Verify}(\mathsf{stmt}_{\mathsf{NIZK}}, \pi) = 0$, then return $\perp$. Otherwise return $\mathsf{WE.Dec}(\mathsf{stmt}, \mathsf{wit}, \mathsf{ct}_{\mathsf{msg}})$

Theorem 10 (Security and Extractability). *Scheme 2 is a (extractably) secure witness encryption if* WE *is a (extractably) secure witness encryption, and* NIZK *is zero-knowledge (in the common reference string or RO model).*

Theorem 11 (Robustness and Plaintext Awareness). *Scheme 2 is robust if the witness encryption scheme* WE *is perfectly correct, and the* NIZK *proof system is perfectly sound (in the common reference string or RO model). If the* NIZK *proof system is a proof of knowledge (in the common string or RO model), then Scheme 2 is plaintext aware.*

5.1 Fully Explainable Arguments from Robust Witness Encryption

In this subsection, we will tackle the problem of constructing fully explainable arguments. The system is described in more detail by Scheme 3.

Scheme 3 (Interactive Explainable Argument). The argument system consists of Prove, Verify and Expl, where the protocol between Prove and Verify is specified as follows. Prove takes as input a statement stmt and a witness wit, and Verify takes as input stmt. First Verify chooses $r \leftarrow_{\$} \{0,1\}^{\lambda}$, computes $\mathsf{ct} \leftarrow \mathsf{Enc}_{rob}(\lambda, \mathsf{stmt}, r)$ and sends ct to Prove. Then Prove computes $\mathsf{arg} \leftarrow \mathsf{Dec}_{rob}(\mathsf{stmt}, \mathsf{wit}, \mathsf{ct})$ and sends arg to Verify. Finally, Verify returns iff $\mathsf{arg} = r$. The explain algorithm Expl is as follows.

$\mathsf{Expl}(\mathsf{stmt}, \mathsf{wit}, \mathsf{trans})$: On input the statement stmt, the witness wit and a transcript trans, output $\perp$.

Theorem 12 (Soundness). *Scheme 3 is an argument system for* NP *language* L *assuming the witness encryption scheme* WE *for* L *is secure. Furthermore, if the underlying witness encryption scheme* WE *scheme is extractable, then Scheme 3 is an argument of knowledge.*

Theorem 13 (Zero-Knowledge). *Scheme 3 is zero-knowledge given the underlying witness encryption scheme* WE *is plaintext aware.*

Theorem 14 (Explainability). *Scheme 3 is fully explainable assuming the used witness encryption scheme is robust (or plaintext aware) and correct.*

Remark 2. Scheme 3 is predictable in the sense that the verifier can "predict" the value of the prover's arguments/proof [33]. Furthermore, the protocol is optimally laconic [12], as the verifier can encrypt single bits.

Theorem 15. *Let* WE *be a (non-robust) perfectly correct witness encryption scheme for* NP*. Let Π be an interactive public-coin zero-knowledge proof protocol for* NP*. Then there exists a malicious verifier explainable (and witness-indistinguishable) argument for* NP*.*

6 Applications

In this section, we show how to apply explainable arguments. We focus on constructing a CCA1 secure publicly deniable encryption scheme using as a building block malicious verifier explainable arguments. Our transformation is based on the one from Naor and Yung [56] but we replace the NIZK proof system with a NIWI. In the full version we show how to build a deniable anonymous credential scheme from malicious prover explainable arguments. Here we note that the anonymous credential system is a straightforward application of malicious prover explainable arguments and standard signature schemes.

The main idea behind the Naor and Yung construction is to use two CPA secure ciphertexts $\mathsf{ct}_1, \mathsf{ct}_2$ and a NIZK that both contain the same plaintext. The soundness property ensures that a decryption oracle can use either of the secret keys (since the decrypted message would be the same) and zero-knowledge allows the security reduction to change the challenged ciphertext, i.e. change the two CPA ciphertexts. We note that in our approach we replace NIZK with NIWI, that to the best of our knowledge has not been do before.

Scheme 4 (Generic Transformation from CPA to CCA). Let $\mathcal{E} = (\mathsf{KeyGen}_{\mathsf{cpa}}, \mathsf{Enc}_{\mathsf{cpa}}, \mathsf{Dec}_{\mathsf{cpa}})$ be a CPA secure encryption scheme, (NIWI.Setup, NIWI.Prove, NIWI.Verify) be a non-interactive witness-indistinguishable proof system. Additionally we define the following statement $\mathsf{stmt}_{\mathsf{cpa}}$ be defined as

$$\{(\exists_{\mathsf{msg}}\ \mathsf{ct}_1 \leftarrow \mathsf{Enc}_{\mathsf{cpa}}(\mathsf{pk}_1, \mathsf{msg}) \ \wedge\ \mathsf{ct}_2 \leftarrow \mathsf{Enc}_{\mathsf{cpa}}(\mathsf{pk}_2, \mathsf{msg})) \ \vee \\ (\exists_{\alpha,\beta} \mathcal{H}_{\mathbb{G}}(\mathsf{ct}_1, \mathsf{ct}_2) = (g^{\alpha}, g^{\beta}, g^{\alpha\cdot\beta}))\},$$

where $\mathcal{H}_{\mathbb{G}}$ is defined as above.

$\mathsf{KeyGen}_{\mathsf{cca1}}(\lambda)$:

1. generate CPA secure encryption key pairs $(\mathsf{pk}_1, \mathsf{sk}_1) \leftarrow \mathsf{KeyGen}_{\mathsf{cpa}}(\lambda)$ and $(\mathsf{pk}_2, \mathsf{sk}_2) \leftarrow \mathsf{KeyGen}_{\mathsf{cpa}}(\lambda)$,
2. generate a common reference string $\mathsf{crs} \leftarrow \mathsf{NIWI.Setup}(\lambda)$,
3. set $\mathsf{pk}_{\mathsf{cca1}} = (\mathsf{pk}_1, \mathsf{pk}_2, \mathsf{crs})$ and $\mathsf{sk}_{\mathsf{cca1}} = \mathsf{sk}_1$.

$\mathsf{Enc}_{\mathsf{cca1}}(\mathsf{pk}_{\mathsf{cca1}}, \mathsf{msg})$:

1. compute ciphertexts $\mathsf{ct}_1 \leftarrow \mathsf{Enc}_{\mathsf{cpa}}(\mathsf{pk}_1, \mathsf{msg})$ and $\mathsf{ct}_2 \leftarrow \mathsf{Enc}_{\mathsf{cpa}}(\mathsf{pk}_2, \mathsf{msg})$,
2. compute NIWI proof $\Pi \leftarrow \mathsf{NIWI.Prove}(\mathsf{crs}, \mathsf{stmt}_{\mathsf{cpa}}, (\mathsf{msg})$,
3. return ciphertext $\mathsf{ct} = (\mathsf{ct}_1, \mathsf{ct}_2, \Pi)$.

$\mathsf{Dec}_{\mathsf{cca1}}(\mathsf{sk}_{\mathsf{cca1}}, \mathsf{ct})$:

1. return $\perp$ if $\mathsf{NIWI.Verify}(\mathsf{crs}, \mathsf{stmt}_{\mathsf{cpa}}, \Pi) = 0$,
2. return $\mathsf{msg} \leftarrow \mathsf{Dec}_{\mathsf{cpa}}(\mathsf{sk}_1, \mathsf{ct}_1)$.

Theorem 16. *Scheme 4 is an encryption scheme secure against non-adaptive chosen ciphertext attacks (CCA1) in the random oracle model assuming the encryption scheme $\mathcal{E}$ is an encryption scheme secure against chosen plaintext attacks and* NIWI *is a sound and witness indistinguishable proof system.*

Theorem 17. *Scheme 4 is an publicly deniable encryption scheme secure against non-adaptive chosen ciphertext attacks (CCA1) in the random oracle model assuming the encryption scheme $\mathcal{E}$ is an publicly deniable encryption scheme secure against chosen plaintext attacks and* NIWI *is a malicious setup explainable argument system.*

7 Conclusions

In this paper, we introduce new security definitions for interactive and non-interactive argument systems that formally capture a property called explainability. Such arguments can be used to construct CCA1 deniable encryption and deniable anonymous authentication. We also introduced a new property for witness encryption called robustness which can be of independent interest. An interesting open question is whether such arguments systems can be constructed from simpler primitives or we need such strong primitives because malicious prover explainability implies uniqueness of the proof.

Acknowledgements. This work has been partially funded/supported by the German Ministry for Education and Research through funding for the project CISPA-Stanford Center for Cybersecurity (Funding numbers: 16KIS0762 and 16KIS0927).

References

1. Apon, D., Fan, X., Liu, F.-H.: Deniable attribute based encryption for branching programs from LWE. In: Hirt, M., Smith, A. (eds.) TCC 2016-B, Part II. LNCS, vol. 9986, pp. 299–329. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-53644-5_12
2. Babai, L., Moran, S.: Arthur-merlin games: a randomized proof system, and a hierarchy of complexity classes. J. Comput. Syst. Sci. **36**(2), 254–276 (1988)
3. Barak, B., et al.: On the (im)possibility of obfuscating programs. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 1–18. Springer, Heidelberg (2001). https://doi.org/10.1007/3-540-44647-8_1
4. Barak, B., Ong, S.J., Vadhan, S.: Derandomization in cryptography. In: Boneh, D. (ed.) CRYPTO 2003. LNCS, vol. 2729, pp. 299–315. Springer, Heidelberg (2003). https://doi.org/10.1007/978-3-540-45146-4_18

5. Bellare, M., Rogaway, P.: Optimal asymmetric encryption. In: De Santis, A. (ed.) EUROCRYPT 1994. LNCS, vol. 950, pp. 92–111. Springer, Heidelberg (1995). https://doi.org/10.1007/BFb0053428
6. Benaloh, J.C., Tuinstra, D.: Receipt-free secret-ballot elections (extended abstract). In: 26th ACM STOC, pp. 544–553. ACM Press, May 1994
7. Bendlin, R., Nielsen, J.B., Nordholt, P.S., Orlandi, C.: Lower and upper bounds for deniable public-key encryption. In: Lee, D.H., Wang, X. (eds.) ASIACRYPT 2011. LNCS, vol. 7073, pp. 125–142. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-25385-0_7
8. Bitansky, N., Choudhuri, A.R.: Characterizing deterministic-prover zero knowledge. In: Pass, R., Pietrzak, K. (eds.) TCC 2020, Part I. LNCS, vol. 12550, pp. 535–566. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-64375-1_19
9. Bitansky, N., Paneth, O.: ZAPs and non-interactive witness indistinguishability from indistinguishability obfuscation. In: Dodis, Y., Nielsen, J.B. (eds.) TCC 2015, Part II. LNCS, vol. 9015, pp. 401–427. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-46497-7_16
10. Blum, M., Feldman, P., Micali, S.: Non-interactive zero-knowledge and its applications (extended abstract). In: 20th ACM STOC, pp. 103–112. ACM Press, May 1988
11. Bohli, J.-M., Steinwandt, R.: Deniable group key agreement. In: Nguyen, P.Q. (ed.) VIETCRYPT 2006. LNCS, vol. 4341, pp. 298–311. Springer, Heidelberg (2006). https://doi.org/10.1007/11958239_20
12. Boneh, D., Ishai, Y., Sahai, A., Wu, D.J.: Quasi-optimal SNARGs via linear multiprover interactive proofs. In: Nielsen, J.B., Rijmen, V. (eds.) EUROCRYPT 2018, Part III. LNCS, vol. 10822, pp. 222–255. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-78372-7_8
13. Boneh, D., Waters, B.: Constrained pseudorandom functions and their applications. In: Sako, K., Sarkar, P. (eds.) ASIACRYPT 2013, Part II. LNCS, vol. 8270, pp. 280–300. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-42045-0_15
14. Boyle, E., Goldwasser, S., Ivan, I.: Functional signatures and pseudorandom functions. In: Krawczyk, H. (ed.) PKC 2014. LNCS, vol. 8383, pp. 501–519. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-642-54631-0_29
15. Canetti, R., Dwork, C., Naor, M., Ostrovsky, R.: Deniable encryption. In: Kaliski, B.S. (ed.) CRYPTO 1997. LNCS, vol. 1294, pp. 90–104. Springer, Heidelberg (1997). https://doi.org/10.1007/BFb0052229
16. Canetti, R., Park, S., Poburinnaya, O.: Fully deniable interactive encryption. In: Micciancio, D., Ristenpart, T. (eds.) CRYPTO 2020, Part I. LNCS, vol. 12170, pp. 807–835. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-56784-2_27
17. Chaidos, P., Cortier, V., Fuchsbauer, G., Galindo, D.: BeleniosRF: a non-interactive receipt-free electronic voting scheme. In: Weippl, E.R., Katzenbeisser, S., Kruegel, C., Myers, A.C., Halevi, S. (eds.) ACM CCS 2016, pp. 1614–1625. ACM Press, October 2016
18. Chakraborty, S., Prabhakaran, M., Wichs, D.: Witness maps and applications. In: Kiayias, A., Kohlweiss, M., Wallden, P., Zikas, V. (eds.) PKC 2020, Part I. LNCS, vol. 12110, pp. 220–246. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-45374-9_8
19. Cramer, R., Shoup, V.: Signature schemes based on the strong RSA assumption. In: Motiwalla, J., Tsudik, G. (eds.) ACM CCS 1999, pp. 46–51. ACM Press, November 1999

20. Dachman-Soled, D.: On the impossibility of sender-deniable public key encryption. Cryptology ePrint Archive, Report 2012/727 (2012). https://eprint.iacr.org/2012/727
21. Dachman-Soled, D.: A black-box construction of a CCA2 encryption scheme from a plaintext aware (sPA1) encryption scheme. In: Krawczyk, H. (ed.) PKC 2014. LNCS, vol. 8383, pp. 37–55. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-642-54631-0_3
22. De Caro, A., Iovino, V., O'Neill, A.: Deniable functional encryption. In: Cheng, C.-M., Chung, K.-M., Persiano, G., Yang, B.-Y. (eds.) PKC 2016, Part II. LNCS, vol. 9614, pp. 196–222. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-49384-7_8
23. De Santis, A., Micali, S., Persiano, G.: Non-interactive zero-knowledge proof systems. In: Pomerance, C. (ed.) CRYPTO 1987. LNCS, vol. 293, pp. 52–72. Springer, Heidelberg (1988). https://doi.org/10.1007/3-540-48184-2_5
24. Di Raimondo, M., Gennaro, R.: New approaches for deniable authentication. In: Atluri, V., Meadows, C., Juels, A. (eds.) ACM CCS 2005, pp. 112–121. ACM Press, November 2005
25. Di Raimondo, M., Gennaro, R.: New approaches for deniable authentication. J. Cryptol. **22**(4), 572–615 (2009)
26. Di Raimondo, M., Gennaro, R., Krawczyk, H.: Deniable authentication and key exchange. In: Juels, A., Wright, R.N., Capitani di Vimercati, S.D. (eds.) ACM CCS 2006, pp. 400–409. ACM Press, October/November 2006
27. Dodis, Y., Fiore, D.: Interactive encryption and message authentication. In: Abdalla, M., De Prisco, R. (eds.) SCN 2014. LNCS, vol. 8642, pp. 494–513. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10879-7_28
28. Dodis, Y., Katz, J., Smith, A., Walfish, S.: Composability and on-line deniability of authentication. In: Reingold, O. (ed.) TCC 2009. LNCS, vol. 5444, pp. 146–162. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-00457-5_10
29. Dodis, Y., Yampolskiy, A.: A verifiable random function with short proofs and keys. In: Vaudenay, S. (ed.) PKC 2005. LNCS, vol. 3386, pp. 416–431. Springer, Heidelberg (2005). https://doi.org/10.1007/978-3-540-30580-4_28
30. Dolev, D., Dwork, C., Naor, M.: Non-malleable cryptography (extended abstract). In: 23rd ACM STOC, pp. 542–552. ACM Press, May 1991
31. Dwork, C., Naor, M.: Zaps and their applications. In: 41st FOCS, pp. 283–293. IEEE Computer Society Press, November 2000
32. Dwork, C., Naor, M., Sahai, A.: Concurrent zero-knowledge. In: 30th ACM STOC, pp. 409–418. ACM Press, May 1998
33. Faonio, A., Nielsen, J.B., Venturi, D.: Predictable arguments of knowledge. In: Fehr, S. (ed.) PKC 2017, Part I. LNCS, vol. 10174, pp. 121–150. Springer, Heidelberg (2017). https://doi.org/10.1007/978-3-662-54365-8_6
34. Feige, U., Lapidot, D., Shamir, A.: Multiple noninteractive zero knowledge proofs under general assumptions. SIAM J. Comput. **29**(1), 1–28 (1999)
35. Fiat, A., Shamir, A.: How to prove yourself: practical solutions to identification and signature problems. In: Odlyzko, A.M. (ed.) CRYPTO 1986. LNCS, vol. 263, pp. 186–194. Springer, Heidelberg (1987). https://doi.org/10.1007/3-540-47721-7_12
36. Garg, S., Gentry, C., Sahai, A., Waters, B.: Witness encryption and its applications. In: Boneh, D., Roughgarden, T., Feigenbaum, J. (eds.) 45th ACM STOC, pp. 467–476. ACM Press, June 2013
37. Gennaro, R., Halevi, S., Rabin, T.: Secure hash-and-sign signatures without the random oracle. In: Stern, J. (ed.) EUROCRYPT 1999. LNCS, vol. 1592, pp. 123–139. Springer, Heidelberg (1999). https://doi.org/10.1007/3-540-48910-X_9

38. Goldreich, O.: Basing non-interactive zero-knowledge on (enhanced) trapdoor permutations: the state of the art. In: Goldreich, O. (ed.) Studies in Complexity and Cryptography. Miscellanea on the Interplay between Randomness and Computation. LNCS, vol. 6650, pp. 406–421. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-22670-0_28
39. Goldreich, O., Levin, L.A.: A hard-core predicate for all one-way functions. In: 21st ACM STOC, pp. 25–32. ACM Press, May 1989
40. Goldwasser, S., Kalai, Y.T., Popa, R.A., Vaikuntanathan, V., Zeldovich, N.: How to run turing machines on encrypted data. In: Canetti, R., Garay, J.A. (eds.) CRYPTO 2013, Part II. LNCS, vol. 8043, pp. 536–553. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-40084-1_30
41. Goldwasser, S., Klein, S., Wichs, D.: The edited truth. In: Kalai, Y., Reyzin, L. (eds.) TCC 2017, Part I. LNCS, vol. 10677, pp. 305–340. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-70500-2_11
42. Goldwasser, S., Ostrovsky, R.: *Invariant* signatures and non-interactive zero-knowledge proofs are equivalent. In: Brickell, E.F. (ed.) CRYPTO 1992. LNCS, vol. 740, pp. 228–245. Springer, Heidelberg (1993). https://doi.org/10.1007/3-540-48071-4_16
43. Groth, J., Ostrovsky, R., Sahai, A.: Non-interactive zaps and new techniques for NIZK. In: Dwork, C. (ed.) CRYPTO 2006. LNCS, vol. 4117, pp. 97–111. Springer, Heidelberg (2006). https://doi.org/10.1007/11818175_6
44. Groth, J., Ostrovsky, R., Sahai, A.: Perfect non-interactive zero knowledge for NP. In: Vaudenay, S. (ed.) EUROCRYPT 2006. LNCS, vol. 4004, pp. 339–358. Springer, Heidelberg (2006). https://doi.org/10.1007/11761679_21
45. Hanzlik, L., Kluczniak, K.: Explainable arguments. Cryptology ePrint Archive, Report 2021/xxxx (2021, to appear). https://ia.cr/2021/xxxx
46. Hanzlik, L., Kluczniak, K., Kutyłowski, M., Krzywiecki, Ł: Mutual restricted identification. In: Katsikas, S., Agudo, I. (eds.) EuroPKI 2013. LNCS, vol. 8341, pp. 119–133. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-642-53997-8_8
47. Hirt, M., Sako, K.: Efficient receipt-free voting based on homomorphic encryption. In: Preneel, B. (ed.) EUROCRYPT 2000. LNCS, vol. 1807, pp. 539–556. Springer, Heidelberg (2000). https://doi.org/10.1007/3-540-45539-6_38
48. Jakobsson, M., Sako, K., Impagliazzo, R.: Designated verifier proofs and their applications. In: Maurer, U. (ed.) EUROCRYPT 1996. LNCS, vol. 1070, pp. 143–154. Springer, Heidelberg (1996). https://doi.org/10.1007/3-540-68339-9_13
49. Jiang, S., Safavi-Naini, R.: An efficient deniable key exchange protocol (extended abstract). In: Tsudik, G. (ed.) FC 2008. LNCS, vol. 5143, pp. 47–52. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-85230-8_4
50. Kiayias, A., Papadopoulos, S., Triandopoulos, N., Zacharias, T.: Delegatable pseudorandom functions and applications. In: Sadeghi, A.-R., Gligor, V.D., Yung, M. (eds.) ACM CCS 2013, pp. 669–684. ACM Press, November 2013
51. Krzywiecki, L., Kluczniak, K., Kozieł, P., Panwar, N.: Privacy-oriented dependency via deniable sigma protocol. Comput. Secur. **79**, 53–67 (2018)
52. Lysyanskaya, A.: Unique signatures and verifiable random functions from the DH-DDH separation. In: Yung, M. (ed.) CRYPTO 2002. LNCS, vol. 2442, pp. 597–612. Springer, Heidelberg (2002). https://doi.org/10.1007/3-540-45708-9_38
53. Micali, S., Rabin, M.O., Vadhan, S.P.: Verifiable random functions. In: 40th FOCS, pp. 120–130. IEEE Computer Society Press, October 1999

54. Moran, T., Naor, M.: Receipt-free universally-verifiable voting with everlasting privacy. In: Dwork, C. (ed.) CRYPTO 2006. LNCS, vol. 4117, pp. 373–392. Springer, Heidelberg (2006). https://doi.org/10.1007/11818175_22
55. Naor, M.: Deniable ring authentication. In: Yung, M. (ed.) CRYPTO 2002. LNCS, vol. 2442, pp. 481–498. Springer, Heidelberg (2002). https://doi.org/10.1007/3-540-45708-9_31
56. Naor, M., Yung, M.: Public-key cryptosystems provably secure against chosen ciphertext attacks. In: 22nd ACM STOC, pp. 427–437. ACM Press, May 1990
57. O'Neill, A., Peikert, C., Waters, B.: Bi-deniable public-key encryption. In: Rogaway, P. (ed.) CRYPTO 2011. LNCS, vol. 6841, pp. 525–542. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-22792-9_30
58. Park, S., Sealfon, A.: It wasn't me! Repudiability and claimability of ring signatures. In: Boldyreva, A., Micciancio, D. (eds.) CRYPTO 2019, Part III. LNCS, vol. 11694, pp. 159–190. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-26954-8_6
59. Pass, R.: On deniability in the common reference string and random oracle model. In: Boneh, D. (ed.) CRYPTO 2003. LNCS, vol. 2729, pp. 316–337. Springer, Heidelberg (2003). https://doi.org/10.1007/978-3-540-45146-4_19
60. Peikert, C., Shiehian, S.: Noninteractive zero knowledge for NP from (plain) learning with errors. In: Boldyreva, A., Micciancio, D. (eds.) CRYPTO 2019, Part I. LNCS, vol. 11692, pp. 89–114. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-26948-7_4
61. Rivest, R.L., Shamir, A., Tauman, Y.: How to leak a secret. In: Boyd, C. (ed.) ASIACRYPT 2001. LNCS, vol. 2248, pp. 552–565. Springer, Heidelberg (2001). https://doi.org/10.1007/3-540-45682-1_32
62. Ryan, P.Y.A., Rønne, P.B., Iovino, V.: Selene: voting with transparent verifiability and coercion-mitigation. In: Clark, J., Meiklejohn, S., Ryan, P.Y.A., Wallach, D., Brenner, M., Rohloff, K. (eds.) FC 2016. LNCS, vol. 9604, pp. 176–192. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-53357-4_12
63. Sahai, A., Waters, B.: How to use indistinguishability obfuscation: deniable encryption, and more. In: Shmoys, D.B. (ed.) 46th ACM STOC, pp. 475–484. ACM Press, May/June (2014)
64. Sako, K., Kilian, J.: Receipt-free mix-type voting scheme. In: Guillou, L.C., Quisquater, J.-J. (eds.) EUROCRYPT 1995. LNCS, vol. 921, pp. 393–403. Springer, Heidelberg (1995). https://doi.org/10.1007/3-540-49264-X_32
65. Unger, N., Goldberg, I.: Improved strongly deniable authenticated key exchanges for secure messaging. Proc. Priv. Enhanc. Technol. **2018**(1), 21–66 (2018)
66. Unger, N., Goldberg, I.: Deniable key exchanges for secure messaging. In: Ray, I., Li, N., Kruegel, C. (eds.) ACM CCS 2015, pp. 1211–1223. ACM Press, October 2015
67. Vatandas, N., Gennaro, R., Ithurburn, B., Krawczyk, H.: On the cryptographic deniability of the signal protocol. In: Conti, M., Zhou, J., Casalicchio, E., Spognardi, A. (eds.) ACNS 2020, Part II. LNCS, vol. 12147, pp. 188–209. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-57878-7_10

68. Yamada, S., Attrapadung, N., Santoso, B., Schuldt, J.C.N., Hanaoka, G., Kunihiro, N.: Verifiable predicate encryption and applications to CCA security and anonymous predicate authentication. In: Fischlin, M., Buchmann, J., Manulis, M. (eds.) PKC 2012. LNCS, vol. 7293, pp. 243–261. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-30057-8_15
69. Yao, A.C.-C., Zhao, Y.: Deniable internet key exchange. In: Zhou, J., Yung, M. (eds.) ACNS 2010. LNCS, vol. 6123, pp. 329–348. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-13708-2_20