Strengthening Access Control Encryption

Christian Badertscher, Christian Matt, and Ueli Maurer
Department of Computer Science, ETH Zurich, 8092 Zurich, Switzerland.
{badi, mattc, maurer}@inf.ethz.ch

Abstract

Access control encryption (ACE) was proposed by Damgård et al. to enable the control of information flow between several parties according to a given policy specifying which parties are, or are not, allowed to communicate. By involving a special party, called the *sanitizer*, policy-compliant communication is enabled while policy-violating communication is prevented, even if sender and receiver are dishonest. To allow outsourcing of the sanitizer, the secrecy of the message contents and the anonymity of the involved communication partners is guaranteed.

This paper shows that in order to be resilient against realistic attacks, the security definition of ACE must be considerably strengthened in several ways. A new, substantially stronger security definition is proposed, and an ACE scheme is constructed which provably satisfies the strong definition under standard assumptions.

Three aspects in which the security of ACE is strengthened are as follows. First, CCA security (rather than only CPA security) is guaranteed, which is important since senders can be dishonest in the considered setting. Second, the revealing of an (unsanitized) ciphertext (e.g., by a faulty sanitizer) cannot be exploited to communicate more in a policy-violating manner than the information contained in the ciphertext. We illustrate that this is not only a definitional subtlety by showing how in known ACE schemes, a single leaked unsanitized ciphertext allows for an arbitrary amount of policy-violating communication. Third, it is enforced that parties specified to receive a message according to the policy cannot be excluded from receiving it, even by a dishonest sender.

1 Introduction

1.1 Access Control Encryption – Model and Security Requirements

The concept of *access control encryption (ACE)* was proposed by Damgård, Haagh, and Orlandi [DHO16] in order to enforce information flow using cryptographic tools rather than a standard access control mechanism (e.g., a reference monitor) within an information system. If the encryption scheme provides certain operations (e.g., ciphertext sanitization) and satisfies an adequate security definition, then the reference monitor can be outsourced, as a component called the *sanitizer*, to an only partially trusted service provider. The goal of ACE is that the sanitizer learns nothing not intrinsically necessary. Security must also be guaranteed against dishonest users, whether senders or receivers of information, and against certain types of sanitizer misbehavior.

The information flow problem addressed by ACE is defined in a context with a set $\mathcal{R}$ of roles corresponding, for example, to different security clearances. Each user in a system can be assigned several roles. For example the users are employees of a company collaborating on a sensitive project, and they need to collaborate and exchange information by sending messages. Since the information is sensitive, which information a party can see must be restricted (hence the term *access control*), even if some parties are dishonest. In the most general form, the specification of which role may send to which other role corresponds to a relation (a subset of $\mathcal{R} \times \mathcal{R}$) or, equivalently, to a predicate $P\colon \mathcal{R} \times \mathcal{R} \to \{0,1\}$, where $s \in \mathcal{R}$ is allowed to communicate to $r \in \mathcal{R}$ if and only if $P(s,r) = 1$. The predicate P can be called the *security policy*. Typical examples of such policies arise from the Bell-LaPadula [BL73] model where roles are (partially) ordered, and the so-called "no-write-down" rule specifies that it is forbidden for a user to send information to another user with a lower role. Note that for this specific example, the relation is transitive, but ACE also allows to capture non-transitive security policies.

ACE was designed to work in the following setting. Users can communicate anonymously with a sanitizer. If a user wants to send a message, it is encrypted under a key corresponding to the sender's role. Then the ciphertext is sent (anonymously) to the sanitizer who applies a certain sanitization operation and writes the sanitized ciphertext on a publicly readable bulletin board providing anonymous read-access to the users (receivers). Users who are supposed to receive the message according to the policy (and only those users) can decrypt the sanitized ciphertext.

To ensure security in the described setting, the ACE scheme must at least provide the following guarantees:

1. The encryption must assure privacy and anonymity against dishonest receivers as well as the sanitizer, i.e., neither the sanitizer nor dishonest receivers without access allowed by the policy should be able to obtain information about messages or the sender role.

2. A dishonest sender must be unable to communicate with a dishonest receiver, unless this is allowed according to the policy. In other words, the system must not provide covert channels allowing for policy-violating communication.

As usual in a context with dishonest senders, the first goal requires security against chosen-ciphertext attacks (CCA) because dishonest users can send a ciphertext for which they do not know the contained message and by observing the effects the received message has on the environment, potentially obtain information about the message. This corresponds to the availability of a decryption oracle, as in the CCA-security definition.

Note that the second goal is only achievable if users cannot directly write to the repository or communicate by other means bypassing the sanitizer, and if the sanitizer is not actively dishonest because a dishonest sanitizer can directly write any information received from a dishonest sender to the repository. The assumption that a user cannot bypass the sanitizer and communicate to another party outside of the system can for example be justified by assuming that users, even if dishonest, want to avoid being caught communicating illegitimately, or if only a user's system (not the user) is corrupted, and the system can technically only send message to the sanitizer.

Since the sanitizer is not fully trusted in our setting, one should consider the possibility that an unsanitized ciphertext is leaked (intentionally or unintentionally) to a dishonest party. This scenario can be called *(unsanitized) ciphertext-revealing attack*. Obviously, all information contained in this ciphertext gets leaked to that party. While this cannot be avoided, such attack should not enable dishonest parties to violate the security requirements beyond that.

We point out that previously proposed encryption techniques (before ACE), such as attribute-based encryption [SW05; GPSW06] and functional encryption [BSW11], enable the design of schemes where a sender can encrypt messages such that only designated receivers (who possess the required key) can read the message. This captures the access control aspects of *read* permissions, but it does not allow to capture the control of *write/send* permissions. In other words, such schemes only achieve the first goal listed above, not the second one.

1.2 Contributions of This Paper

While the proposal of the ACE-concept and of efficient ACE-schemes were important first steps towards outsourcing access control, the existing security definition turns out to be insufficient for several realistic attack scenarios. The main contributions of this paper consist of uncovering issues with existing definitions and schemes, fixing these issues by proposing stronger security notions, and constructing a scheme satisfying our stronger notions.

Issues with existing definitions and schemes. As argued above, chosen-ciphertext attacks should be considered since the use case for ACE includes dishonest senders. Existing definitions, however, do not take this into account, i.e., the adversary does not have access to a decryption oracle in the security games.

Furthermore, existing notions do not consider ciphertext-revealing attacks. Technically speaking, the security game that is supposed to prevent dishonest senders from transmitting information to dishonest receivers (called no-write game), gives the adversary only access to an encryption oracle that sanitizes ciphertexts before returning them. This means that the adversary has no access to unsanitized ciphertexts. This is not only a definitional subtlety, but can completely break down any security guarantees. We demonstrate that existing ACE schemes allow the following attack: Assume there are three users A, M, and E in the system, where A is honest and by the policy allowed to send information to E, and M and E are dishonest and not allowed to communicate. If A sends an (innocent) message to E and the corresponding unsanitized ciphertext is leaked to M, malleability of the ciphertext can be exploited by M to subsequently communicate an arbitrary number of arbitrary messages chosen by M to E. Note that while this attack crucially exploits malleability of ciphertexts, it is not excluded by CCA security for two reasons: first, CCA security does not prevent an adversary from producing valid ciphertexts for *unrelated* messages, and second, the integrity should still hold if the adversary has the decryption key (but not the encryption key).

Finally, existing security definitions focus on preventing dishonest parties from communicating if disallowed by the policy, but they do not enforce information flow. For example, if user A only has a role such that according to the policy, users B and C can read what A sends, existing schemes do not prevent A from sending a message that can be read by B but not by C, or sending a message such that B and C receive different messages. This is not as problematic as the two issues above, and one can argue that A could anyway achieve something similar by additionally encrypting the message with another encryption scheme. Nevertheless, for some use cases, actually precisely enforcing the policy can be required (consider, e.g., a logging system), and one might intuitively expect that ACE schemes achieve this.

New security definitions. We propose new, stronger security definitions for ACE that exclude all issues mentioned above. First, we give the adversary access to a decryption oracle. More precisely, the oracle first sanitizes the given ciphertext and then decrypts it, since this is what happens in the application if a dishonest party sends a ciphertext to the sanitizer. Second, we incorporate ciphertext-revealing attacks by giving the adversary access to an encryption oracle that returns unsanitized ciphertexts for arbitrary roles. Finally, we introduce a new security game in which an adversary can obtain encryption keys and decryption keys from an oracle and has to output a ciphertext such that one of the following events occur: either the set of roles that can successfully decrypt the ciphertext (to an arbitrary message) is inconsistent with the policy for all sender roles for which the adversary has an encryption key (in this case, we say the adversary is not *role-respecting*); or the ciphertext can be successfully decrypted with two keys such that two different messages are obtained (in this case, we say the *uniform-decryption* property is violated).

Construction of an ACE scheme for our stronger notions. Our construction proceeds in three steps and follows the general structure of the generic construction by Fuchsbauer et al. [FGKO17]. Since we require much stronger security notions in all three steps, our constructions and proofs are consequently more involved than existing ones. First, we construct a scheme for a primitive we call *enhanced sanitizable public-key encryption (sPKE)*. Second, we use an sPKE scheme to construct an ACE scheme satisfying our strong security notion for the equality policy, i.e., for the policy that allows s to send to r if and only if $r = s$. Third, we show how to lift an ACE scheme for the equality policy to an ACE scheme for the disjunction of equalities policy. This policy encodes roles as vectors $\mathbf{x} = (x_1, \ldots, x_\ell)$ and allows role $\mathbf{x}$ to send to role $\mathbf{y}$ if and only if $x_1 = y_1 \vee \ldots \vee x_\ell = y_\ell$. As shown by Fuchsbauer et al. [FGKO17], useful policies including the inequality predicate corresponding to the Bell-LaPadula model can efficiently be implemented using this policy by encoding the roles appropriately.

Enhanced sanitizable PKE. An sPKE scheme resembles publicy-key encryption with an additional setup algorithm that outputs sanitization parameters and a master secret key. The master secret key is needed to generate a public/private key pair and the sanitization parameters can be used to sanitize a ciphertext. A sanitized ciphertext cannot be linked to the original ciphertext without the decryption key. We require the scheme to be CCA secure (with respect to a sanitize-then-decrypt oracle) and anonymous. Sanitization resembles rerandomization [Gro04; PR07], also called universal re-encryption [GJJS04], but we allow sanitized ciphertexts to be syntactically different form unsanitized ciphertexts. This allows us to achieve full CCA security, which is needed for our ACE construction and unachievable for rerandomizable encryption.

Our scheme is based on ElGamal encryption [Elg85], which can easily be rerandomized and is anonymous. We obtain CCA security using the technique of Naor and Yung [NY90], i.e., encrypting the message under two independent keys and proving in zero-knowledge that the ciphertexts are encryptions of the same message, which was shown to achieve full CCA security if the zero-knowledge proof is simulation-sound by Sahai [Sah99]. A technical issue is that if the verification of the NIZK proof was done by the decrypt algorithm, the sanitization would also need to sanitize the proof. Instead, we let the sanitizer perform the verification. Since we want to preserve anonymity, this needs to be done without knowing under which public keys the message was encrypted. Therefore, the public keys are part of the witness in the NIZK proof. Now the adversary could encrypt the same message under two different public keys that were not produced together by the key-generation, which would break the reduction. To prevent this, the pair of public keys output by the key-generation is signed using a signature key that is contained in the master secret key and the corresponding verification key is contained in the sanitizer parameters.

ACE for equality. The basic idea of our ACE scheme for the equality policy is to use for each role, encryption and decryption keys of an sPKE scheme as the encryption and decryption keys of the ACE scheme, respectively. Since we need to prevent dishonest senders without an encryption key for some role from producing valid ciphertexts for that role even after seeing encryptions of other messages for this role and obtaining encryption keys for other roles, we add a signature key to the encryption key, sign this pair using a separate signing key, where the corresponding verification key is part of the sanitization parameters, and let senders sign their ciphertexts. To preserver anonymity, this signature cannot be part of the ciphertext. Instead, senders prove in zero-knowledge that they know such a signature and that the encryption was performed properly.

ACE for disjunction of equalities. The first step of our lifting is identical to the lifting described by Fuchsbauer et al. [FGKO17]: for each component of the role-vector, the encryption and decryption keys contain corresponding keys of an ACE scheme for the equality policy. To encrypt a message, this message is encrypted under each of the key-components. In a second step, we enforce role-respecting security with the same trick we used in our ACE scheme for equality; that is, we sign encryption key-vectors together with a signing key for that role, and senders prove in zero-knowledge that they have used a valid key combination to encrypt and that they know a signature of the ciphertext vector.

1.3 Related Work

The concept of access control encryption has been introduced by Damgård et al. [DHO16]. They provided the original security definitions and first schemes. Subsequent work by Fuchsbauer et al. [FGKO17] and by Tan et al. [TZMT17] focused on new schemes that are more efficient and based on different assumptions, respectively. In contrast to our work, they did not attempt to strengthen the security guarantees provided by ACE.

2 Preliminaries

2.1 Notation

We write $x \leftarrow y$ for assigning the value y to the variable x. For a finite set X, $x \twoheadleftarrow X$ denotes assigning to x a uniformly random value in X. For $n \in \mathbb{N}$, We use the convention

$$[n] \coloneqq \{1, \ldots, n\}.$$

The probability of an event A in an experiment E is denoted by $\Pr^E[A]$, e.g., $\Pr^{x \twoheadleftarrow \{0,1\}}[x = 0] = \frac{1}{2}$. If the experiment is clear from the context, we omit the superscript. The conditional probability of A given B is denoted by $\Pr[A \mid B]$ and the complement of A is denoted by $\neg A$. For a probabilistic algorithm $\mathcal{A}$ and $r \in \{0,1\}^*$, we denote by $\mathcal{A}(x; r)$ the execution of $\mathcal{A}$ on input x with randomness r. For algorithms $\mathcal{A}$ and $\mathcal{O}$, $\mathcal{A}^{\mathcal{O}(\cdot)}(x)$ denotes the execution of $\mathcal{A}$ on input x, where $\mathcal{A}$ has oracle access to $\mathcal{O}$.

2.2 Security Definitions and Advantages

We define the security of a scheme via a random experiment (or game) involving an adversary algorithm $\mathcal{A}$. For a given scheme $\mathcal{E}$ and adversary $\mathcal{A}$, we define the advantage of $\mathcal{A}$, which is a function of the security parameter κ. To simplify the notation, we omit the security parameter when writing the advantage, e.g., we write $\mathsf{Adv}^{\mathsf{Sig\text{-}EUF\text{-}CMA}}_{\mathcal{E},\mathcal{A}}$ instead of $\mathsf{Adv}^{\mathsf{Sig\text{-}EUF\text{-}CMA}}_{\mathcal{E},\mathcal{A}}(\kappa)$ for the advantage of $\mathcal{A}$ in the existential unforgeability game for the signature scheme $\mathcal{E}$. One can say that a scheme satisfies this notion if all efficient adversaries only have negligible advantage. Following a concrete security treatment, we will, however, only define the advantages and in the security proofs give reductions that relate advantages for different games, without referring to efficiency or negligibility.

2.3 Access Control Encryption

We recall the definition of access control encryption by Damgård et al. [DHO16]. For definitions of other cryptographic primitives used in this paper, see Appendix A. Following Fuchsbauer et al. [FGKO17], we do not have sanitizer keys and require Gen to be deterministic. The set of roles is assumed to be $\mathcal{R} = [n]$.

Definition 2.1. An *access control encryption (ACE) scheme* $\mathcal{E}$ consists of the following five PPT algorithms:

Setup: The algorithm Setup on input a security parameter 1^κ and a *policy* $P\colon [n] \times [n] \to \{0,1\}$, outputs a *master secret key* msk and *sanitizer parameters* sp. We implicitly assume that all keys include the finite *message space* $\mathcal{M}$ and the *ciphertext spaces* $\mathcal{C}, \mathcal{C}'$.

Key Generation: The algorithm Gen is deterministic and on input a master secret key msk, a role $i \in [n]$, and the type $\texttt{sen}$, outputs an *encryption key* ek_i; on input msk, $j \in [n]$, and the type $\texttt{rec}$, outputs a *decryption key* dk_j.

Encrypt: The algorithm Enc on input an encryption key ek_i and a message $m \in \mathcal{M}$, outputs a ciphertext $c \in \mathcal{C}$.

Sanitizer: The algorithm San on input sanitizer parameters sp and a ciphertext $c \in \mathcal{C}$, outputs a *sanitized ciphertext* $c' \in \mathcal{C}' \cup \{\perp\}$.

Decrypt: The algorithm Dec on input a decryption key dk_j and a sanitized ciphertext $c' \in \mathcal{C}'$, outputs a message $m \in \mathcal{M} \cup \{\perp\}$; on input dk_j and $\perp$, it outputs $\perp$.

For a probabilistic algorithm $\mathcal{A}$, consider the experiment $\mathsf{Exp}^{\mathsf{ACE\text{-}corr}}_{\mathcal{E},\mathcal{A}}$ that given a security parameter 1^κ and a policy P, executes $(sp, msk) \leftarrow \mathsf{Setup}(1^\kappa, P)$, $(m, i, j) \leftarrow \mathcal{A}^{\mathsf{Gen}(msk,\cdot,\cdot)}(sp)$, $ek_i \leftarrow \mathsf{Gen}(msk, i, \texttt{sen})$, and $dk_j \leftarrow \mathsf{Gen}(msk, j, \texttt{rec})$. We define the *correctness advantage* of $\mathcal{A}$ (for security parameter κ and policy P) as

$$\mathsf{Adv}^{\mathsf{ACE\text{-}corr}}_{\mathcal{E},\mathcal{A}} := \Pr\bigl[P(i,j) = 1 \ \wedge \ \mathsf{Dec}\bigl(dk_j, \mathsf{San}(sp, \mathsf{Enc}(ek_i, m))\bigr) \neq m\bigr],$$

where the probability is over the randomness in $\mathsf{Exp}^{\mathsf{ACE\text{-}corr}}_{\mathcal{E},\mathcal{A}}$ and the random coins of Enc, San, and Dec.[1]

Remark. Correctness of an encryption scheme is typically not defined via a game with an adversary, but by requiring that decryption of an encryption of m yields m with probability 1. This perfect correctness requirement is difficult to achieve for ACE schemes and not necessary for applications because it is sufficient if a decryption error only occurs with negligible probability in

[1] The scheme $\mathcal{E}$ can be called *correct* if $\mathsf{Adv}^{\mathsf{ACE\text{-}corr}}_{\mathcal{E},\mathcal{A}}$ is negligible for all efficient $\mathcal{A}$. As mentioned in Section 2.2, we do not state this as part of the definition since we follow a concrete security treatment.

any execution of the scheme. Damgård et al. [DHO16] define correctness by requiring that for all m, i, and j with $P(i,j) = 1$, the probability that a decryption fails is negligible, where the probability is over setup, key generation, encrypt, sanitize, and decrypt. While this definition is simpler than ours, it does not guarantee that decryption errors only occur with negligible probability in any execution of the scheme. For example, a scheme could on setup choose a random message m and embed it into all keys such that decryption always fails for encryptions of this particular message. This does not violate the definition by Damgård et al. since for any fixed message, the probability that this message is sampled during setup is negligible (if the message space is large). Nevertheless, an adversary can always provoke a decryption error by sending that particular message m, which is not desirable. The above example might at first sight seem somewhat artificial, and typically, schemes do not have such a structure. However, capturing correctness via an experiment is important when thinking of composition, since we expect that the correctness guarantee still holds when the ACE scheme is run as part of a larger system. In order to meet this expectation, and to exclude the above issue, we formalize correctness via an experiment.

Additionally, Fuchsbauer et al. have defined detectability, which guarantees that decrypting with a wrong key yields $\perp$ with high probability [FGKO17]. This allows receivers to detect whether a message was sent to them. As for correctness, we define it via an experiment. The notion is related to robustness for public-key encryption [ABN10].

Definition 2.2. Let $\mathcal{E} = (\mathsf{Setup}, \mathsf{Gen}, \mathsf{Enc}, \mathsf{San}, \mathsf{Dec})$ be an ACE scheme and let $\mathcal{A}$ be a probabilistic algorithm. Consider the experiment $\mathsf{Exp}^{\mathsf{ACE\text{-}dtct}}_{\mathcal{E},\mathcal{A}}$ that given a security parameter 1^κ and a policy P, executes $(sp) \leftarrow \mathsf{Setup}(1^\kappa, P)$, $(m, i, j) \leftarrow \mathcal{A}^{\mathsf{Gen}(msk,\cdot,\cdot)}(sp, msk)$, $ek_i \leftarrow \mathsf{Gen}(msk, i, \mathtt{sen})$, and $dk_j \leftarrow \mathsf{Gen}(msk, j, \mathtt{rec})$. We define the *detectability advantage* of $\mathcal{A}$ as

$$\mathsf{Adv}^{\mathsf{ACE\text{-}dtct}}_{\mathcal{E},\mathcal{A}} := \Pr\big[P(i,j) = 0 \ \wedge \ \mathsf{Dec}\big(dk_j, \mathsf{San}(sp, \mathsf{Enc}(ek_i, m))\big) \neq \perp\big],$$

where the probability is over the randomness in $\mathsf{Exp}^{\mathsf{ACE\text{-}dtct}}_{\mathcal{E},\mathcal{A}}$ and the random coins of Enc, San, and Dec.

2.4 Existing Security Definitions

Existing notions for ACE specify two core properties: the so-called *no-read rule* and the *no-write rule*. The no-read rule formalizes privacy and anonymity: roughly, an honestly generated ciphertext should not leak anything about the message, except possibly about its length. The ciphertext should in addition not reveal the role of the sender. The security game allows an adversary to interact with a key-generation oracle (to obtain encryption and decryption keys for selected roles), and an encryption oracle to obtain encryptions of chosen messages and roles for which the adversary does not posses the encryption key. This attack model reflects that an adversary cannot obtain useful information by observing the ciphertexts that are sent to the sanitizer. To exclude trivial attacks, it is not considered a privacy breach if the adversary knows a decryption key that allows to decrypt the challenge ciphertext according to the policy. Similarly, it is not considered an anonymity breach if the encrypted messages are different. We state the definition of the no-read rule.[2]

[2] For anonymity, we adopt here the definition of [DHO16], which is stronger than the one used by Fuchsbauer et al. [FGKO17] since there, anonymity is not guaranteed against parties who can decrypt.

Experiment $\mathsf{Exp}_{\mathcal{E},\mathcal{A}}^{\mathsf{ACE\text{-}no\text{-}read}}$

Input: $1^\kappa, P$
$(sp, msk) \leftarrow \mathsf{Setup}(1^\kappa, P)$
$(m_0, m_1, i_0, i_1, st) \leftarrow \mathcal{A}^{\mathcal{O}_{G_1}(\cdot,\cdot),\mathcal{O}_E(\cdot,\cdot)}(sp)$
$b \twoheadleftarrow \{0,1\}$
$ek_{i_b} \leftarrow \mathsf{Gen}(msk, i_b, \mathsf{sen})$
$c \leftarrow \mathsf{Enc}(ek_{i_b}, m_b)$
$b' \leftarrow \mathcal{A}^{\mathcal{O}_{G_2}(\cdot,\cdot),\mathcal{O}_E(\cdot,\cdot)}(st, c)$

Experiment $\mathsf{Exp}_{\mathcal{E},\mathcal{A}}^{\mathsf{ACE\text{-}no\text{-}write}}$

Input: $1^\kappa, P$
$(sp, msk) \leftarrow \mathsf{Setup}(1^\kappa, P)$
$(c_0, i', st) \leftarrow \mathcal{A}^{\mathcal{O}_{G_1}(\cdot,\cdot),\mathcal{O}_E(\cdot,\cdot)}(sp)$
$b \twoheadleftarrow \{0,1\}$
$m' \twoheadleftarrow \mathcal{M}$
$c_1 \leftarrow \mathsf{Enc}(\mathsf{Gen}(msk, i', \mathsf{sen}), m')$
$b' \leftarrow \mathcal{A}^{\mathcal{O}_{G_2}(\cdot,\cdot),\mathcal{O}_{ES}(\cdot,\cdot)}(st, \mathsf{San}(sp, c_b))$

Figure 1: The no-read and no-write experiments for an ACE scheme $\mathcal{E}$ and an algorithm $\mathcal{A}$. The oracles in $\mathsf{Exp}_{\mathcal{E},\mathcal{A}}^{\mathsf{ACE\text{-}no\text{-}write}}$ are defined as $\mathcal{O}_{G_1}(\cdot,\cdot) := \mathcal{O}_{G_2}(\cdot,\cdot) := \mathsf{Gen}(msk,\cdot,\cdot)$, $\mathcal{O}_E(\cdot,\cdot) := \mathsf{Enc}(\mathsf{Gen}(msk,\cdot,\mathtt{sen}),\cdot)$, and $\mathcal{O}_{ES}(\cdot,\cdot) := \mathsf{San}(sp, \mathsf{Enc}(\mathsf{Gen}(msk,\cdot,\mathtt{sen}),\cdot))$.

Definition 2.3. Let $\mathcal{E} = (\mathsf{Setup}, \mathsf{Gen}, \mathsf{Enc}, \mathsf{San}, \mathsf{Dec})$ be an ACE scheme and let $\mathcal{A}$ be a probabilistic algorithm. Consider the experiment $\mathsf{Exp}_{\mathcal{E},\mathcal{A}}^{\mathsf{ACE\text{-}no\text{-}read}}$ in Figure 1 and let J be the set of all j such that $\mathcal{A}$ issued the query $(j, \mathtt{rec})$ to the oracle $\mathsf{Gen}(msk,\cdot,\cdot)$. The *payload-privacy advantage* and the *sender-anonymity advantage* of $\mathcal{A}$ are defined as

$$\mathsf{Adv}_{\mathcal{E},\mathcal{A}}^{\mathsf{ACE\text{-}no\text{-}read,priv}} := 2 \cdot \Pr\big[b' = b \;\wedge\; |m_0| = |m_1| \;\wedge\; \forall j \in J\ P(i_0, j) = P(i_1, j) = 0\big] - 1,$$
$$\mathsf{Adv}_{\mathcal{E},\mathcal{A}}^{\mathsf{ACE\text{-}no\text{-}read,anon}} := 2 \cdot \Pr\big[b' = b \;\wedge\; m_0 = m_1 \;\wedge\; \forall j \in J\ P(i_0, j) = P(i_1, j)\big] - 1,$$

respectively, where the probabilities are over the randomness of all algorithms in $\mathsf{Exp}_{\mathcal{E},\mathcal{A}}^{\mathsf{ACE\text{-}no\text{-}read}}$.

The no-write rule of ACE is the core property to capture access control: in a nutshell, if the policy specifies that role i is not allowed to send to role j, then the adversary should not be able to create a ciphertext which, after being sanitized, allows role j to receive any information. To exclude trivial attacks, it is not considered a security breach if the adversary knows the encryption key of a different role i' which is allowed to send information to j. Technically, in the respective security game, the adversary is given a key-generation oracle as above, and in addition an oracle to obtain *sanitized* ciphertexts for selected messages and roles. This attack model corresponds to a setting where an adversary only sees the outputs of a sanitizer, but not its inputs, and in particular no ciphertexts generated for roles for which he does not posses the encryption key. The adversary wins, if he manages to distinguish the sanitized version of a ciphertext of his choice from a sanitized version of a freshly generated ciphertext to a random message, and if he does not obtain the encryption for any role i, and the decryption key of any role j for which $P(i, j) = 1$, as this would trivially allow him to distinguish.

Definition 2.4. Let $\mathcal{E} = (\mathsf{Setup}, \mathsf{Gen}, \mathsf{Enc}, \mathsf{San}, \mathsf{Dec})$ be an ACE scheme and let $\mathcal{A}$ be a probabilistic algorithm. Consider the experiment $\mathsf{Exp}_{\mathcal{E},\mathcal{A}}^{\mathsf{ACE\text{-}no\text{-}write}}$ in Figure 1, let I_1 be the set of all i such that $\mathcal{A}$ issued the query $(i, \mathtt{sen})$ to $\mathcal{O}_{G_1}$, and let J be the set of all j such that $\mathcal{A}$ issued the query $(j, \mathtt{rec})$ to $\mathcal{O}_{G_1}$ or $\mathcal{O}_{G_2}$. We define the *no-write advantage* of $\mathcal{A}$ as

$$\mathsf{Adv}_{\mathcal{E},\mathcal{A}}^{\mathsf{ACE\text{-}no\text{-}write}} := 2 \cdot \Pr\big[b' = b \wedge i \in I_1 \cup \{0\} \wedge \forall i \in I_1\ \forall j \in J\ P(i, j) = 0 \wedge \mathsf{San}(sp, c_0) \neq \bot\big] - 1,$$

where the probability is over the randomness of all algorithms in $\mathsf{Exp}_{\mathcal{E},\mathcal{A}}^{\mathsf{ACE\text{-}no\text{-}write}}$.

3 Ciphertext-Revealing Attacks Against Existing Schemes

3.1 Generic Description of Attack

We describe a fundamental practical issue of schemes which meet the above no-read and no-write definitions and show why the guarantees expected from an ACE scheme need to be strengthened. We show that schemes fulfilling the definition can suffer from what we call a malleability attack, which effectively bypasses the given policy and allows communication that is forbidden by the policy. The attack does not abuse any peculiarities of existing models and in fact only requires that the semi-honest sanitizer shares its inputs and outputs with colluding parties, which is arguably possible when the sanitizer is outsourced. In particular, security against such a sanitizer is desirable from a practical point of view.

We first give a high-level explanation of the attack, formalize it as a second step, and show that several existing schemes are vulnerable. Assume there are three parties, Alice, Bob, and Charlie, each having a different role assigned. We denote by A, B, and C the associated roles. In our example, Alice and Charlie are always honest. Alice is allowed to communicate with Bob and Charlie. Bob is dishonest and forbidden to send messages to Charlie (and to Alice). The attack now proceeds as follows: When Alice sends her first message, Bob requests the corresponding ciphertext and the sanitized ciphertext from the semi-honest sanitizer. He then decrypts the sanitized ciphertext and thus receives the message Alice has sent. With the knowledge of this message, as we show below, he is able to create a valid ciphertext for a chosen message m', which will be correctly sanitized and later decrypted by Charlie, hence allowing unrestricted communication from Bob to Charlie. Details follow.

Assume a policy matrix defined by

$$P(i,j) \coloneqq \begin{cases} 1, & i = \mathsf{A}, \\ 0, & \text{otherwise.} \end{cases}$$

For the sake of presentation, we assume that the ACE scheme $\mathcal{E}$ under consideration enjoys perfect correctness. Also, we assume that the setup-phase has completed and the three parties thus possess the encryption and decryption keys, ek_i and dk_i, respectively. Now, imagine that the ACE scheme admits an efficient function $\mathsf{maul}_{\mathcal{E}}$ with the following property (later we show how to implement such a function for some existing schemes): For all messages m and m', any role i, and sanitizer parameters sp in the range of Setup, and for any fixed randomness r,

$$\mathsf{maul}_{\mathcal{E}}\big(\mathsf{Enc}(ek_i, m; r), sp, m, m'\big) = \mathsf{Enc}(ek_i, m'; r). \tag{1}$$

If such a malleability function exists, the communication policy can be bypassed as follows:

1. Alice encrypts a message $c \leftarrow \mathsf{Enc}(ek_{\mathsf{A}}, m)$ and the sanitizer computes $c' \leftarrow \mathsf{San}(sp, c)$ and gives c and c' to Bob.

2. Bob computes $m \leftarrow \mathsf{Dec}(dk_{\mathsf{B}}, c')$ and creates a new ciphertext $\hat{c} \leftarrow \mathsf{maul}_{\mathcal{E}}(c, sp, m, m')$ and sends it to the sanitizer.

3. The ciphertext is sanitized $\hat{c}' \leftarrow \mathsf{San}(sp, \hat{c})$ and subsequently sent to Charlie. By the (perfect) correctness of the assumed ACE scheme and by our assumption on $\mathsf{maul}_{\mathcal{E}}$, $\hat{c}'$ is a valid ciphertext (under the encryption key of Alice) and Charlie is able to decrypt $m' \leftarrow \mathsf{Dec}(dk_{\mathsf{C}}, \hat{c}')$, effectively receiving Bob's message m'.

In the following sections, we show that several existing ACE schemes $\mathcal{E}$ admit an efficient function $\mathsf{maul}_{\mathcal{E}}$. More specifically, we consider the "linear" scheme by Damgård et al. [DHO16] based on ElGamal and the ElGamal-based scheme by Fuchsbauer et al. [FGKO17].

3.2 DHO Scheme Based on ElGamal

We briefly recall the ElGamal based ACE scheme for a single identity. The public parameters of the scheme contain the description of a finite cyclic group $G = \langle g \rangle$ and its group order q, and additionally an element $h = g^x$ for a uniform random $x \in \mathbb{Z}_q$. The encryption key for A is a random value $ek \in \mathbb{Z}_q$, and the decryption key is $-x$. The algorithm Enc on input an encryption key ek_i and a message $m \in \mathcal{M}$, samples $r_1, r_2 \in \mathbb{Z}_q$ uniformly at random and outputs the ciphertext

$$c = (c_0, c_1, c_2, c_3) \coloneqq (g^{r_1}, h^{r_1} g^{ek_i}, g^{r_2}, m \cdot h^{r_2}).$$

We can define the function $\mathsf{maul}_{\mathsf{DHO}}$ as

$$\mathsf{maul}_{\mathsf{DHO}}\big((c_0, c_1, c_2, c_3), sp, m, m'\big) \coloneqq \big(c_0, c_1, c_2, m' \cdot m^{-1} \cdot c_3\big).$$

Since the group order q is part of sp, this function is efficiently computable. For $c_3 = m \cdot h^{r_2}$, we thus get a new fourth component $c_3' = m' \cdot h^{r_2}$ and equation (1) is satisfied.

The malleability for more than one identity (and in particular in our scenario described above) follows since the scheme for several identities is composed of independent instances of the basic single-identity scheme.

3.3 FGKO Scheme Based on ElGamal

Description of the scheme. In that scheme, the public parameters consist of the description of a finite cyclic group $G = \langle g \rangle$ including the group order q and a generator g, a verification key vk^{Sig} of a signature scheme Sig, and a common-reference string crs^{NIZK} of a NIZK proof system NIZK for the language $L \coloneqq \{x \mid \exists w \ (x, w) \in R\}$, where R is defined as follows: for $x = \big(vk^{\mathsf{Sig}}, c_0, c_1, c_2, c_3\big)$ and a witness $w = \big(g^x, \sigma^{\mathsf{Sig}}, m, r, s\big)$, $R(x, w) = 1$ if and only if

$$\mathsf{Sig.Ver}\big(vk^{\mathsf{Sig}}, g^x, \sigma^{\mathsf{Sig}}\big) = 1 \ \wedge \ (c_0, c_1, c_2, c_3) = \big(g^r, g^{x \cdot r}, g^s, m \cdot g^{x \cdot s}\big).$$

The encryption and decryption keys are given by $ek \coloneqq (g^x, \sigma^{\mathsf{Sig}})$, $dk \coloneqq x$ for a uniformly chosen $x \twoheadleftarrow \mathbb{Z}_q$, where σ^{Sig} is a signature on g^x. To encrypt a message m, first choose $r \twoheadleftarrow \mathbb{Z}_q^*$ and $s \twoheadleftarrow \mathbb{Z}_q$ uniformly at random and compute $(c_0, c_1, c_2, c_3) \coloneqq (g^r, g^{x \cdot r}, g^s, m \cdot g^{x \cdot s})$. Then run $\pi^{\mathsf{NIZK}} \leftarrow \mathsf{NIZK.Prove}\big(crs^{\mathsf{NIZK}}, (vk^{\mathsf{Sig}}, c_0, c_1, c_2, c_3), (g^x, \sigma^{\mathsf{Sig}}, m, r, s)\big)$ and output the ciphertext $c \coloneqq (c_0, c_1, c_2, c_3, \pi)$.

Potential malleability. We define the function $\mathsf{maul}_{\mathsf{FGKO}}$ as

$$\mathsf{maul}_{\mathsf{FGKO}}\big((c_0, c_1, c_2, c_3, \pi), sp, m, m'\big) \coloneqq \big(c_0, c_1, c_2, m' \cdot m^{-1} \cdot c_3, \pi\big).$$

This function satisfies equation (1) if, for example, the non-interactive zero-knowledge proof is independent of the last component c_3. We show that such a NIZK proof system exists without violating the properties assumed by Fuchsbauer et al. [FGKO17]. To this end, let NIZK' be a

NIZK proof system for the language $L' := \{x \mid \exists w \ (x, w) \in R'\}$, where the relation R' is defined as follows: for $x = \big(vk^{\mathsf{Sig}}, c_0, c_1, c_2\big)$ and $w = \big(g^x, \sigma^{\mathsf{Sig}}, m, r, s\big)$, $(x, w) \in R'$ if and only if

$$\mathsf{Sig.Ver}\big(vk^{\mathsf{Sig}}, g^x, \sigma^{\mathsf{Sig}}\big) = 1 \ \wedge \ (c_0, c_1, c_2) = \big(g^r, g^{x \cdot r}, g^s\big).$$

Given NIZK', we construct a NIZK proof system NIZK for the original language L as follows:

$$\begin{aligned}
&\mathsf{NIZK.Gen}(1^\kappa) := \mathsf{NIZK'.Gen}(1^\kappa),\\
&\mathsf{NIZK.Prove}\big(crs^{\mathsf{NIZK}}, (vk^{\mathsf{Sig}}, c_0, c_1, c_2, c_3), (g^x, \sigma^{\mathsf{Sig}}, m, r, s)\big) :=\\
&\qquad\qquad\qquad \mathsf{NIZK'.Prove}\big(crs^{\mathsf{NIZK}}, (vk^{\mathsf{Sig}}, c_0, c_1, c_2), (g^x, \sigma^{\mathsf{Sig}}, m, r, s)\big),\\
&\mathsf{NIZK.Ver}\big(crs^{\mathsf{NIZK}}, (vk^{\mathsf{Sig}}, c_0, c_1, c_2, c_3), \pi^{\mathsf{NIZK}}\big) := \mathsf{NIZK'.Ver}\big(crs^{\mathsf{NIZK}}, (vk^{\mathsf{Sig}}, c_0, c_1, c_2), \pi^{\mathsf{NIZK}}\big).
\end{aligned}$$

Correctness and zero-knowledge of NIZK follow straightforwardly from the underlying scheme NIZK'. For knowledge-extraction, consider that the underlying NIZK is capable of extracting the valid witness $(g^x, \sigma^{\mathsf{Sig}}, r, s)$ given a valid proof for the statement $(vk^{\mathsf{Sig}}, c_0, c_1, c_2)$. Then, by computing $m := c_3 \cdot (g^{x \cdot s})^{-1}$, we can obtain a valid message encoded in c_3. Finally, for soundness, note that if $(vk^{\mathsf{Sig}}, c_0, c_1, c_2) \in L'$, this implies that any group element $c_3 \in G$ is a valid last component, i.e., $(vk^{\mathsf{Sig}}, c_0, c_1, c_2, c_3) \in L$ for any $c_3 \in G$, since there exists the message $m := c_3 \cdot (g^{x \cdot s})^{-1}$, and thus a valid witness $w = (g^x, \sigma^{\mathsf{Sig}}, m, r, s)$.

For the constructed scheme NIZK and the function $\mathsf{maul}_{\mathsf{FGKO}}$, equation (1) clearly holds. Hence, the FGKO scheme can be instantiated such that the malleability attack works. It could potentially be excluded by requiring stronger properties from the NIZK scheme.

4 A Stronger Notion of ACE

4.1 ACE with Modification Detection

In this section, we introduce our new security definitions, which exclude the issues we have discovered. To be resilient against the ciphertext-revealing attacks described in Section 3, the sanitizer should ideally only sanitize fresh encryptions and block ciphertexts that are either replays or obtained by modifying previous ciphertexts. Therefore, we introduce an additional algorithm for detecting modified ciphertexts. If the sanitizer receives a ciphertext that is detected to be a modification of a previously received one, this ciphertext is blocked. Since such ciphertexts will not be stored in the repository and consequently not be decrypted, we define the chosen-ciphertext security with respect to a decryption oracle that does not return a decryption if the received ciphertext is detected to be a modification of the challenge ciphertext. Our definitions can therefore be seen as a variant of publicly-detectable replayable-CCA security, which was introduced by Canetti et al. [CKN03] for public key encryption. Before defining the security, we define the syntax of ACE schemes with this additional algorithm.

Definition 4.1. An *access control encryption with modification detection scheme* is an ACE scheme $\mathcal{E}$ together with a PPT algorithm DMod that on input sanitizer parameters sp and two ciphertexts $c, \tilde{c} \in \mathcal{C}$, outputs a bit b (where $b = 1$ means that $\tilde{c}$ was obtained via modifying c).

Except for Section 4.3, where we show that our new definitions imply the existing ones, we will in this paper only consider ACE schemes with modification detection and thus often refer to them simply as ACE schemes.

The algorithm DMod should output 1 if $\tilde{c}$ is an adversarial modification of c, and 0 otherwise. We have the following intuitive requirements on DMod:

1. All ciphertexts $\tilde{c}$ an adversary can produce given ciphertexts $c_1, \ldots, c_l$ and no encryption key, are either invalid (i.e., sanitize to $\perp$) or we have $\mathsf{DMod}(sp, c_i, \tilde{c}) = 1$ for some $i \in \{1, \ldots, n\}$.
2. Given encryption and decryption keys, an adversary is unable to produce a ciphertext that is detected to be in relation with a ciphertext produced by Enc for a message of the adversary's choice. In particular, independent encryptions of messages collide only with negligible probability.

The first requirement is captured by role-respecting security as defined in Definition 4.5, the second one by ciphertext unpredictability defined in Definition 4.4.

Remark. Canetti et al. (translated to our setting) also require that if $\mathsf{DMod}(sp, c, \tilde{c}) = 1$, then c and $\tilde{c}$ decrypt to the same message [CKN03]. For our purpose, this is not needed. This means that we do not want to detect replays in the sense that the same message is replayed, but more generally, whether the given ciphertext was obtain via some modification of another ciphertext.

4.2 Security Definitions

We formalize chosen-ciphertext attacks by giving the adversary access to an oracle $\mathcal{O}_{SD}$ that first sanitizes a given ciphertext and then decrypts the result. One could also consider *chosen-sanitized-ciphertext attacks* by providing the adversary access to an oracle $\mathcal{O}_D$ that only decrypts. This is potentially stronger since the adversary can emulate the oracle $\mathcal{O}_{SD}$ by first sanitizing the ciphertexts and then giving the result to $\mathcal{O}_D$, but given $\mathcal{O}_{SD}$, it is not necessarily possible to emulate $\mathcal{O}_D$. Since in the application, users can only send ciphertexts to the sanitizer but not directly write ciphertexts to the repository such that they are decrypted without being sanitized, the weaker notion is sufficient.

In principle, the adversary has in all definitions access to $\mathcal{O}_{SD}$, as well as to an encryption oracle and a key-generation oracle. To simplify the definitions, we omit the encryption or decryption oracles if the winning condition places no restriction on the encryption or decryption keys obtained from the key-generation oracle, respectively.

Privacy and anonymity. We now define (payload) privacy and sender-anonymity. The former guarantees that encryptions of different messages under the same encryption key cannot be distinguished as long as the adversary has no decryption key that allows to decrypt. We also require this for messages of different length, i.e., schemes satisfying our definition do not leak the length of the encrypted message, which means that the message space has to be bounded. Anonymity guarantees that encryptions of the same message under different keys cannot be distinguished. We distinguish a weak and a strong variant of anonymity, where the weak one provides no guarantees if the adversary can decrypt the ciphertext, and the strong one guarantees that even if the adversary has decryption keys, nothing is leaked about the sender role beyond which of the adversary's decryption keys can be used to decrypt.

Definition 4.2. Let $\mathcal{E} = (\mathsf{Setup}, \mathsf{Gen}, \mathsf{Enc}, \mathsf{San}, \mathsf{Dec}, \mathsf{DMod})$, be an ACE with modification detection scheme and let $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ be a pair of probabilistic algorithms. Consider the experiment $\mathsf{Exp}^{\mathsf{ACE\text{-}privAnon\text{-}CCA}}_{\mathcal{E},\mathcal{A}}$ in Figure 2 and let J be the set of all j such that $\mathcal{A}_1$ or $\mathcal{A}_2$ issued the query $(j, \mathtt{rec})$ to the oracle $\mathcal{O}_G$. We define the *privacy advantage* and the *sender-anonymity*

Experiment $\mathsf{Exp}_{\mathcal{E},\mathcal{A}}^{\text{ACE-privAnon-CCA}}$

Input: $(1^\kappa, P), \kappa \in \mathbb{N}, P\colon [n] \times [n] \to \{0,1\}$
$(sp, msk) \leftarrow \mathsf{Setup}(1^\kappa, P)$
$(m_0, m_1, i_0, i_1, st) \leftarrow \mathcal{A}_1^{\mathcal{O}_G(\cdot,\cdot), \mathcal{O}_{SD}(\cdot,\cdot)}(sp)$
$b \twoheadleftarrow \{0,1\}$
$ek_{i_b} \leftarrow \mathsf{Gen}(msk, i_b, \mathtt{sen})$
$c^* \leftarrow \mathsf{Enc}(ek_{i_b}, m_b)$
$b' \leftarrow \mathcal{A}_2^{\mathcal{O}_G(\cdot,\cdot), \mathcal{O}_{SD^*}(\cdot,\cdot)}(st, c^*)$

Experiment $\mathsf{Exp}_{\mathcal{E},\mathcal{A}}^{\text{ACE-san-CCA}}$

Input: $(1^\kappa, P), \kappa \in \mathbb{N}, P\colon [n] \times [n] \to \{0,1\}$
$(sp, msk) \leftarrow \mathsf{Setup}(1^\kappa, P)$
$(c_0, c_1, st) \leftarrow \mathcal{A}_1^{\mathcal{O}_G(\cdot,\cdot), \mathcal{O}_{SD}(\cdot,\cdot)}(sp)$
$c'_0 \leftarrow \mathsf{San}(sp, c_0)$; $c'_1 \leftarrow \mathsf{San}(sp, c_1)$
$b \twoheadleftarrow \{0,1\}$
$b' \leftarrow \mathcal{A}_2^{\mathcal{O}_G(\cdot,\cdot), \mathcal{O}_{SD}(\cdot,\cdot)}(st, c'_b)$
for $j \in [n]$ **do**
  $m_{0,j} \leftarrow \mathsf{Dec}(\mathsf{Gen}(msk, j, \mathtt{rec}), c'_0)$
  $m_{1,j} \leftarrow \mathsf{Dec}(\mathsf{Gen}(msk, j, \mathtt{rec}), c'_1)$

Experiment $\mathsf{Exp}_{\mathcal{E},\mathcal{A}}^{\text{ACE-ctxt-unpred}}$

Input: $(1^\kappa, P), \kappa \in \mathbb{N}, P\colon [n] \times [n] \to \{0,1\}$
$(sp, msk) \leftarrow \mathsf{Setup}(1^\kappa, P)$
$(m, i, c) \leftarrow \mathcal{A}^{\mathcal{O}_G(\cdot,\cdot)}(sp)$
$ek_i \leftarrow \mathsf{Gen}(msk, i, \mathtt{sen})$
$c^* \leftarrow \mathsf{Enc}(ek_i, m)$
$b \leftarrow \mathsf{DMod}(sp, c^*, c)$

Experiment $\mathsf{Exp}_{\mathcal{E},\mathcal{A}}^{\text{ACE-URR}}$

Input: $(1^\kappa, P), \kappa \in \mathbb{N}, P\colon [n] \times [n] \to \{0,1\}$
$(sp, msk) \leftarrow \mathsf{Setup}(1^\kappa, P)$
$c \leftarrow \mathcal{A}^{\mathcal{O}_G(\cdot,\cdot), \mathcal{O}_E(\cdot,\cdot)}(sp)$
$\mathsf{dct} \leftarrow \mathtt{false}$
for $\tilde{c} \in C_{\mathcal{O}}$ **do** $\triangleright$ $C_{\mathcal{O}}$ = set of answers from $\mathcal{O}_E$
  $\mathsf{dct} \leftarrow \mathsf{dct} \vee \mathsf{DMod}(sp, \tilde{c}, c) = 1$
$c' \leftarrow \mathsf{San}(sp, c)$
for $j \in [n]$ **do**
  $m_j \leftarrow \mathsf{Dec}(\mathsf{Gen}(msk, j, \mathtt{rec}), c')$

Definitions of oracles

$$
\begin{aligned}
\mathcal{O}_G(i,t) &\coloneqq \mathsf{Gen}(msk, i, t)\\
\mathcal{O}_E(i,m) &\coloneqq \mathsf{Enc}\big(\mathsf{Gen}(msk, i, \mathtt{sen}), m\big)\\
\mathcal{O}_{SD}(j,c) &\coloneqq \mathsf{Dec}\big(\mathsf{Gen}(msk, j, \mathtt{rec}), \mathsf{San}(sp, c)\big)\\
\mathcal{O}_{SD^*}(j,c) &\coloneqq \begin{cases} \mathsf{Dec}\big(\mathsf{Gen}(msk, j, \mathtt{rec}), \mathsf{San}(sp, c)\big), & \mathsf{DMod}(sp, c^*, c) = 0\\ \mathtt{test}, & \text{else}\end{cases}
\end{aligned}
$$

Figure 2: Security experiments for an ACE with modification detection scheme $\mathcal{E}$ and an adversary $\mathcal{A}$, where $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ in the first two experiments.

advantage of $\mathcal{A}$ as

$$
\begin{aligned}
\mathsf{Adv}_{\mathcal{E},\mathcal{A}}^{\text{ACE-priv-CCA}} &\coloneqq 2 \cdot \Pr\big[b' = b \ \wedge \ i_0 = i_1 \ \wedge \ \forall j \in J \ P(i_0, j) = 0\big] - 1,\\
\mathsf{Adv}_{\mathcal{E},\mathcal{A}}^{\text{ACE-wAnon-CCA}} &\coloneqq 2 \cdot \Pr\big[b' = b \ \wedge \ m_0 = m_1 \ \wedge \ \forall j \in J \ P(i_0, j) = P(i_1, j) = 0\big] - 1,\\
\mathsf{Adv}_{\mathcal{E},\mathcal{A}}^{\text{ACE-sAnon-CCA}} &\coloneqq 2 \cdot \Pr\big[b' = b \ \wedge \ m_0 = m_1 \ \wedge \ \forall j \in J \ P(i_0, j) = P(i_1, j)\big] - 1,
\end{aligned}
$$

respectively, where the probabilities are over the randomness in $\mathsf{Exp}_{\mathcal{E},\mathcal{A}}^{\text{ACE-privAnon-CCA}}$.

Remark. Weak anonymity corresponds to the anonymity considered by Fuchsbauer et al. [FGKO17] and strong anonymity to the one considered by Damgård et al. [DHO16]. We state both definitions because weak anonymity is easier to achieve but strong anonymity might be required by some applications. If anonymity is only required against the sanitizer or if all messages are anyway signed by the sender, weak anonymity is sufficient. Strong anonymity is required in settings where senders also want to retain as much anonymity as possible against legitimate receivers.

Sanitization security. We next define sanitization security, which excludes that dishonest parties can communicate via the ciphertexts. We formalize this by requiring that the output of the sanitizer for two different ciphertexts cannot be distinguished, as long as both sanitized ciphertexts are not $\perp$ and the adversary has no decryption key that decrypts one of the ciphertexts. We do not need to guarantee any security if the adversary can decrypt the ciphertexts since in this case, the parties can directly communicate via the messages. Since the adversary provides the two ciphertexts that are sanitized, we do not know to which roles they correspond; they could even be particularly crafted without belonging to an existing role. Hence, we cannot state the requirement that the adversary should not be able to decrypt by only considering the policy and the obtained decryption keys. Instead, we require that the decryption algorithm returns $\perp$ for all decryption keys the adversary possesses. For this to provide the intended security, we need that the decrypt algorithm returns $\perp$ whenever the receiver role corresponding to the used key is not supposed to read the message. This is guaranteed by role-respecting security which is defined later.

Definition 4.3. Let $\mathcal{E} = (\mathsf{Setup}, \mathsf{Gen}, \mathsf{Enc}, \mathsf{San}, \mathsf{Dec}, \mathsf{DMod})$ be an ACE with modification detection scheme and let $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ be a pair of probabilistic algorithms. Consider the experiment $\mathsf{Exp}^{\mathsf{ACE\text{-}san\text{-}CCA}}_{\mathcal{E},\mathcal{A}}$ in Figure 2 and let J be the set of all j such that $\mathcal{A}_1$ or $\mathcal{A}_2$ issued the query $(j, \mathtt{rec})$ to the oracle $\mathcal{O}_G$. We define the *sanitization advantage* of $\mathcal{A}$ as

$$\mathsf{Adv}^{\mathsf{ACE\text{-}san\text{-}CCA}}_{\mathcal{E},\mathcal{A}} := 2 \cdot \Pr\big[b' = b \ \wedge \ c'_0 \neq \perp \neq c'_1 \ \wedge \ \forall j \in J \ m_{0,j} = m_{1,j} = \perp\big] - 1,$$

where the probability is over the randomness in $\mathsf{Exp}^{\mathsf{ACE\text{-}san\text{-}CCA}}_{\mathcal{E},\mathcal{A}}$.

Ciphertext unpredictability. In the intended way of using a scheme satisfying our notions, the sanitizer only adds sanitized ciphertexts to the repository if the given ciphertext is not detected to be a modification of a previously received ciphertext. This means that if an adversary can find a ciphertext c such that another ciphertext c^* that is later honestly generated is detected as a modification of c, the delivery of the message at that later point can be prevented by sending the ciphertext c to the sanitizer earlier. We exclude this by the following definition, which can be seen as an extended correctness requirement.

Definition 4.4. Let $\mathcal{E} = (\mathsf{Setup}, \mathsf{Gen}, \mathsf{Enc}, \mathsf{San}, \mathsf{Dec}, \mathsf{DMod})$ be an ACE with modification detection scheme and let $\mathcal{A}$ be a probabilistic algorithm. Consider the experiment $\mathsf{Exp}^{\mathsf{ACE\text{-}ctxt\text{-}unpred}}_{\mathcal{E},\mathcal{A}}$ in Figure 2. We define the *ciphertext unpredictability advantage* of $\mathcal{A}$ as

$$\mathsf{Adv}^{\mathsf{ACE\text{-}ctxt\text{-}unpred}}_{\mathcal{E},\mathcal{A}} := \Pr\big[b = 1\big],$$

where the probability is over the randomness in $\mathsf{Exp}^{\mathsf{ACE\text{-}ctxt\text{-}unpred}}_{\mathcal{E},\mathcal{A}}$.

Role-respecting and uniform-decryption security. We finally define role-respecting and uniform-decryption security. The former means that an adversary cannot produce a ciphertext for which the pattern of roles that can decrypt does not correspond to a role for which the adversary has an encryption key. For example, if the adversary has only an encryption key for the role i such that roles j_0 and j_1 are the only roles j with $P(i, j) = 1$, all ciphertexts produced by the adversary are either invalid (i.e., sanitized to $\perp$ or detected as a modification) or decrypt to a message different from $\perp$ precisely under the decryption keys for j_0 and j_1. On the one hand, this means that receivers who are not allowed to receive the message get $\perp$ and hence know that the message is not for them. On the other hand, it also guarantees that the adversary cannot prevent receivers with role j_1 from receiving a message that is sent to receivers with role j_0. Furthermore, uniform decryption guarantees for all ciphertexts c output by an adversary that if c decrypts to a message different from $\perp$ for different decryption keys, it always decrypts to the same message. In the example above, this means that j_0 and j_1 not only both receive *some* message, but they both receive *the same* one.

Definition 4.5. Let $\mathcal{E} = (\mathsf{Setup}, \mathsf{Gen}, \mathsf{Enc}, \mathsf{San}, \mathsf{Dec}, \mathsf{DMod})$, be an ACE with modification detection scheme and let $\mathcal{A}$ be a probabilistic algorithm. Consider the experiment $\mathsf{Exp}^{\mathsf{ACE\text{-}URR}}_{\mathcal{E},\mathcal{A}}$ in Figure 2 and let I and J be the sets of all i and j such that $\mathcal{A}$ issued the query $(i, \mathtt{sen})$ and $(j, \mathtt{rec})$ to the oracle $\mathcal{O}_G$, respectively. We define the *role-respecting advantage* and the *uniform-decryption advantage* of $\mathcal{A}$ as

$$\mathsf{Adv}^{\mathsf{ACE\text{-}RR}}_{\mathcal{E},\mathcal{A}} := \Pr\big[c' \neq \perp \ \wedge \ \mathsf{dct} = \mathtt{false} \ \wedge \ \neg\big(\exists i \in I \ \forall j \in J \ (m_j \neq \perp \leftrightarrow P(i,j) = 1)\big)\big],$$
$$\mathsf{Adv}^{\mathsf{ACE\text{-}uDec}}_{\mathcal{E},\mathcal{A}} := \Pr\big[\exists j, j' \in J \ m_j \neq \perp \neq m_{j'} \ \wedge \ m_j \neq \ m_{j'}\big],$$

respectively, where the probabilities are over the randomness in $\mathsf{Exp}^{\mathsf{ACE\text{-}URR}}_{\mathcal{E},\mathcal{A}}$.

Remark. Note that in Definition 4.5, we only check the decryptions for receiver roles for which $\mathcal{A}$ has requested the corresponding decryption key. This means that an adversary in addition to producing a ciphertext that causes an inconsistency, also has to find a receiver role for which this inconsistency manifests. If the total number of roles n is small (say polynomial in the security parameter), $\mathcal{A}$ can simply query $\mathcal{O}_G$ on all receiver keys, but for large n this condition is nontrivial. For example, we consider a scheme secure if an adversary can efficiently produce a ciphertext such that there is a receiver role that can decrypt it even though the policy does not allow it, as long as this receiver role is hard to find. The rationale is that in this case, the inconsistency cannot be exploited and will only be observed with negligible probability in an execution of the protocol.

4.3 Relation to the Original Security Notions

In this section, we show that our notions imply the original security definitions (see Section 2.4). We first show that the no-read and no-write rules are implied by our new security definitions.

Theorem 4.6. *Let* $\mathcal{E}' = (\mathsf{Setup}, \mathsf{Gen}, \mathsf{Enc}, \mathsf{San}, \mathsf{Dec}, \mathsf{DMod})$ *be an ACE with modification detection scheme and let* $\mathcal{E} = (\mathsf{Setup}, \mathsf{Gen}, \mathsf{Enc}, \mathsf{San}, \mathsf{Dec})$ *be the corresponding ACE scheme. For adversaries* $\mathcal{A}$, $\mathcal{B}$, $\mathcal{C}$ *for the security games of ACE, where we assume that adversary* $\mathcal{A}$ *makes at most* q *queries to its encryption oracle, we derive adversaries* $\mathcal{A}_i$, $\mathcal{B}_i$, *and* $\mathcal{C}_i$ *such that the following inequalities hold:*

$$\begin{aligned}
\mathsf{Adv}^{\mathsf{ACE\text{-}no\text{-}write}}_{\mathcal{E},\mathcal{A}} &\leq \mathsf{Adv}^{\mathsf{ACE\text{-}san\text{-}CCA}}_{\mathcal{E},\mathcal{A}_1} + 2 \cdot \mathsf{Adv}^{\mathsf{ACE\text{-}RR}}_{\mathcal{E},\mathcal{A}_2} \\
&\qquad + 2q \cdot \big(\mathsf{Adv}^{\mathsf{ACE\text{-}san\text{-}CCA}}_{\mathcal{E},\mathcal{A}_3} + \mathsf{Adv}^{\mathsf{ACE\text{-}ctxt\text{-}unpred}}_{\mathcal{E},\mathcal{A}_4}\big) + 2 \cdot \mathsf{Adv}^{\mathsf{ACE\text{-}corr}}_{\mathcal{E},\mathcal{A}_5}, \\
\mathsf{Adv}^{\mathsf{ACE\text{-}no\text{-}read,priv}}_{\mathcal{E},\mathcal{B}} &\leq \mathsf{Adv}^{\mathsf{ACE\text{-}priv\text{-}CCA}}_{\mathcal{E},\mathcal{B}_1} + \mathsf{Adv}^{\mathsf{ACE\text{-}sAnon\text{-}CCA}}_{\mathcal{E},\mathcal{B}_2}, \\
\mathsf{Adv}^{\mathsf{ACE\text{-}no\text{-}read,anon}}_{\mathcal{E},\mathcal{C}} &\leq \mathsf{Adv}^{\mathsf{ACE\text{-}sAnon\text{-}CCA}}_{\mathcal{E},\mathcal{C}_1}.
\end{aligned}$$

Proof. We distinguish the above three cases.

No-write advantage. We start with the first reduction. Assume there is an adversary $\mathcal{A}$ that plays the security game $\mathsf{Exp}^{\mathsf{ACE\text{-}no\text{-}write}}_{\mathcal{E},\mathcal{A}}$. We construct the adversary $\mathcal{A}_1 = (\mathcal{A}_{1,1}, \mathcal{A}_{1,2})$ that plays the security game $\mathsf{Exp}^{\mathsf{ACE\text{-}san\text{-}CCA}}_{\mathcal{E},\mathcal{A}'}$ and relate their advantages. When invoked on input sp, $\mathcal{A}_{1,1}$ internally emulates a (black-box) execution of $\mathcal{A}$ on input sp. In particular, the only oracle queries that $\mathcal{A}$ asks are queries to generate keys and to produce sanitized ciphertexts for a given message m and role j. These are emulated by $\mathcal{A}_{1,1}$ as follows:

$\mathcal{O}_{G_i}(\cdot,\cdot)$**:** On query $(i, \mathtt{sen})$ or $(i, \mathtt{rec})$, $\mathcal{A}_{1,1}$ queries $(i, \mathtt{sen})$, respectively $(i, \mathtt{rec})$, to its own oracle $\mathcal{O}_G$ and returns the answer to $\mathcal{A}$.

$\mathcal{O}_{ES}(\cdot,\cdot)$**:** On query $\big(j, m\big)$, $\mathcal{A}_{1,1}$ queries $(j, \mathtt{sen})$ to its oracle $\mathcal{O}_G$ to receive the encryption key ek_j.[3] It then computes $c' \leftarrow \mathsf{San}(sp, \mathsf{Enc}(ek_j, m))$ and outputs c' to $\mathcal{A}$.

When $\mathcal{A}$ outputs its challenge (c_0, i', st), $\mathcal{A}_{1,1}$ chooses a uniformly random message $m \leftarrow \mathcal{M}$, queries $(i', \mathtt{sen})$ to its oracle $\mathcal{O}_G$ to receive the encryption key $ek_{i'}$. It then computes $c_1 \leftarrow \mathsf{Enc}(ek_{i'}, m)$. Finally, $\mathcal{A}_{1,1}$ outputs (c_0, c_1, st) as its challenge.

When $\mathcal{A}_{1,2}$ is invoked with the sanitized version of one of its challenge ciphertexts and the state, i.e., on input (c', st), it invokes $\mathcal{A}$ on the same input and emulates the oracles the same way as $\mathcal{A}_{1,2}$ does above. When $\mathcal{A}$ outputs its decision bit b', $\mathcal{A}_{1,2}$ outputs b' as its own guess and terminates.

We observe that the view $\mathcal{A}_1$ emulates towards $\mathcal{A}$ in the experiment $\mathsf{Exp}^{\mathsf{ACE\text{-}san\text{-}CCA}}_{\mathcal{E},\mathcal{A}_1}$ is identical to the view that $\mathcal{A}$ has in the experiment $\mathsf{Exp}^{\mathsf{ACE\text{-}no\text{-}write}}_{\mathcal{E},\mathcal{A}}$. In particular, if $b = 0$, $\mathcal{A}$ receives sanitized ciphertext which was part of the challenge, and if $b = 1$ it receives the sanitized ciphertext of an encryption to a random message. We can thus conclude that the probability of the event $b = b'$ is the same. What remains to prove is that the winning conditions required from Definition 2.4, and the emulation strategy above, lead to valid winning conditions according to Definition 4.3.

To see this, observe that the winning event for $\mathcal{A}$ is

$$\mathrm{W_{noW}} := [b = b' \ \wedge \ i' \in I_1 \cup \{0\} \ \wedge \ \forall i \in I_1 \ \forall j \in J \ P(i,j) = 0 \ \wedge \ \mathsf{San}(sp, c_0) \neq \bot]$$

and the winning event for $\mathcal{A}_1$ is

$$\mathrm{W_{san}} := [b' = b \ \wedge \ c_0' \neq \bot \neq c_1' \ \wedge \ \forall j \in J \ m_{0,j} = m_{1,j} = \bot].$$

By correctness of the assumed ACE scheme, we have $\mathsf{San}(sp, c_1) \neq \bot$ (except with probability p) in the above emulation. Furthermore, the winning conditions of Definition 4.3 do not restrict the sets of requested keys, but requires that for all decryption keys requested, the decrypted ciphertexts yield the same message. Define the event E_1 in the above emulation, if $\forall i \in I_1 \ \forall j \in J \ P(i,j) = 0 \ \wedge \ \exists j : \mathsf{Dec}\big(\mathsf{Gen}(msk, j, \mathtt{rec}), c_i'\big) \neq \bot$. We see that if event E_1 does not occur and correctness is not violated then the winning condition of $\mathcal{A}$ implies the winning condition of $\mathcal{A}_1$. Thus

$$\Pr^{\mathsf{Exp}^{\mathsf{ACE\text{-}san\text{-}CCA}}_{\mathcal{E},\mathcal{A}_1}}[\mathrm{W_{san}}] \geq \Pr^{\mathsf{Exp}^{\mathsf{ACE\text{-}no\text{-}write}}_{\mathcal{E},\mathcal{A}}}[W_{\mathrm{noW}} \ \wedge \ \neg E_1] - p.$$

[3] Looking ahead, we note that obtaining encryption keys is not problematic in the sanitization game, since the winning condition is not restricted regarding the obtained encryption keys.

Note that if we violate the correctness property with probability p, we can define a straightforward reduction to obtain another adversary such that $p \leq \mathsf{Adv}^{\mathsf{ACE\text{-}corr}}_{\mathcal{E},\mathcal{A}_5}$.

To cover the remaining cases, let us introduce a hybrid process $\mathsf{Hyb}_{\mathcal{E},\mathcal{A}}$, which is identical to $\mathsf{Exp}^{\mathsf{ACE\text{-}no\text{-}write}}_{\mathcal{E},\mathcal{A}}$, but where we introduce the additional steps. Hyb evaluates $\mathsf{DMod}(sp, \tilde{c}, c_0)$ for all ciphertexts $\tilde{c}$ which are generated by $\mathcal{O}_{ES}$ before being sanitized. We define the event D which occurs iff at least one of these evaluations yields output 1. Note that the hybrid process has the same input-output behavior as the original experiment.

We partition the probability space accordingly to design the remaining reductions.

2nd Reduction, Role-Respecting: When invoked on input sp, $\mathcal{A}_2$ internally emulates a (black-box) execution of $\mathcal{A}$ on input sp and emulates the oracles as above. When $\mathcal{A}$ outputs its challenge (c, i', st), $\mathcal{A}_2$ chooses a uniformly random message $m \leftarrow \mathcal{M}$, queries (i', m) to its oracle $\mathcal{O}_E$ to receive the corresponding ciphertext c_1. Finally, $\mathcal{A}_2$ outputs as its challenge either c_0 or c_1, each with probability one-half. The winning condition of the role-respecting game says

$$W_{RR} \coloneqq [c' \neq \bot \ \wedge \ \mathsf{dct} = \mathtt{false} \ \wedge \ \neg\big(\exists i \in I \ \forall j \in J \ (m_j \neq \bot \leftrightarrow P(i,j) = 1)\big)].$$

Observing that adversary $\mathcal{A}_2$ perfectly emulates the experiment Hyb towards $\mathcal{A}$ (where the event D of Hyb is represented by the event $\mathsf{dtc} = 1$ in the emulation), we conclude that

$$\Pr^{\mathsf{Exp}^{\mathsf{ACE\text{-}URR}}_{\mathcal{E},\mathcal{A}_2}}[W_{RR}] = \frac{1}{2} \Pr^{\mathsf{Hyb}_{\mathcal{E},\mathcal{A}}}[W_{\text{noW}} \ \wedge \ E_1 \ \wedge \neg D].$$

3rd Reduction, Sanitization: Assume there is an adversary $\mathcal{A}$ that plays the security game $\mathsf{Exp}^{\mathsf{ACE\text{-}no\text{-}write}}_{\mathcal{E},\mathcal{A}}$. We construct the adversary $\mathcal{A}_3 = (\mathcal{A}_{3,1}, \mathcal{A}_{3,2})$ against the sanitization game. When invoked on input sp, $\mathcal{A}_{3,1}$, first chooses $q_0 \leftarrow \{1 \dots q\}$ uniformly at random, sets $k \leftarrow 1$ and internally emulates a (black-box) execution of $\mathcal{A}$ on input sp. In particular, the oracle queries that $\mathcal{A}$ asks are queries to generate keys and to produce sanitized ciphertexts for a given message m and role j. These are emulated by $\mathcal{A}_{3,1}$ as follows:

$\mathcal{O}_{G_i}(\cdot,\cdot)$**:** On query $(i, \mathtt{sen})$ or $(i, \mathtt{rec})$, $\mathcal{A}_{3,1}$ queries $(i, \mathtt{sen})$, respectively $(i, \mathtt{rec})$, to its own oracle $\mathcal{O}_G$ and returns the answer to $\mathcal{A}$.

$\mathcal{O}_{ES}(\cdot,\cdot)$**:** On query $\big(j, m\big)$, if $k \neq q_0$, it queries $(j, \mathtt{sen})$ to its oracle $\mathcal{O}_G$ to receive the encryption key ek_j, then computes $c' \leftarrow \mathsf{San}(sp, \mathsf{Enc}(ek_j, m))$ and outputs c' to $\mathcal{A}$. Finally, set $k \leftarrow k + 1$.

If $k = q_0$, then $\mathcal{A}_{3,1}$ queries $(j, \mathtt{sen})$ to its oracle $\mathcal{O}_G$ to receive the encryption key ek_j. It then creates two independent encryptions of m by computing $\tilde{c'}_0 \leftarrow \mathsf{San}(sp, \mathsf{Enc}(ek_j, m))$ and $\tilde{c'}_1 \leftarrow \mathsf{San}(sp, \mathsf{Enc}(ek_j, m))$, set $k \leftarrow k + 1$, and outputs $((sp, st, k, \tilde{c}_0, \tilde{c}_0), \tilde{c}_0, \tilde{c}_1)$, where st denotes the current state of the emulation of $\mathcal{A}_1$. This ends phase 1 of the experiment.

When $\mathcal{A}_{3,2}$ is invoked with $((sp, st, k, \tilde{c}_0, \tilde{c}_1), \tilde{c}_b)$ (where b is the bit chosen by the game), then it continues executing $\mathcal{A}$ (using state st) and emulates the oracles as above. If $\mathcal{A}_1$ terminates, outputting a challenge ciphertext c_0, $\mathcal{A}_{3,2}$ evaluates $d_0 \leftarrow \mathsf{DMod}(sp, \tilde{c}_i, c_0)$ and $d_1 \leftarrow \mathsf{DMod}(sp, \tilde{c}_i, c_0)$. Given challenge c_0, let C denote the event that $\mathsf{DMod}(sp, \tilde{c}_0, \tilde{c}_1) = 1$

or $\mathsf{DMod}(sp, \tilde{c}_{1-b}, c) = 1$ and let us denote its probability by ε.[4] Let further D_{q_0} be the event that the output c_0 from adversary $\mathcal{A}$ detects with $\tilde{c}_i$ for exactly one $i \in \{0,1\}$. Assuming occurrence of D, this happens with probability at least $\frac{1}{q}$. We further see that if D_{q_0} occurs, but not C, then the adversary $\mathcal{A}_3$ can correctly distinguish $\tilde{c}_0$ and $\tilde{c}_1$. Details follow. To decide on its output bit, $\mathcal{A}_{3,2}$ proceeds as follows: if event $(E_1 \wedge D_{q_0} \wedge D \wedge \neg C)$ does not occur, output a uniform random bit b'. Else, if $d_0 = 1$, then output $b' = 0$, and if $d_1 = 1$, then output $b' = 1$. We get

$$\begin{aligned}\Pr^{\mathsf{Exp}^{\text{ACE-san-CCA}}_{\mathcal{E},\mathcal{A}_3}}[W_{\text{san}}] &= \Pr^{\mathsf{Hyb}_{\mathcal{E},\mathcal{A}}}[E_1 \ \wedge \ D_{q_0} \ \wedge \ D \ \wedge \ \neg C] \\ &\quad + \frac{1}{2}(1 - \Pr^{\mathsf{Hyb}'_{\mathcal{E},\mathcal{A}}}[E_1 \ \wedge \ D_{q_0} \ \wedge \ D \ \wedge \ \neg C]) \\ &= \frac{1}{2}\Pr^{\mathsf{Hyb}'_{\mathcal{E},\mathcal{A}}}[E_1 \ \wedge \ D_{q_0} \ \wedge \ D \ \wedge \ \neg C] + \frac{1}{2} \geq \frac{1-\varepsilon}{2q}\Pr^{\mathsf{Hyb}'_{\mathcal{E},\mathcal{A}}}[E_1 \ \wedge \ D] + \frac{1}{2}.\end{aligned}$$

Overall, we conclude

$$\begin{aligned}\Pr^{\mathsf{Exp}^{\text{ACE-no-write}}_{\mathcal{E},\mathcal{A}}}[\text{win}_{\text{noW}}] &= \Pr^{\mathsf{Hyb}_{\mathcal{E},\mathcal{A}}}[\text{win}_{\text{noW}}] \\ &= \underbrace{\Pr^{\mathsf{Hyb}_{\mathcal{E},\mathcal{A}}}[\text{win}_{\text{noW}} \ \wedge \ \neg E_1]}_{\leq \Pr^{\mathsf{Exp}^{\text{ACE-san-CCA}}_{\mathcal{E},\mathcal{A}_1}}[\text{W}_{\text{san}}]+p} + \underbrace{\Pr^{\mathsf{Hyb}_{\mathcal{E},\mathcal{A}}}[\text{win}_{\text{noW}} \ \wedge \ E_1 \ \wedge \neg D]}_{\leq 2\cdot\Pr^{\mathsf{Exp}^{\text{ACE-URR}}_{\mathcal{E},\mathcal{A}_2}}[W_{RR}]} \\ &\quad + \underbrace{\Pr^{\mathsf{Hyb}_{\mathcal{E},\mathcal{A}}}[\text{win}_{\text{noW}} \ \wedge \ E_1 \ \wedge \ D]}_{\leq 2q\cdot\left(\Pr^{\mathsf{Exp}^{\text{ACE-san-CCA}}_{\mathcal{E},\mathcal{A}_3}}[\text{W}_{\text{san}}]-\frac{1}{2}+\varepsilon\right)}.\end{aligned}$$

We observe that the occurrence of event C violates the ciphertext unpredictability requirement and it is straightforward to construct an adversary $\mathcal{A}_4$ against that game with advantage at least ε. Therefore,

$$\begin{aligned}\mathsf{Adv}^{\text{ACE-no-write}}_{\mathcal{E},\mathcal{A}} &= 2 \cdot \Pr^{\mathsf{Exp}^{\text{ACE-no-write}}_{\mathcal{E},\mathcal{A}}}[\text{win}_{\text{noW}}] - 1 \\ &\leq \mathsf{Adv}^{\text{ACE-san-CCA}}_{\mathcal{E},\mathcal{A}_1} + 2 \cdot \mathsf{Adv}^{\text{ACE-RR}}_{\mathcal{E},\mathcal{A}_2} \\ &\quad + 2 \cdot \mathsf{Adv}^{\text{ACE-corr}}_{\mathcal{E},\mathcal{A}'} + 2q \cdot (\mathsf{Adv}^{\text{ACE-san-CCA}}_{\mathcal{E},\mathcal{A}_3} + \mathsf{Adv}^{\text{ACE-ctxt-unpred}}_{\mathcal{E},\mathcal{A}_4}).\end{aligned}$$

No-read, privacy advantage. Assume there is an adversary $\mathcal{B}$ that plays the security game $\mathsf{Exp}^{\text{ACE-no-read}}_{\mathcal{E},\mathcal{B}}$. We construct the adversary $\mathcal{B}_1 = (\mathcal{B}_{1,1}, \mathcal{B}_{1,2})$ that plays the security game $\mathsf{Exp}^{\text{ACE-privAnon-CCA}}_{\mathcal{E},\mathcal{B}_1}$ and relate their advantages. When invoked on input sp, $\mathcal{B}_{1,1}$ internally emulates a (black-box) execution of $\mathcal{B}$ on input sp. In particular, the only oracle queries that $\mathcal{B}$ asks are queries to generate keys, which are emulated by $\mathcal{B}_{1,1}$ as follows:

$\mathcal{O}_G(\cdot,\cdot)$**:** On query $(i, \mathtt{sen})$ or $(i, \mathtt{rec})$, $\mathcal{B}_{1,1}$ queries $(i, \mathtt{sen})$, respectively $(i, \mathtt{rec})$, to its own oracle $\mathcal{O}_G$ and returns the answer to $\mathcal{B}$.

$\mathcal{O}_E(\cdot,\cdot)$**:** On query $\big(j, m\big)$, $\mathcal{B}_{1,1}$ queries $(j, \mathtt{sen})$ to its oracle $\mathcal{O}_G$ to receive the encryption key ek_j.[5] It then computes $c \leftarrow \mathsf{Enc}(ek_j, m)$ and outputs c to $\mathcal{B}$.

[4]This probability is bounded by the ciphertext-unpredictability advantage.

[5]Looking ahead, we note that obtaining encryption keys is not problematic in the privacy game since the winning condition is not restricted regarding the obtained encryption keys.

When $\mathcal{B}$ outputs its challenge (m_0, m_1, i_0, i_1, st), $\mathcal{B}_{1,1}$ outputs the challenge (m_0, m_1, i_0, i_0, st). When invoked (in the second phase of the experiment, $\mathcal{B}_{1,2}$ invokes $\mathcal{B}$ on input (st, c) and emulates oracle $\mathcal{O}_G$ exactly the way $\mathcal{B}_{1,1}$ did in the first phase. When adversary $\mathcal{B}$ terminates with output b, $\mathcal{B}_{1,2}$ outputs b as its own guess and terminates.

Consider the experiment $\mathsf{Exp}^{\mathsf{ACE\text{-}privAnon\text{-}CCA}}_{\mathcal{E},\mathcal{B}_1}$: we observe that

$$\Pr^{\mathsf{Exp}^{\mathsf{ACE\text{-}privAnon\text{-}CCA}}_{\mathcal{E},\mathcal{B}_1}}[b' = 0 \mid b = 0] = \Pr^{\mathsf{Exp}^{\mathsf{ACE\text{-}no\text{-}read}}_{\mathcal{E},\mathcal{B}}}[b' = 0 \mid b = 0].$$

Let us further consider a hybrid process $\mathsf{Hyb}_{\mathcal{E},\mathcal{B}}$ which is identical to $\mathsf{Exp}^{\mathsf{ACE\text{-}no\text{-}read}}_{\mathcal{E},\mathcal{B}}$ except that the output (m_0, m_1, i_0, i_1, st) of $\mathcal{B}$ gets overwritten by (m_0, m_1, i_0, i_0, st). We thus have

$$\begin{aligned}\Pr^{\mathsf{Exp}^{\mathsf{ACE\text{-}privAnon\text{-}CCA}}_{\mathcal{E},\mathcal{B}_1}}[b' = \ \mid b = 1] &= \Pr^{\mathsf{Hyb}}[b' = 1 \mid b = 1] \\ &= \Pr^{\mathsf{Exp}^{\mathsf{ACE\text{-}no\text{-}read}}_{\mathcal{E},\mathcal{B}}}[b' = 1 \mid b = 1] - \delta,\end{aligned}$$

where the difference δ can be bounded by $\mathsf{Adv}^{\mathsf{ACE\text{-}sAnon\text{-}CCA}}_{\mathcal{E},\mathcal{B}_2}$ for an adversary $\mathcal{B}_2 = (\mathcal{B}_{2,1}, \mathcal{B}_{2,2})$ against the anonymity of the ACE scheme:

When invoked on input sp, $\mathcal{B}_{2,1}$ internally emulates a (black-box) execution of $\mathcal{B}$ on input sp. In particular, the only oracle queries that $\mathcal{B}$ asks are queries to generate keys, which are emulated the same way as done by $\mathcal{B}_{1,1}$ before. When $\mathcal{B}$ outputs its challenge (m_0, m_1, i_0, i_1, st), $\mathcal{B}_{2,1}$ outputs the challenge (m_1, m_1, i_0, i_1, st). When invoked (in the second phase of the experiment, $\mathcal{B}_{2,2}$ invokes $\mathcal{B}$ on input (st, c) and emulates oracle $\mathcal{O}_G$ as before. When adversary $\mathcal{B}$ terminates with output b, $\mathcal{B}_{2,2}$ outputs b as its own guess and terminates. For this adversary, we have

$$\begin{aligned}&\Pr^{\mathsf{Exp}^{\mathsf{ACE\text{-}privAnon\text{-}CCA}}_{\mathcal{E},\mathcal{B}_2}}[b' = b] = \\ &\frac{1}{2}\Pr^{\mathsf{Exp}^{\mathsf{ACE\text{-}no\text{-}read}}_{\mathcal{E},\mathcal{B}}}[b' = 1 \mid b = 1] + \frac{1}{2}(1 - \Pr^{\mathsf{Hyb}_{\mathcal{E},\mathcal{B}}}[b' = 1 \mid b = 1]) \\ &= \frac{1}{2} + \frac{\delta}{2}.\end{aligned}$$

The proof is concluded by observing that if $\mathcal{B}$ satisfies the condition $\forall j \in J\ P(i_0, j) = P(i_1, j) = 0$, this implies the respective necessary winning conditions $\forall j \in J\ P(i_0, j) = 0$ (for the privacy adversary $\mathcal{B}_1$) and in addition $\forall j \in J\ P(i_0, j) = P(i_1, j)$ (for the anonymity adversary $\mathcal{B}_2$) as required by Definition 4.2.

No-read, anonymity advantage. Assume there is an adversary $\mathcal{C}$ that plays the security game $\mathsf{Exp}^{\mathsf{ACE\text{-}no\text{-}read}}_{\mathcal{E},\mathcal{C}}$. We construct the adversary $\mathcal{C}_1 = (\mathcal{C}_{1,1}, \mathcal{C}_{1,2})$ that plays the security game $\mathsf{Exp}^{\mathsf{ACE\text{-}privAnon\text{-}CCA}}_{\mathcal{E},\mathcal{C}_1}$ and relate their advantages. When invoked on input sp, $\mathcal{C}_{1,1}$ internally emulates a (black-box) execution of $\mathcal{C}$ on input sp. In particular, the only oracle queries that $\mathcal{C}$ asks are queries to generate keys, which are emulated by $\mathcal{C}_{1,1}$ as follows:

$\mathcal{O}_G(\cdot,\cdot)$: On query $(i, \mathtt{sen})$ or $(i, \mathtt{rec})$, $\mathcal{C}_{1,1}$ queries $(i, \mathtt{sen})$, respectively $(i, \mathtt{rec})$, to its own oracle $\mathcal{O}_G$ and returns the answer to $\mathcal{C}$.

$\mathcal{O}_E(\cdot,\cdot)$: On query (j, m), $\mathcal{C}1, 1$ queries $(j, \mathtt{sen})$ to its oracle $\mathcal{O}_G$ to receive the encryption key ek_j. It then computes $c \leftarrow \mathsf{Enc}(ek_j, m)$ and outputs c to $\mathcal{C}$.

When $\mathcal{C}$ outputs its challenge (m_0, m_1, i_0, i_1, st), $\mathcal{C}_{1,1}$ outputs the challenge (m_0, m_1, i_0, i_1, st). When invoked (in the second phase of the experiment, $\mathcal{C}_{1,2}$ invokes $\mathcal{C}$ on input (st, c) and emulates oracle $\mathcal{O}_G$ exactly the way $\mathcal{C}1, 1$ did in the first phase. When adversary $\mathcal{C}$ terminates with output b, $\mathcal{C}_{1,2}$ outputs b as its own guess and terminates.

We observe that the view $\mathcal{C}_1$ emulates towards $\mathcal{C}$ in the experiment $\mathsf{Exp}^{\text{ACE-privAnon-CCA}}_{\mathcal{E},\mathcal{C}_1}$ is identical to the view that $\mathcal{C}$ has in the experiment $\mathsf{Exp}^{\text{ACE-no-read}}_{\mathcal{E},\mathcal{C}}$. In particular, if $b = 0$, $\mathcal{C}$ receives the encryption of m_0 relative to encryption key ek_{i_0} and if $b = 1$, $\mathcal{C}$ receives the encryption of m_1 relative to encryption key ek_{i_1}. We thus conclude that

$$\begin{aligned}
&\Pr^{\mathsf{Exp}^{\text{ACE-privAnon-CCA}}_{\mathcal{E},\mathcal{C}_1}}\big[b' = b \ \wedge\ m_0 = m_1 \ \wedge\ \forall j \in J\ P(i_0, j) = P(i_1, j)\big] \\
&= \Pr^{\mathsf{Exp}^{\text{ACE-no-read}}_{\mathcal{E},\mathcal{C}}}\big[b' = b \ \wedge\ m_0 = m_1 \ \wedge\ \forall j \in J\ P(i_0, j) = P(i_1, j)\big].
\end{aligned}$$

This concludes the third case and the proof of the theorem. □

5 Enhanced Sanitizable Public-Key Encryption

5.1 Definitions

As a stepping stone toward ACE schemes satisfying our new security definitions, we introduce *enhanced sanitizable public-key encryption*. Sanitizable public-key encryption has been considered by Damgård et al. [DHO16] and Fuchsbauer et al. [FGKO17] as a relaxation of universal re-encryption [GJJS04] and rerandomizable encryption [Gro04; PR07]. It allows to *sanitize* a ciphertext to obtain a *sanitized ciphertext* that cannot be linked to the original ciphertext except that it decrypts to the correct message. In contrast to rerandomizable encryption, sanitized ciphertexts can have a different syntax than ciphertexts, i.e., it is not required that a sanitized ciphertext is indistinguishable from a fresh encryption. We introduce an enhanced variant with a different syntax and stronger security guarantees.

Definition 5.1. An *enhanced sanitizable public-key encryption (sPKE) scheme* consists of the following five PPT algorithms:

Setup: The algorithm Setup on input a security parameter 1^κ, outputs *public parameters sp*, which contain a *message space* $\mathcal{M}$ and a *ciphertext space* $\mathcal{C}$, and a secret parameter msk.

Key Generation: The algorithm Gen private parameters msk, outputs an *encryption key ek* and a *decryption key* Dec.

Encrypt: The algorithm Enc on input an encryption key ek and a message $m \in \mathcal{M}$, outputs a ciphertext $c \in \mathcal{C}$.

Sanitizer: The algorithm San on input public parameters sp and a ciphertext $c \in \mathcal{C}$, outputs a *sanitized ciphertext* $c' \in \mathcal{C}' \cup \{\perp\}$.

Decrypt: The algorithm Dec on input a decryption key dk and a sanitized ciphertext $c' \in \mathcal{C}'$, outputs a message $m \in \mathcal{M} \cup \{\perp\}$; on input dk and $\perp$, it outputs $\perp$.

For correctness, we require for all (sp, msk) in the range of Setup, all (ek, dk) in the range of $\mathsf{Gen}(msk)$, and all $m \in \mathcal{M}$ that

$$\mathsf{Dec}\big(dk, \mathsf{San}\big(sp, \mathsf{Enc}(ek, m)\big)\big) = m$$

with probability 1.

We require robustness in the sense that no ciphertext decrypts to a message different from $\perp$ for two different decryption keys (except with small probability). This is similar to detectability for ACE schemes, but we allow the adversary to directly output a ciphertext, instead of a message, which is then honestly encrypted. Our notion therefore closely resembles *unrestricted strong robustness (USROB)*, introduced by Farshim et al. [FLPQ13] for public-key encryption, which also allows the adversary to choose a ciphertext and, in contrast to strong robustness by Abdalla et al. [ABN10], gives the adversary access to decryption keys.

Definition 5.2. Let $\mathcal{E} = (\mathsf{Setup}, \mathsf{Gen}, \mathsf{Enc}, \mathsf{San}, \mathsf{Dec})$ be an sPKE scheme. For a probabilistic algorithm $\mathcal{A}$, we define the experiment $\mathsf{Exp}^{\mathsf{sPKE\text{-}USROB}}_{\mathcal{E},\mathcal{A}}$ that executes $(sp, msk) \leftarrow \mathsf{Setup}(1^\kappa)$ and $(c, i_0, i_1) \leftarrow \mathcal{A}^{\mathcal{O}_G(\cdot)}(sp)$, where the oracle $\mathcal{O}_G$ on input `getNew`, outputs a fresh key pair $(ek, dk) \leftarrow \mathsf{Gen}(msk)$. Let q be the number of oracle queries and let for $i \in \{1, \ldots, q\}$, (ek_i, dk_i) be the i-th answer from $\mathcal{O}_G$. We define the *robustness advantage* of $\mathcal{A}$ as

$$\mathsf{Adv}^{\mathsf{sPKE\text{-}USROB}}_{\mathcal{E},\mathcal{A}} := \Pr\big[1 \leq i_0, i_1 \leq q \ \wedge\ i_0 \neq i_1 \\ \wedge\ \mathsf{Dec}\big(dk_{i_0}, \mathsf{San}(sp, c)\big) \neq \perp \neq \mathsf{Dec}\big(dk_{i_1}, \mathsf{San}(sp, c)\big)\big],$$

where the probability is over the randomness in $\mathsf{Exp}^{\mathsf{sPKE\text{-}USROB}}_{\mathcal{E},\mathcal{A}}$ and the random coins of San and Dec (both executed independently twice).

We next define IND-CCA security analogous to the definition for ordinary public-key encryption. In contrast to the usual definition, we do not require the adversary to output two message of equal length, which implies that schemes satisfying our definition do not leak the length of the encrypted message.

Definition 5.3. Let $\mathcal{E} = (\mathsf{Setup}, \mathsf{Gen}, \mathsf{Enc}, \mathsf{San}, \mathsf{Dec})$ be an sPKE scheme and let $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ be a pair of probabilistic algorithms. Consider the experiment $\mathsf{Exp}^{\mathsf{sPKE\text{-}IND\text{-}CCA}}_{\mathcal{E},\mathcal{A}}$ in Figure 3 and let $C_{\mathcal{A}_2}$ be the set of all ciphertexts that $\mathcal{A}_2$ queried to the oracle $\mathcal{O}_{SD}$. We define the *ciphertext indistinguishability under chosen-ciphertext attacks advantage* of $\mathcal{A}$ as

$$\mathsf{Adv}^{\mathsf{sPKE\text{-}IND\text{-}CCA}}_{\mathcal{E},\mathcal{A}} := 2 \cdot \Pr\big[b' = b \ \wedge\ c^* \notin C_{\mathcal{A}_2}\big] - 1,$$

where the probability is over the randomness in $\mathsf{Exp}^{\mathsf{sPKE\text{-}IND\text{-}CCA}}_{\mathcal{E},\mathcal{A}}$.

We also need that it is hard to predict a ciphertext generated by Enc from a message of the adversary's choice given encryption and decryption keys. Note that this is not implied by IND-CCA security since the adversary obtains the decryption key.

Definition 5.4. Let $\mathcal{E} = (\mathsf{Setup}, \mathsf{Gen}, \mathsf{Enc}, \mathsf{San}, \mathsf{Dec})$ be an sPKE scheme and let $\mathcal{A}$ be a probabilistic algorithm. Consider the experiment $\mathsf{Exp}^{\mathsf{sPKE\text{-}ctxt\text{-}unpred}}_{\mathcal{E},\mathcal{A}}$ in Figure 3. We define the *ciphertext unpredictability advantage* of $\mathcal{A}$ as

$$\mathsf{Adv}^{\mathsf{sPKE\text{-}ctxt\text{-}unpred}}_{\mathcal{E},\mathcal{A}} := \Pr\big[c = c^*\big],$$

where the probability is over the randomness in $\mathsf{Exp}^{\mathsf{sPKE\text{-}ctxt\text{-}unpred}}_{\mathcal{E},\mathcal{A}}$.

We further define anonymity or indistinguishability of keys following Bellare et al. [BBDP01].

Experiment $\mathsf{Exp}^{\mathsf{sPKE\text{-}IND\text{-}CCA}}_{\mathcal{E},\mathcal{A}}$

Input: 1^κ

$(sp, msk) \leftarrow \mathsf{Setup}(1^\kappa)$
$(ek, dk) \leftarrow \mathsf{Gen}(msk)$
$(m_0, m_1, st) \leftarrow \mathcal{A}_1^{\mathcal{O}_G(\cdot),\mathcal{O}_{SD}(\cdot)}(sp, ek)$
$b \twoheadleftarrow \{0,1\}$
$c^* \leftarrow \mathsf{Enc}(ek, m_b)$
$b' \leftarrow \mathcal{A}_2^{\mathcal{O}_G(\cdot),\mathcal{O}_{SD}(\cdot)}(st, c^*)$

Experiment $\mathsf{Adv}^{\mathsf{sPKE\text{-}ctxt\text{-}unpred}}_{\mathcal{E},\mathcal{A}}$

Input: 1^κ

$(sp, msk) \leftarrow \mathsf{Setup}(1^\kappa)$
$(ek, dk) \leftarrow \mathsf{Gen}(msk)$
$(m, c) \leftarrow \mathcal{A}^{\mathcal{O}_G(\cdot)}(sp, ek, dk)$
$c^* \leftarrow \mathsf{Enc}(ek, m)$

Experiment $\mathsf{Exp}^{\mathsf{sPKE\text{-}IK\text{-}CCA}}_{\mathcal{E},\mathcal{A}}$

Input: 1^κ

$(sp, msk) \leftarrow \mathsf{Setup}(1^\kappa)$
$(ek_0, dk_0) \leftarrow \mathsf{Gen}(msk)$
$(ek_1, dk_1) \leftarrow \mathsf{Gen}(msk)$
$(m, st) \leftarrow \mathcal{A}_1^{\mathcal{O}_G(\cdot),\mathcal{O}_{SD_0}(\cdot),\mathcal{O}_{SD_1}(\cdot)}(sp, ek_0, ek_1)$
$b \twoheadleftarrow \{0,1\}$
$c^* \leftarrow \mathsf{Enc}(ek_b, m)$
$b' \leftarrow \mathcal{A}_2^{\mathcal{O}_G(\cdot),\mathcal{O}_{SD_0}(\cdot),\mathcal{O}_{SD_1}(\cdot)}(st, c^*)$

Experiment $\mathsf{Exp}^{\mathsf{sPKE\text{-}san\text{-}CCA}}_{\mathcal{E},\mathcal{A}}$

Input: 1^κ

$(sp, msk) \leftarrow \mathsf{Setup}(1^\kappa)$
$(ek_0, dk_0) \leftarrow \mathsf{Gen}(msk)$
$(ek_1, dk_1) \leftarrow \mathsf{Gen}(msk)$
$(c_0, c_1, st) \leftarrow \mathcal{A}_1^{\mathcal{O}_G(\cdot),\mathcal{O}_{SD_0}(\cdot),\mathcal{O}_{SD_1}(\cdot)}(sp, ek_0, ek_1)$
$c'_0 \leftarrow \mathsf{San}(sp, c_0)$; $c'_1 \leftarrow \mathsf{San}(sp, c_1)$
$m_{0,0} \leftarrow \mathsf{Dec}(dk_0, c'_0)$; $m_{0,1} \leftarrow \mathsf{Dec}(dk_1, c'_0)$
$m_{1,0} \leftarrow \mathsf{Dec}(dk_0, c'_1)$; $m_{1,1} \leftarrow \mathsf{Dec}(dk_1, c'_1)$
$b \twoheadleftarrow \{0,1\}$
$b' \leftarrow \mathcal{A}_2^{\mathcal{O}_G(\cdot),\mathcal{O}_{SD_0}(\cdot),\mathcal{O}_{SD_1}(\cdot)}(st, c'_b)$

Figure 3: Security experiments for an sPKE scheme $\mathcal{E}$ and an adversary $\mathcal{A}$, where $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ in the experiments $\mathsf{Exp}^{\mathsf{sPKE\text{-}IND\text{-}CCA}}_{\mathcal{E},\mathcal{A}}$, $\mathsf{Exp}^{\mathsf{sPKE\text{-}IK\text{-}CCA}}_{\mathcal{E},\mathcal{A}}$, and $\mathsf{Exp}^{\mathsf{sPKE\text{-}san\text{-}CCA}}_{\mathcal{E},\mathcal{A}}$. The oracle $\mathcal{O}_{SD}$ is defined as $\mathcal{O}_{SD}(c) = \mathsf{Dec}(dk, \mathsf{San}(sp, c))$ and the oracle $\mathcal{O}_{SD_j}$ as $\mathcal{O}_{SD_j}(c) = \mathsf{Dec}(dk_j, \mathsf{San}(sp, c))$. Moreover, the oracle $\mathcal{O}_G$ on input `getNew`, outputs a fresh key pair $(ek, dk) \leftarrow \mathsf{Gen}(msk)$.

Definition 5.5. Let $\mathcal{E} = (\mathsf{Setup}, \mathsf{Gen}, \mathsf{Enc}, \mathsf{San}, \mathsf{Dec})$ be an sPKE scheme and let $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ be a pair of probabilistic algorithms. Consider the experiment $\mathsf{Exp}^{\mathsf{sPKE\text{-}IK\text{-}CCA}}_{\mathcal{E},\mathcal{A}}$ in Figure 3 and let $C_{\mathcal{A}_2}$ be the set of all ciphertexts that $\mathcal{A}_2$ queried to the oracle $\mathcal{O}_{SD_0}$ or $\mathcal{O}_{SD_1}$. We define the *indistinguishability of keys under chosen-ciphertext attacks advantage* of $\mathcal{A}$ as

$$\mathsf{Adv}^{\mathsf{sPKE\text{-}IK\text{-}CCA}}_{\mathcal{E},\mathcal{A}} := 2 \cdot \Pr\big[b' = b \ \wedge \ c^* \notin C_{\mathcal{A}_2}\big] - 1,$$

where the probability is over the randomness in $\mathsf{Exp}^{\mathsf{sPKE\text{-}IK\text{-}CCA}}_{\mathcal{E},\mathcal{A}}$.

Sanitization security formalizes that given certain public keys and a sanitized ciphertext, it is hard to tell which of two adversarially chosen ciphertexts was actually sanitized. To exclude trivial attacks, we require that both ciphertexts decrypt are encryptions relative to the two challenge public keys ek_0 and ek_1.

Definition 5.6. Let $\mathcal{E} = (\mathsf{Setup}, \mathsf{Gen}, \mathsf{Enc}, \mathsf{San}, \mathsf{Dec})$ be an sPKE scheme and let $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ be a pair of probabilistic algorithms. Consider the experiment $\mathsf{Exp}^{\mathsf{sPKE\text{-}san\text{-}CCA}}_{\mathcal{E},\mathcal{A}}$ in Figure 3. We define the *sanitization under chosen-ciphertext attacks advantage* of $\mathcal{A}$ as

$$\mathsf{Adv}^{\mathsf{sPKE\text{-}san\text{-}CCA}}_{\mathcal{E},\mathcal{A}} := 2 \cdot \Pr\big[b' = b \ \wedge \ \exists j, j' \in \{0,1\} \ m_{0,j} \neq \bot \neq m_{1,j'}\big] - 1,$$

where the probability is over the randomness in $\mathsf{Exp}^{\mathsf{sPKE\text{-}IK\text{-}CCA}}_{\mathcal{E},\mathcal{A}}$.

We finally define the probability that two independent executions of the key-generation algorithm produce the same encryption key. This probability has to be small for all IND-CCA-secure schemes because an attacker can otherwise generate a new key pair himself and if the obtained encryption key matches the one with which the challenge ciphertext is generated, the attacker can decrypt and win the IND-CCA game. We anyway explicitly define this probability to simplify our reductions later.

Definition 5.7. Let $\mathcal{E} = (\mathsf{Setup}, \mathsf{Gen}, \mathsf{Enc}, \mathsf{San}, \mathsf{Dec})$ be an sPKE scheme. We define the *encryption-key collision probability* $\mathsf{Col}^{\mathsf{ek}}_{\mathcal{E}}$ as the maximum over all (sp, msk) in the range of $\mathsf{Setup}(1^\kappa)$ of

$$\Pr^{(ek_0,dk_0)\leftarrow \mathsf{Gen}(msk);\ (ek_1,dk_1)\leftarrow \mathsf{Gen}(msk)}[ek_0 = ek_1].$$

5.2 Constructing an sPKE Scheme

We next construct an sPKE scheme satisfying our security definitions. Our construction resembles the weakly sanitizable PKE scheme by Fuchsbauer et al. [FGKO17]. We obtain security against chosen-ciphertext attacks using the technique of Naor and Yung [NY90], i.e., encrypting the message under two independent keys and proving in zero-knowledge that the ciphertexts are encryptions of the same message, which was shown to achieve full IND-CCA security if the zero-knowledge proof is simulation-sound by Sahai [Sah99].

Setup: The setup algorithm first generates a key pair $(ek^{\mathsf{PKE}}, dk^{\mathsf{PKE}}) \leftarrow \mathsf{PKE.Gen}$ of a (IND-CPA secure) public-key encryption scheme, and a key pair $(vk^{\mathsf{Sig}}, sk^{\mathsf{Sig}}) \leftarrow \mathsf{Sig.Gen}$ of a (EU-CMA-secure) signature scheme. Additionally, it samples a uniformly random common-reference string for the non-interactive zero-knowledge proof system NIZK for language $L = \{x \mid \exists w : (x, w) \in R\}$, where the relation R is defined as follows: for $x = (c_1, c_2, c_\sigma)$ and $w = (m, g^x, g^y; r_1, s_1, r_2, s_2; \sigma, r)$, $R(x, w) = 1$ if and only if

$$c_1 = (g^{r_1}, g^{x\cdot r_1}, g^{s_1}, g^{x\cdot s_1}\cdot m) \ \wedge\ c_2 = (g^{r_2}, g^{y\cdot r_2}, g^{s_2}, g^{y\cdot s_2}\cdot m)\}$$
$$\wedge\ \mathsf{Ver}(vk^{\mathsf{Sig}}, (g^x, g^y), \sigma) = 1 \ \wedge\ c_\sigma = \mathsf{PKE.Enc}(ek^{\mathsf{PKE}}, (g^x, g^y, \sigma); r).$$

The *public parameters* sp contains a description of a finite cyclic group G with prime order, a generator g (i.e., $G = \langle g\rangle$), the order q of G, the message space $\mathcal{M} \subset G$ of size n, and the verification key vk^{Sig}, the public key ek^{PKE}, and the CRS crs. We assume that $q > 2^\kappa$, and that $n/q \le 2^{-\kappa}$,

The *private parameters* msk consist of the signing key sk^{Sig} and a decryption key dk^{PKE} and the public parameter.

Key Generation: The algorithm Gen on input sp, and msk, samples two elements $dk_1, dk_2 \in \mathbb{Z}_q$ and computes $ek_1 := g^{dk_1}$, $ek_2 := g^{dk_2}$, as well as $\sigma \leftarrow \mathsf{Sig.Sign}(sk^{\mathsf{Sig}}, (ek_1, ek_2))$. Finally, it outputs $ek := (ek_1, ek_2, \sigma)$ and $dk := (dk_1, dk_2)$.

Encrypt: The algorithm Enc on input an encryption key ek and a message $m \in \mathcal{M}$, computes the following: choose randomness (r_1, s_1) and (r_2, s_2) (each component from set $\mathbb{Z}_q^*$) and compute two ElGamal ciphertexts

$$c_1 := (g^{r_1}, ek_1^{r_1}, g^{s_1}, ek_1^{s_1}\cdot m),$$
$$c_2 := (g^{r_2}, ek_2^{r_2}, g^{s_2}, ek_2^{s_2}\cdot m),$$

and the ciphertext $c_\sigma := \mathsf{PKE.Enc}(ek^{\mathsf{PKE}}, (ek_1, ek_2, \sigma); r)$, and finally $\pi := \mathsf{NIZK.Prove}(crs, x = (c_1, c_2, c_\sigma), w = (m, ek_1, ek_2; r_1, s_1, r_2, s_2; \sigma, r)$. The output of the encryption function is the ciphertext $c := (c_1, c_2, c_\sigma, \pi)$.

Sanitizer: The algorithm San on input public parameters sp and a ciphertext $c \in \mathcal{C}$, computes a *sanitized ciphertext* c' as follows: First verify the NIZK proofs by evaluating $f := \mathsf{Ver}(crs, x = (c_1, c_2, c_\sigma), \pi)$. If $f = 1$, then sanitize the ElGamal ciphertext $c_1 = (a, b, c, d) = (g^{r_1}, ek_1^{r_1}, g^{s_1}, ek_1^{s_1} \cdot m)$ as follows: if $a \neq 1 \neq b$, choose a random $t \in \mathbb{Z}_q^*$ and output the following value:

$$c' := (a^t \cdot c,\, b^t \cdot d) = (g^{r_1 \cdot t + s_1},\, ek_1^{r_1 \cdot t + s_1} \cdot m).$$

In case $f = 0$ or $a = 1$ or $b = 1$, then output $\perp$.

Decrypt: The algorithm Dec on input a decryption key dk and a sanitized ciphertext $c' = (a, b)$, computes the message $m := b \cdot (a^{dk_1})^{-1}$. It outputs m if $m \in \mathcal{M}$, and otherwise it outputs $\perp$; on input dk and $\perp$, it outputs $\perp$.

The main result of this section is the security of the scheme, summarized as follows.

Theorem 5.8 (Informal)**.** *The above sPKE scheme is secure, i.e., all efficient adversaries have only negligible advantage in breaking the privacy, anonymity, or sanitization property, if the DDH problem is hard in group G, the underlying encryption scheme is CPA secure, the signature scheme is unforgeable, and if the proof system is correct and provides simulation-soundness and zero-knowledge. The scheme is further correct and robust, has unpredictable ciphertexts, and a negligible encryption-key collision-probability.*

5.3 Security Proof

5.3.1 Proof Intuition

The basic motivation behind our scheme is the same as for the original idea to lift CPA security to CCA security. The general idea is to preserve the desirable properties that (this particular version of) ElGamal encryption has in a "CPA" world. More technically, we would like to reduce the required properties in Definitions 5.3 to 5.6 to the respective properties of ElGamal that hold in a world where no decryption oracle is available. The proof intuition (for example for the anonymity or privacy game) is closely related to the standard result in [Sah99]: the basic idea of the above construction is that it is actually enough to decrypt a ciphertext with one decryption key. Since the NIZK proof can be verified by anyone, and since it assures that both are valid ElGamal encryption to the same message, it does not matter which of the two ciphertexts (corresponding decryption key) is used to decrypt. In a reduction, where we assume an adversary $\mathcal{A}$ against the desired properties stated above and would like to attack the corresponding CPA properties of ElGamal, we need only get one public key and no decryption oracle. In order to emulate the view towards $\mathcal{A}$, the reduction chooses an additional public key and a CRS for the NIZK scheme. Since the reduction thus knows one of the secret keys, it can now emulate a decryption oracle. A subtle point here is that the verification can be done without knowing which public keys are used - to resolve this, the key generation process signs valid key pairs, and the verifier only needs to know that the key pair (which is not revealed to the verifier) was signed by the key generation process.

To generate a challenge ciphertext, the reduction will obtain one challenge ciphertext from its CPA-Game, and encrypt another, arbitrary message to obtain a second ciphertext. The reduction uses the NIZK simulator to obtain an accepting proof that is indistinguishable from a "real proof", even if the underlying statement is not true. A crucial point here is that the NIZK scheme has to be what is known as *1-proof simulation sound*. Even if the adversary sees one simulated (accepting) proof, even of a wrong statement, it is still not capable of producing an accepting proofs of wrong statements (except by reproducing the exact proof it obtained within the challenge, but which $\mathcal{A}$ is not allow to ask by the CCA definition). The fundamental result of [Sah99] is that the above strategy successfully simulates a complete CCA attack towards $\mathcal{A}$. While the intuition seems to match for our versions of IK-CCA and IND-CCA, it is unclear for the sanitization game and hence proven first.

5.3.2 Correctness, Robustness, Unpredictability, and Key Collision

The correctness of the scheme follows directly from the correctness of the underlying cryptographic primitives. Since we base our scheme on the ElGamal encryption scheme (and consists of two independent encryptions of which each contains two randomly chosen group elements), we have $\mathsf{Adv}^{\mathsf{sPKE\text{-}ctxt\text{-}unpred}}_{\mathcal{E},\mathcal{A}} \leq 1/q^4$ (where $q > 2^{-\kappa}$ is the order of the group). Also, since the encryption key contains a pair of random group elements, we also have that $\mathsf{Col}^{\mathsf{ek}}_{\mathcal{E}} \leq 1/q^2$ (for group order $q > 2^{-\kappa}$).

Finally, for robustness, let us denote the generated keys of the experiment by $ek_i = (ek_{i,1}, ek_{i,1})$ and $dk_i = (dk_{i,1}, dk_{i,1})$ (recall that each key consists of two parts for the respective ElGamal encryption). We observe that as long as $ek_{i,1} \neq ek_{j,1}$ for all generated keys $i \neq j$ in the experiment $\mathsf{Exp}^{\mathsf{sPKE\text{-}USROB}}_{\mathcal{E},\mathcal{A}}$ implies that $dk_{i,1} \neq dk_{j,1}$ for the corresponding decryption keys. Assume the adversary outputs the two indices i_0 and i_1 and some arbitrary ciphertext $\tilde{c} := (c_1, c_2, c_\sigma, \pi)$, let $c_1 = (g^a, g^b, g^c, g^d)$, and assume it is sanitized, i.e., $a \neq 0$ and $b \neq 0$, then this yields

$$c' := (g^{at+c},\, g^{bt+d}).$$

and hence is decrypted to $\bar{m}_0 := g^{bt+d-dk_{i_0,1}(at+c)})$. Similarly, $\bar{m}_1 := g^{bt'+d-dk_{i_1,1}(at'+c)})$ would be the result of a second sanitization followed by a decryption using $dk_{i_1,1}$. Thus, the fraction $\frac{\bar{m}_0}{\bar{m}_1} = g^{t(b-a\cdot dk_{i_0,1})+t'(a\cdot dk_{i_1,1}-b)+c(dk_{i_1,1}-dk_{i_0,1})}$ is a random group element, because t and t' are chosen at random and not both depending terms can be canceled (since computation in the exponent is done in the field $\mathbb{Z}_q$ and the decryption keys are different under our assumption above). Hence, for all adversaries that obtain at most m key pairs in experiment $\mathsf{Exp}^{\mathsf{sPKE\text{-}USROB}}_{\mathcal{E},\mathcal{A}}$ we have that $\mathsf{Adv}^{\mathsf{sPKE\text{-}USROB}}_{\mathcal{E},\mathcal{A}} \leq (m^2+1)\cdot 2^{-\kappa}$ due to the collision probability on the respective first components $ek_{i,1}$ and $dk_{i,1}$ of the obtained keys, and by the restriction on the size of the message space $\mathcal{M}$ relative to the size of the group G. Taking the union bound yields the result.

5.3.3 Sanitization

To clarify the syntax, note that an adversary in the experiment $\mathsf{Exp}^{\mathsf{sPKE\text{-}san\text{-}CCA}}_{\mathcal{E},\mathcal{A}}$ obtains two independent encryption keys of the scheme. This means, in our particular case of the above ElGamal scheme, that the adversary obtains, for $i \in \{0,1\}$, $ek_i = (ek_{i,1}, ek_{i,2}, \sigma_i)$, where $ek_{i,j}$ are (independent) instances of ElGamal public keys as defined above.

Let us define the following hybrid experiments $\mathsf{Exp}^0_{\mathcal{A}}$ to $\mathsf{Exp}^4_{\mathcal{A}}$: they behave as the experiment $\mathsf{Exp}^{\mathsf{sPKE\text{-}san\text{-}CCA}}_{\mathcal{E},\mathcal{A}}$, except as follows: after the values $m_{i,j}$ have been calculated, then

- $\mathsf{Exp}^0_{\mathcal{A}}$ is the real experiment.

- $\mathsf{Exp}^1_{\mathcal{A}}$, if $m_{0,0} = \perp$, replaces c'_0 by two uniformly random group elements (g^b, g^c).
- $\mathsf{Exp}^2_{\mathcal{A}}$, does the above replacement, but if $m_{0,0} = \perp$, replaces c'_0 by two uniformly at random chosen group elements (g^b, g^c) if $m_{0,1} \neq \perp$.
- $\mathsf{Exp}^3_{\mathcal{A}}$ does the above replacements, and in addition replaces c'_1 by two uniformly at random chosen group elements (g^b, g^c) if $m_{1,0} \neq \perp$.
- $\mathsf{Exp}^4_{\mathcal{A}}$ does the above replacement, but if $m_{1,0} = \perp$, it replaces c'_1 by two uniformly at random chosen group elements (g^b, g^c) if $m_{1,1} \neq \perp$.

We observe that $\Pr^{\mathsf{Exp}^4_{\mathcal{A}}}[W_{\text{san}}] \leq \frac{1}{2}$ for all adversaries, since the outputs are independent of bit b chosen by the experiment.

Lemma 5.9. *Let $\mathcal{A}$ be an adversary in the experiment $\mathsf{Exp}^i_{\mathcal{A}}$. Let BAD_1 be the event that $\mathcal{A}$ queries at least one of its decryption oracles $\mathcal{O}_{SD_i}$ with a valid but improper ciphertext (c_1, c_2, π), i.e., $(c_1, c_2, c_\sigma) \notin L$, but where π is an accepting proof, i.e., $\mathsf{NIZK.Ver}(crs, (c_1, c_2, c_\sigma), \pi) = 1)$. We construct an adversary $\mathcal{A}'$ such that $\Pr^{\mathsf{Exp}^i_{\mathcal{A}}}[\mathsf{BAD}_1] \leq \mathsf{Adv}^{\mathsf{NIZK\text{-}snd}}_{\mathsf{NIZK},\mathcal{A}'}$.*

Proof. The claim follows from the soundness property of the assumed NIZK scheme: upon receiving a CRS crs' (from its own challenger), $\mathcal{A}'$ defines $crs \leftarrow crs'$ and emulates towards $\mathcal{A}$ all further steps of his experiment $\mathsf{Exp}^i_{\mathcal{A}}$. This is in particular possible when possessing all secret keys. Whenever $\mathcal{A}$ submits a query to a decryption oracle, $\mathcal{A}'$ verifies the NIZK proof, and additionally checks that decryption is possible relative to the specified keys and that the signature is valid (these values are part of c_σ which the $\mathcal{A}'$ is able to decrypt[6]). If any check fails, i.e., if BAD_1 occurs, then it holds that a valid forgery against the challenged NIZK scheme (with CRS crs) occurred and that $\mathcal{A}'$ can present this forgery to its challenger. □

Lemma 5.10. *Let $\mathcal{A}$ be an adversary in the experiment $\mathsf{Exp}^i_{\mathcal{A}}$. Let BAD_2 be the event that $\mathcal{A}$ queries at least one of its decryption oracles $\mathcal{O}_{SD_i}$ with a valid and proper ciphertext $(c_1, c_2, c_\sigma, \pi)$ (i.e., $(c_1, c_2, c_\sigma) \in L$ and π is accepting), but where c_σ is the encryption of a triple (ek_1, ek_2, σ), such that the pair (ek_1, ek_2) has never been output by the experiment or the oracle $\mathcal{O}_G$. We construct an adversary $\mathcal{A}'$ such that $\Pr^{\mathsf{Exp}^i_{\mathcal{A}}}[\mathsf{BAD}_2] \leq \mathsf{Adv}^{\mathsf{Sig\text{-}EUF\text{-}CMA}}_{\mathsf{Sig},\mathcal{A}'}$*

Proof. We construct an adversary $\mathcal{A}'$ which produces a valid forgery in the signature experiment $\mathsf{Exp}^{\mathsf{Sig\text{-}EUF\text{-}CMA}}_{\mathsf{Sig},\mathcal{A}'}$ as follows: $\mathcal{A}'$ receives the signature verification key vk^{Sig} from the experiment. It sets up the experiment $\mathsf{Exp}^i_{\mathcal{A}}$, where it generates all necessary keys, except for the signature key pair, where it chooses defines vk^{Sig} as the verification key. Upon queries to the the key generation oracles, $\mathcal{A}'$ selects the key pair (ek_1, ek_2) as $\mathsf{Exp}^{S,j}_{\mathcal{A}}$, but lets it sign using its own oracle $\mathsf{Sign}(sk, \cdot)$. The rest of the emulation, in particular the decryption oracle, can be done with the knowledge of the remaining keys (no signing key is required). Furthermore, whenever $\mathcal{A}$ submits a valid query to the decryption oracle, $\mathcal{A}'$ decrypts c_σ to extract a pair (ek'_1, ek'_1) and a signature σ', and if the pair has never been queried to its own signing oracle, but the signature is valid, then it outputs $((ek'_1, ek'_1), \sigma')$ as its forgery. □

Recall that the winning condition of an assumed adversary $\mathcal{A}$ in the sanitization game is

$$W_{san} \coloneqq [b' = b \ \wedge \ \exists j, j' \in \{0,1\} \ m_{0,j} \neq \perp \neq m_{1,j'}].$$

[6]Note that we assume that the underlying PKE scheme has no decryption error.

We further observe, that we can construct from an adversary $\mathcal{A}$ against hybrid $\mathsf{Exp}^5_{\mathcal{A}}$, an adversary with success probability at least $\frac{1}{2} - (q^2+4)\cdot 2^{-\kappa}$, where q is an upper bound on the number of key pairs obtained by $\mathcal{A}$. To see this, let $\mathcal{A}'$ be the adversary that runs $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ as subroutine and relays back and forth oracle queries and answers to and from $\mathcal{A}$. When $\mathcal{A}_1$ outputs its challenge (c_0, c_1), $\mathcal{A}'$ asks both of its oracles to decrypt to compute $m_{i,j}$. If the winning condition W_{san} is not already violated, then $\mathcal{A}'$ outputs the received challenge as its own challenge. Otherwise, $\mathcal{A}'$ encrypts twice the same message using one of the public key, say ek_0 and outputs a uniform random bit. The proof follows by observing that all outputs of this last hybrid are independent of b, and that the probability of a wrong prediction is upper bounded by the probability that the first part of a public key collides with the first part of another public key (and hence robustness says that the outcome is $\perp$ except with probability $2^{-\kappa}$) and taking the union bound.

Lemma 5.11. *For any adversary $\mathcal{A}$, we have that*

$$\Pr^{\mathsf{Exp}^{\mathsf{sPKE\text{-}IK\text{-}CCA}}_{\mathcal{E},\mathcal{A}}}[W_{san}] = \underbrace{\Pr^{\mathsf{Exp}^5_{\mathcal{A}}}[W_{san}]}_{\geq\ \frac{1}{2}-(q^2+4)\cdot 2^{-\kappa}} + \sum_{i=0}^{3} \underbrace{\Pr^{\mathsf{Exp}^i_{\mathcal{A}}}[W_{san}] - \Pr^{\mathsf{Exp}^{i+1}_{\mathcal{A}}}[W_{san}]}_{\leq\ \mathsf{Adv}^{\mathsf{DDH}}_{\mathcal{B}_i} +\ p_{\mathsf{fail}}}.$$

where algorithms $\mathcal{B}_i$ are described in the proof below.

Proof. Let c_0 and c_1 be the challenge ciphertexts of $\mathcal{A}_1$. Each c_i in particular contains two candidate ElGamal encryptions. Recall the sanitizer first checks the validity of the proof and, if successful, only the first ElGamal ciphertext is processed.

Let us denote by c_0^1 and c_1^1, the respective first ElGamal encryption.

$$\begin{aligned} c_0^1 &= (d_0, d_1, e_0, e_1), \\ c_1^1 &= (d_0', d_1', e_0', e_1'). \end{aligned}$$

If $\neg\mathsf{BAD}_1$, then each c_i consists of two valid ElGamal encryptions to the same message m_0 and m_1, respectively. We are thus guaranteed in this case that there exist values r_i and s_i such that

$$\begin{aligned} c_0^1 &= (g^{r_0}, g^{r_0\cdot dk'_{0,1}}, g^{s_0}, g^{s_0\cdot dk'_{0,1}} m_0), \\ c_1^1 &= (g^{r_1}, g^{r_1\cdot dk'_{1,1}}, g^{s_1}, g^{s_1\cdot dk'_{1,1}} m_1). \end{aligned}$$

If $\neg\mathsf{BAD}_2$, then both keys $g^{dk'_{i,1}}$ have been generated either via a call to the key generation oracle, or they correspond to the respective challenge keys output by the experiment. Let us denote the set of such generated keys by $\mathcal{K}$.

Additionally, and without loss of generality, we assume that $d_i \neq \perp \neq d_i'$ for all $i \in \{0,1\}$ as otherwise, the sanitization algorithm return $\perp$ and the game cannot be won (since decryption of $\perp$ yields $\perp$). We now show that, given the above conditions, that the difference between $\Pr^{\mathsf{Exp}^{i+1}_{\mathcal{A}}}[W_{\mathrm{san}}]$ and $\Pr^{\mathsf{Exp}^{i}_{\mathcal{A}}}[W_{\mathrm{san}}]$ is a lower bound on the advantage of a DDH adversary $\mathcal{B}_0$ in distinguishing triples of the form (g^a, g^b, g^c) and (g^a, g^b, g^{ab}) for uniformly at random chosen exponents a, b, c.

We discuss the case $i = 0$, the other cases are similar: The adversary $\mathcal{B}_0$ is defined as follows: on input at DDH triple $(g^a, g^b, g^{c'})$, where c' is either the product of a and b or a uniformly random exponent, $\mathcal{B}_0$ defines $ek_{0,0} \coloneqq g^a$. All remaining keys are generated by $\mathcal{B}_0$ (and thus, only the decryption key of $ek_{0,0}$ is actually missing). It then emulates the experiment $\mathsf{Exp}^0_{\mathcal{A}}$ (by internally running $\mathcal{A}$ and performing the operations of the experiment). The only crucial point is how to emulate the decryption oracle $\mathcal{O}_{DS_0}$ towards $\mathcal{A}$: $\mathcal{B}_0$ verifies the NIZK proof π of each queried ciphertext, and, if valid, sanitizes and decrypts the second ElGamal encryption c_0^2 of the received ciphertext using $dk_{0,2}$ (here is where we use the Sahai-trick). The remaining oracles are straightforward to emulate.

When $\mathcal{A}$ outputs its challenge (c_0, c_1), both are processed as in experiment Exp, except that in order to sanitize c_0, $\mathcal{B}_0$ first computes $m_{i,j}$ as the experiment does, and, if $m_{0,0} \neq \perp$, and defines $c_0' := g^b, g^{c'} \cdot m_{0,0}$. Finally, when $\mathcal{A}$ terminates, $\mathcal{B}_0$ outputs 1 if condition W_{san} occurs (i.e, if the guess b' of $\mathcal{A}$ is equal to the emulated bit b of the experiment) and 0 otherwise.

We state the sufficient conditions under which the emulation of oracle $\mathcal{O}_{SD_0}$ is perfect: assume $\neg\mathsf{BAD}_1$ and $\neg\mathsf{BAD}_2$ hold and that none of the first parts $ek_{k,1}$ of all keys generated by the experiment or the oracles collide. Then, it holds that decrypting c_0^1 with (unknown) decryption key a would yield the same result as decrypting c_0^2 with $dk_{0,2}$, except with probability $2 \cdot 2^{-\kappa}$. In particular, since any valid ciphertext is proper and keys are unique under the above assumption, if c_0^2 is an encryption relative to public key $g^{dk_{0,2}}$, then also c_1^0 decrypts under a with probability 1 due to correctness. On the contrary, due to the robustness of the scheme in this case, if c_0^2 is not an encryption relative to $g^{dk_{0,2}}$ then c_0^1 is not an encryption relative to g^a, and thus, except with probability $1 - 2 \cdot 2^{-\kappa}$ (by robustness), both ciphertexts do not decrypt in this case. By taking the union bound over all these undesirable events, we see that the probability of an incorrect emulation is bounded by

$$\begin{aligned} p_{\mathsf{fail}} &:= 2 \cdot q^2 \cdot 2^{-\kappa} + \Pr^{\mathsf{Exp}^i_{\mathcal{A}}}[\mathsf{BAD}_1] + \Pr^{\mathsf{Exp}^i_{\mathcal{A}}}[\mathsf{BAD}_2] \\ &\leq 2 \cdot q^2 \cdot 2^{-\kappa} + \mathsf{Adv}^{\mathsf{NIZK\text{-}snd}}_{\mathsf{NIZK},\mathcal{A}'} + \mathsf{Adv}^{\mathsf{Sig\text{-}EUF\text{-}CMA}}_{\mathsf{Sig},\mathcal{A}''}, \end{aligned}$$

where q is an upper bound on the number of queries that $\mathcal{A}$ asks to his oracles.

Furthermore, if $c' = a \cdot b$, then $c_0' := g^b, g^{a \cdot b} \cdot m_{0,0}$ is identically distributed to a normal sanitization of c_0^1, since b is a random exponent, just as $r_0 \cdot t + s_0$ is for t chosen at random (and computation takes place in the field Z_q). Hence, we emulate the experiment $\mathsf{Exp}^0_{\mathcal{A}}$. In the other case, i.e., if c' is a random element, then c_0^1 is a pair of uniformly random group elements - just as in experiment $\mathsf{Exp}^1_{\mathcal{A}}$. □

This concludes in the following theorem.

Theorem 5.12. *The above enhanced sPKE scheme $\mathcal{E}$ satisfies*

$$\mathsf{Adv}^{\mathsf{sPKE\text{-}san\text{-}CCA}}_{\mathcal{E},\mathcal{A}} \leq 8 \cdot (\mathsf{Adv}^{\mathsf{DDH}}_{\bar{\mathcal{B}}} + \mathsf{Adv}^{\mathsf{NIZK\text{-}snd}}_{\mathsf{NIZK},\mathcal{A}'} + \mathsf{Adv}^{\mathsf{Sig\text{-}EUF\text{-}CMA}}_{\mathsf{Sig},\mathcal{A}''} + 2 \cdot q^2 \cdot 2^{-\kappa}),$$

where the adversaries are described in the respective lemmata. In particular, adversary $\bar{\mathcal{B}}$ is defined as being a mixture of adversaries $\mathcal{B}_0$ up to $\mathcal{B}_3$ of Lemma 5.11, i.e., $\bar{\mathcal{B}}$ samples a random number i between 0 and 3 and executes adversary $\mathcal{B}_i$.

The theorem implies in particular, that if the underlying cryptographic primitives are secure, then the scheme $\mathcal{E}$ has a secure sanitization procedure.

5.3.4 Privacy

For this case, we consider a hybrid experiment $\mathsf{Exp}^{S,b_1,b_2}_{\mathcal{A}}$: Let $\mathsf{Exp}^{S,b_1,b_2}_{\mathcal{A}}$ be as $\mathsf{Exp}^{\mathsf{sPKE\text{-}IND\text{-}CCA}}_{\mathcal{E},\mathcal{A}}$, but where the common reference string crs is obtained via evaluation $(crs, \tau^{\mathsf{NIZK}}) \leftarrow S_1^{\mathsf{NIZK}}$ (instead via an invocation of $\mathsf{NIZK.Gen}$). Also, when computing the challenge ciphertext $c^* = (c_1, c_2, c_\sigma, \pi)$, π is generated by an invocation of $S_2(crs, \tau^{\mathsf{NIZK}}, (c_1, c_2, c_\sigma))$. The following versions of this hybrid system exist:

- Upon receiving the challenge (m_0, m_1), then compute c_1 as the encryption of m_{b_1} and c_2 as the encryption of m_{b_2}, and compute c_σ as in the real experiment (namely as the encryption of the two ElGamal public keys plus the accompanying signature). Simulate the proof $\pi \leftarrow S_2(crs, \tau^{\mathsf{NIZK}}, (c_1, c_2, c_\sigma))$ and output $c^* := (c_1, c_2, c_\sigma, \pi)$.

Lemma 5.13. *Let $\mathcal{A}$ be an adversary in the experiment $\mathsf{Exp}_{\mathcal{A}}^{S,b_1,b_2}$. Let BAD_1 be the event that $\mathcal{A}$ queries its decryption oracle $\mathcal{O}_{SD}$ with a valid but improper ciphertext $(c_1, c_2, c_\sigma, \pi)$, i.e., $(c_1, c_2, c_\sigma) \notin L$, but where π is an accepting proof, i.e., $\mathsf{NIZK.Ver}(crs, (c_1, c_2, c_\sigma), \pi) = 1)$, and which is not the equal to the challenge c^*. We construct an adversary $\mathcal{B}_0$ such that $\Pr^{\mathsf{Exp}_{\mathcal{A}}^{S,b_0,b_1}}[\mathsf{BAD}_1] \leq \mathsf{Adv}_{\mathsf{NIZK},\mathcal{A}'}^{\mathsf{NIZK\text{-}sim\text{-}snd}}$.*

Proof. The claim follows from the simulation soundness property of the assumed NIZK scheme: upon receiving a CRS crs' (from its own challenger), $\mathcal{A}'$ defines $crs \leftarrow crs'$ and emulates all further steps of the experiment $\mathsf{Exp}_{\mathcal{A}}^{S,b_0,b_1}$. This is in particular possible when possessing all secret keys. In particular, simulate the challenge ciphertext using the prove-oracle of its challenger. Note further that without loss of generality the assumed adversary never asks to decrypt the challenge query c^*. If BAD_1 occurs, then it holds that a valid forgery against the challenged NIZK scheme (with CRS crs) occurred. □

Lemma 5.14. *Let $\mathcal{A}$ be an adversary in the experiment $\mathsf{Exp}_{\mathcal{A}}^{S,b_0,b_1}$. Let BAD_2 be the event that $\mathcal{A}$ queries its decryption oracle $\mathcal{O}_{SD}$ with a valid and proper ciphertext (not equal to the challenge) $(c_1, c_2, c_\sigma, \pi)$ (i.e., $(c_1, c_2, c_\sigma) \in L$ and π is accepting), but where c_σ is the encryption of a triple (ek_1, ek_2, σ), such that the pair (ek_1, ek_2) has never been output by the experiment or the oracle $\mathcal{O}_G$. We construct an adversary $\mathcal{A}'$ such that $\Pr^{\mathsf{Exp}_{\mathcal{A}}^{S,b_0,b_1}}[\mathsf{BAD}_2] \leq \mathsf{Adv}_{\mathsf{Sig},\mathcal{A}'}^{\mathsf{Sig\text{-}EUF\text{-}CMA}}$*

Proof. Follows analogously to the previous paragraph. □

Recall that the winning condition of an assumed adversary $\mathcal{A}$ in the IND-CCA game is

$$W_{pr} := [b' = b \ \wedge \ c^* \notin C_{\mathcal{A}_2}].$$

We assume without loss of generality that the assumed adversary $\mathcal{A}$ against the privacy game does not query the challenge c^* to its decryption oracle since one can construct, from an adversary that does so, a new adversary that simply guesses the bit once it observes that a violation of the condition $c^* \notin C_{\mathcal{A}_2}$ would happen.

Lemma 5.15. *For any adversary $\mathcal{A}$, we have that*

$$\begin{aligned}
&\Pr^{\mathsf{Exp}^{\mathsf{sPKE\text{-}IND\text{-}CCA}}_{\mathcal{E},\mathcal{A}}}[W_{pr}] = \frac{1}{2}\cdot \Pr^{\mathsf{Exp}^{\mathsf{sPKE\text{-}IND\text{-}CCA}}_{\mathcal{E},\mathcal{A}}}[b'=0 \mid b=0] + \frac{1}{2}\Pr^{\mathsf{Exp}^{\mathsf{sPKE\text{-}IND\text{-}CCA}}_{\mathcal{E},\mathcal{A}}}[b'=1 \mid b=1] \\
&= \frac{1}{2} + \frac{1}{2}(\Pr^{\mathsf{Exp}^{\mathsf{sPKE\text{-}IND\text{-}CCA}}_{\mathcal{E},\mathcal{A}}}[b'=1 \mid b=1] - \Pr^{\mathsf{Exp}^{\mathsf{sPKE\text{-}IND\text{-}CCA}}_{\mathcal{E},\mathcal{A}}}[b'=1 \mid b=0]) \\
&= \frac{1}{2} + \frac{1}{2}(\underbrace{(\Pr^{\mathsf{Exp}^{\mathsf{sPKE\text{-}IND\text{-}CCA}}_{\mathcal{E},\mathcal{A}}}[b'=1 \mid b=1] - \Pr^{\mathsf{Exp}^{S,1,1}_{\mathcal{A}}}[b'=1])}_{\overset{(1)}{\leq}\ \mathsf{Adv}^{\mathsf{NIZK\text{-}ZK}}_{\mathsf{NIZK},S,\mathcal{B}_1}} \\
&\qquad + \underbrace{(\Pr^{\mathsf{Exp}^{S,1,1}_{\mathcal{A}}}[b'=1] - \Pr^{\mathsf{Exp}^{S,1,0}_{\mathcal{A}}}[b'=1])}_{\overset{(2)}{\leq}\ 2\cdot\mathsf{Adv}^{\mathsf{DDH}}_{\mathcal{B}_2} + p_{\mathsf{fail}}} + \underbrace{(\Pr^{\mathsf{Exp}^{S,1,0}_{\mathcal{A}}}[b'=1] - \Pr^{\mathsf{Exp}^{S,0,0}_{\mathcal{A}}}[b'=1])}_{\overset{(3)}{\leq}\ 2\cdot\mathsf{Adv}^{\mathsf{DDH}}_{\mathcal{B}_3} + p_{\mathsf{fail}}} \\
&\qquad + \underbrace{(\Pr^{\mathsf{Exp}^{S,0,0}_{\mathcal{A}}}[b'=1] - \Pr^{\mathsf{Exp}^{\mathsf{sPKE\text{-}IND\text{-}CCA}}_{\mathcal{E},\mathcal{A}}}[b'=1 \mid b=0])}_{\overset{(4)}{\leq}\ \mathsf{Adv}^{\mathsf{NIZK\text{-}ZK}}_{\mathsf{NIZK},S,\mathcal{B}_1}}).
\end{aligned}$$

where the respective adversaries $\mathcal{B}_i$ are derived in the proof below.

Proof. The proof closely follows the proof proposed by Lindell [Lin06]. In particular, (1) and (4) follow by a straightforward reduction to the underlying zero-knowledge property. Consider (1) and define the following adversary $\mathcal{B}_1$ which receives a crs from its own challenger. Then, it emulates towards $\mathcal{A}$ the experiment $\mathsf{Exp}^{S,1,1}_{\mathcal{A}}$. This can be done, when possessing all the decryption keys generated in the experiment. Furthermore, when generating the challenge $c^* = (c_1, c_2, c_\sigma)$, which is the correct decryption of m_1 in this case, $\mathcal{B}_1$ asks its proving oracle to obtain a valid proof to this correct statement. We observe that if the CRS and the proofs are real, then this is equivalent to the experiment $\mathsf{Exp}^{\mathsf{sPKE\text{-}IND\text{-}CCA}}_{\mathcal{E},\mathcal{A}}$ when $b=1$, and if the CRS and the proofs are simulated, then this is equivalent to $\mathsf{Exp}^{S,1,1}_{\mathcal{A}}$ and (1) follows.

Also, equations (2) and (3) follow along the lines of the proof in [Lin06]: consider (2) and define the adversary $\mathcal{B}_2$ as follows. $\mathcal{B}_2$ receives a candidate DDH triple (g^a, g^b, g^c) and declares g^a to be the encryption key ek_2. It further generates all the remaining keys of the experiment (and thus lacks only the decryption key $dk_2 = a$). In particular, $\mathcal{B}_2$ chooses the bit b of the game and key pair (ek_1, dk_1) and defines the public key $ek := (ek_1, ek_2, \sigma)$ (where the signature can be generated by $\mathcal{B}_2$). When receiving the challenge (m_0, m_1) of $\mathcal{A}$, $\mathcal{B}_2$ generates one ElGamal encryption of m_b as follows:

$$c_2 := ((g^b)^{\bar{r}}, (g^c)^{\bar{r}}, g^b, g^c \cdot m_b).$$

It further defines c_1 as an ElGamal encryption of m_1, encrypts both keys and the signature to obtain c_σ, and simulates a NIZK proof π. We note that if the candidate triple is a DDH triple then $c_2 = (g^{b\cdot\bar{r}}, (g^a)^{b\cdot\bar{r}}, g^b, (g^a)^b \cdot m_b) = (g^{b\cdot\bar{r}}, (ek_2)^{b\cdot\bar{r}}, g^b, (ek_2)^b \cdot m_b)$ and thus corresponds to a correctly distributed ElGamal encryption.

When $\mathcal{A}_2$ outputs its decision bit b', $\mathcal{B}_2$ outputs $d=1$ if $b=b'$ and $d=0$ otherwise. Assume that none of the above defined bad events happen. Then, if (g^a, g^b, g^c) is a random triple, $\mathcal{B}_2$ outputs a uniform bit since the challenge ciphertext is independent of b. If the candidate triple is indeed a DDH triple (g^a, g^b, g^{ab}), then $\mathcal{B}_2$ emulates either the experiment $\mathsf{Exp}^{S,1,0}_{\mathcal{A}}$ or the experiment $\mathsf{Exp}_{\mathcal{A}}^{S,1,1}$, each with probability one-half. The output of $\mathcal{B}_2$ is thus

$$\Pr^{\mathsf{DDH}_{\mathcal{B}_2}^{\mathrm{rand}}}[d=1] = \frac{1}{2} \quad \text{and}$$
$$\Pr^{\mathsf{DDH}_{\mathcal{B}_2}^{\mathrm{real}}}[d=1] = \frac{1}{2} \cdot \Pr^{\mathsf{Exp}_{\mathcal{A}}^{S,1,1}}[b'=1] + \frac{1}{2} \cdot \Pr^{\mathsf{Exp}_{\mathcal{A}}^{S,1,0}}[b'=0]$$
$$= \frac{1}{2} \cdot \Pr^{\mathsf{Exp}_{\mathcal{A}}^{S,1,1}}[b'=1] + \frac{1}{2} \cdot (1 - \Pr^{\mathsf{Exp}_{\mathcal{A}}^{S,1,0}}[b'=1])$$
$$= \frac{1}{2} + \frac{1}{2} \cdot (\Pr^{\mathsf{Exp}_{\mathcal{A}}^{S,1,1}}[b'=1] - \Pr^{\mathsf{Exp}_{\mathcal{A}}^{S,1,0}}[b'=1]).$$

and therefore $\mathsf{Adv}_{\mathcal{B}_2}^{\mathsf{DDH}} = \frac{1}{2} \cdot (\Pr^{\mathsf{Exp}_{\mathcal{A}}^{S,1,1}}[b'=1] - \Pr^{\mathsf{Exp}_{\mathcal{A}}^{S,1,0}}[b'=1])$.

The proof is concluded by the observation that the oracle $\mathcal{O}_{SD}$ can be emulated perfectly, even without knowledge of $dk_{0,1} = a$, except with probability at most p_{fail} derived the same way as in the previous paragraph (except that we need here simulation soundness as opposed to ordinary soundness). Equation (3) follows similarly. □

This concludes in the following theorem.

Theorem 5.16. *The above enhanced sPKE scheme $\mathcal{E}$ satisfies*

$$\mathsf{Adv}_{\mathcal{E},\mathcal{A}}^{\mathsf{sPKE\text{-}IND\text{-}CCA}} \leq 4 \cdot \mathsf{Adv}_{\bar{\mathcal{B}}}^{\mathsf{DDH}} + 2 \cdot \mathsf{Adv}_{\mathsf{NIZK},S,\mathcal{B}_1}^{\mathsf{NIZK\text{-}ZK}} + 2 \cdot (\mathsf{Adv}_{\mathsf{NIZK},\mathcal{A}'}^{\mathsf{NIZK\text{-}sim\text{-}snd}} + \mathsf{Adv}_{\mathsf{Sig},\mathcal{A}''}^{\mathsf{Sig\text{-}EUF\text{-}CMA}} + 2 \cdot q^2 \cdot 2^{-\kappa}),$$

where the adversaries are defined in the respective lemmata and adversary $\bar{\mathcal{B}}$ is the mixture of adversaries $\mathcal{B}_2$ and $\mathcal{B}_3$ of Lemma 5.15, i.e., $\bar{\mathcal{B}}$ samples a random coin and either executes adversary $\mathcal{B}_2$ or $\mathcal{B}_3$.

The theorem implies in particular that if the underlying cryptographic primitives are secure, then the scheme $\mathcal{E}$ is IND-CCA secure.

5.3.5 Anonymity

For this case, we consider the hybrid experiment $\mathsf{Exp}_{\mathcal{A}}^{S,i}$: Let $\mathsf{Exp}_{\mathcal{A}}^{S,i}$ be basically as $\mathsf{Exp}_{\mathcal{E},\mathcal{A}}^{\mathsf{sPKE\text{-}IK\text{-}CCA}}$, but where, for all of them, the common reference string crs is obtained via $(crs, \tau^{\mathsf{NIZK}}) \leftarrow S_1^{\mathsf{NIZK}}$ (instead via an invocation of $\mathsf{NIZK.Gen}$). Furthermore, when computing the challenge ciphertext $c^* = (c_1, c_2, c_\sigma, \pi)$, π is generated by an invocation of $S_2(crs, \tau^{\mathsf{NIZK}}, (c_1, c_2, c_\sigma))$. The following versions of this hybrid system exist:

- For $i = 0$, upon receiving the challenge m, then compute c_b^1 as the encryption of m under key $ek_{b,1}$ and c_b^2 as the encryption of m under $ek_{b,2}$, and compute c_σ as in the real experiment (namely as the encryption of the two ElGamal public keys plus the accompanying signature). Simulate the proof $\pi \leftarrow S_2(crs, \tau^{\mathsf{NIZK}}, (c_b^1, c_b^2, c_\sigma))$ and output $c^* := (c_b^1, c_b^2, c_\sigma, \pi)$.
- For $i = 1$, the hybrid system acts as above, but upon receiving the challenge m, instead of computing c_σ as an encryption of $(ek_{b,1}, ek_{b,2}, \sigma)$, encrypt the message 0^n, for an appropriate length n that matches the length of (an appropriate encoding of) the triple $(ek_{b,1}, ek_{b,2}, \sigma)$. The rest is done as for the previous hybrid.

- For $i = 2$, the hybrid acts as above, but in addition, when receiving the challenge m, instead of computing c_0^1 as the encryption of m under key $ek_{0,1}$, choose a new encryption key g^x, for a uniformly random exponent x. If $b = 0$ in the game, then the first ciphertext c_0^1 would be an encryption under "key" g^x. The remaining steps are as usual.

- For $i = 3$, in addition to all the steps above, here upon a challenge, we also compute c_0^2 relative to a freshly chosen public key $g^{x'}$.

- For $i = 4$, in addition to all the steps above, here upon a challenge, we also compute c_1^1 relative to a freshly chosen public key $g^{x''}$ instead of as an encryption under $ek_{1,1}$.

- For $i = 5$, in addition to all the steps above, here upon a challenge, we also compute c_1^2 relative to a freshly chosen public key $g^{x'''}$.

We observe that in the final hybrid experiment $\mathsf{Exp}_{\mathcal{A}}^{S,5}$, the advantage in guessing b is at most $\frac{1}{2}$, since all outputs given to the adversary are independent of the actual keys ek_0 or ek_1 and hence of the bit b chosen by the game.

As in the analysis above, we have the analogous lemmata:

Lemma 5.17. *Let $\mathcal{A}$ be an adversary in the experiment $\mathsf{Exp}_{\mathcal{A}}^{S,i}$. Let BAD_1 be the event that $\mathcal{A}$ queries at least one of its decryption oracles $\mathcal{O}_{SD_j}$ with a valid but improper ciphertext $(c_1, c_2, c_\sigma, \pi)$ (which is not the challenge c^*), i.e., $(c_1, c_2, c_\sigma) \notin L$, but where π is an accepting proof, i.e., $\mathsf{NIZK.Ver}(crs, (c_1, c_2, c_\sigma), \pi) = 1$). We construct an adversary $\mathcal{A}'$ such that $\Pr^{\mathsf{Exp}_{\mathcal{A}}^{S,1}}[\mathsf{BAD}_1] \leq \mathsf{Adv}_{\mathsf{NIZK},\mathcal{A}'}^{\mathsf{NIZK\text{-}sim\text{-}snd}}$.*

Proof. The claim follows from the simulation soundness property of the assumed NIZK scheme similar to the statements in the previous paragraph. $\square$

Lemma 5.18. *Let $\mathcal{A}$ be an adversary in the experiment $\mathsf{Exp}_{\mathcal{A}}^{S,i}$. Let BAD_2 be the event that $\mathcal{A}$ queries at least one of its decryption oracles $\mathcal{O}_{SD_j}$ with a valid and proper ciphertext (which is not the challenge) $(c_1, c_2, c_\sigma, \pi)$ (i.e., $(c_1, c_2, c_\sigma) \in L$ and π is accepting), but where c_σ is the encryption of a triple (ek_1, ek_2, σ), such that the pair (ek_1, ek_2) has never been output by the experiment or the oracle $\mathcal{O}_G$. We construct an adversary $\mathcal{A}'$ such that $\Pr^{\mathsf{Exp}_{\mathcal{A}}^{S,i}}[\mathsf{BAD}_2] \leq \mathsf{Adv}_{\mathsf{Sig},\mathcal{A}'}^{\mathsf{Sig\text{-}EUF\text{-}CMA}}$*

Proof. Follows analogously to the previous paragraph. $\square$

Recall that the winning condition of an assumed adversary $\mathcal{A}$ in the IK-CCA game is

$$W_{ik} := [b' = b \ \wedge \ c^* \notin C_{\mathcal{A}_2}].$$

We assume without loss of generality that the assumed adversary $\mathcal{A}$ against the anonymity game does not query the challenge c^* to its decryption oracle since one can construct, from an adversary that does so, a new adversary that simply guesses the bit once it observes that a violation of the condition $c^* \notin C_{\mathcal{A}_2}$ would happen.

Lemma 5.19. *For any adversary $\mathcal{A}$, we have that*

$$\Pr^{\mathsf{Exp}^{\mathsf{sPKE\text{-}IK\text{-}CCA}}_{\mathcal{E},\mathcal{A}}}[W_{ik}] = \underbrace{\Pr^{\mathsf{Exp}^{S,5}_{\mathcal{A}}}[W_{ik}]}_{=\frac{1}{2}} + \sum_{i=1}^{4} \underbrace{\Pr^{\mathsf{Exp}^{S,i}_{\mathcal{A}}}[W_{ik}] - \Pr^{\mathsf{Exp}^{S,i+1}_{\mathcal{A}}}[W_{ik}]}_{\overset{(1)}{\leq}\ \mathsf{Adv}^{\mathsf{DDH}}_{\mathcal{B}_{i+1}} + p_{\mathsf{fail}}}$$
$$+ \underbrace{\Pr^{\mathsf{Exp}^{S,0}_{\mathcal{A}}}[W_{ik}] - \Pr^{\mathsf{Exp}^{S,1}_{\mathcal{A}}}[W_{ik}]}_{\overset{(2)}{\leq}\ 2\cdot\mathsf{Adv}^{\mathsf{PKE\text{-}IND\text{-}CPA}}_{\mathsf{PKE},\mathcal{B}_1}} + \underbrace{\Pr^{\mathsf{Exp}^{S,0}_{\mathcal{A}}}[W_{ik}] - \Pr^{\mathsf{Exp}^{\mathsf{sPKE\text{-}IK\text{-}CCA}}_{\mathcal{E},\mathcal{A}}}[W_{ik}]}_{\overset{(3)}{\leq}\ \mathsf{Adv}^{\mathsf{NIZK\text{-}ZK}}_{\mathsf{NIZK},S,\mathcal{B}_0}}.$$

where the respective adversaries $\mathcal{B}_i$ are derived in the proof below.

Proof sketch. The proof closely follows along the lines of the previous proofs. In particular, (3) is again a straightforward reduction to the zero-knowledge property of the underlying NIZK scheme. (2) Follows from the CPA security of the assumed PKE scheme: consider the adversary $\mathcal{B}_1$ that obtains a public key ek from its CPA challenger. $\mathcal{B}_1$ generates all remaining keys himself to be able to emulate the steps of the experiment $\mathsf{Exp}^{S,1}$ towards the (assumed) adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$. Note that when $\mathcal{B}_1$ never needs to decrypt any of the ciphertexts c_σ in the experiment (and thus, no decryption key is needed). When $\mathcal{A}_1$ outputs his challenge, $\mathcal{B}_1$ first asks the challenge $(m_0 = (ek_{b,1}, ek_{b,2}, \sigma), m_1 = 0^n)$ to its own challenger and obtains a ciphertext c_σ. If the challenger outputs an encryption of 0^n, then this is equivalent to experiment $\mathsf{Exp}^{S,1}_{\mathcal{A}}$, and if the challenger returns an encryption of $(ek_{b,1}, ek_{b,2}, \sigma)$, then this is equivalent to $\mathsf{Exp}^{S,0}_{\mathcal{A}}$. Hence, if $\mathcal{B}_1$ returns 1 in case the guess b' of $\mathcal{A}_2$ is correct (i.e., is equal to the bit b that $\mathcal{B}_1$ emulated) and 0 otherwise, then this is a biased bit with bias half the difference between the two experiments in question. Equation (2) hence follows from an analogous computation as done, for example, in the proof of Lemma 5.15.

Finally, we consider equation (1) and describe adversary $\mathcal{B}_1$ (the remaining cases are analogous). Adversary $\mathcal{B}_1$ works as follows. It receives a candidate DDH triple (g^a, g^b, g^c) and declares g^a to be the encryption key $ek_{0,1}$. It further generates all the remaining keys of the experiment (and thus lacks only the decryption key $dk_{0,1} = a$). In particular, $\mathcal{B}_1$ chooses the bit b of the game and the remaining key pairs, which includes $(ek_{0,2}, dk_{0,2})$ (to be able to emulate the oracle $\mathcal{O}_{SD_0}$) and defines the public key $ek0 \coloneqq (ek_{0,1}, ek_{0,2}, \sigma)$ (where the signature can be generated by $\mathcal{B}_1$). When receiving the challenge (m) of $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, generate one ElGamal encryption of m as follows:

$$c_0^1 \coloneqq ((g^b)^{\bar{r}}, (g^c)^{\bar{r}}, g^b, g^c \cdot m).$$

It further defines the other parts c_j^k as the experiment does (including encrypting the zero-string and simulating a proof). We note that if the candidate triple is a DDH triple then $c_2 = (g^{b\cdot\bar{r}}, (g^a)^{b\cdot\bar{r}}, g^b, (g^a)^b \cdot m_b) = (g^{b\cdot\bar{r}}, (ek_2)^{b\cdot\bar{r}}, g^b, (ek_2)^b \cdot m_b)$ and thus corresponds to a correctly distributed ElGamal encryption as in the experiment $\mathsf{Exp}^{S,1}_{\mathcal{A}}$. However, if (g^a, g^b, g^c) then the encryption c_0^1 is distributed identically to a fresh encryption of m relative to a random public key g^x, just as it is done in experiment $\mathsf{Exp}^{S,2}_{\mathcal{A}}$. Hence when $\mathcal{A}_2$ outputs its decision bit b', $\mathcal{B}_2$ outputs $d = 1$ if $b = b'$ and $d = 0$ otherwise and constitutes a distinguihser of DDH triples and random triples with advantage the same as the difference in probabilities of winning the respective experiments. □

This concludes in the following theorem.

Theorem 5.20. *The above enhanced sPKE scheme $\mathcal{E}$ satisfies*

$$\begin{aligned}\mathsf{Adv}^{\text{sPKE-IK-CCA}}_{\mathcal{E},\mathcal{A}} \leq 8\cdot\mathsf{Adv}^{\text{DDH}}_{\bar{\mathcal{B}}} + 4\cdot\mathsf{Adv}^{\text{PKE-IND-CPA}}_{\text{PKE},\mathcal{B}_1} + 2\cdot\mathsf{Adv}^{\text{NIZK-ZK}}_{\text{NIZK},S,\mathcal{B}_0} \\ + 8\cdot\big(\mathsf{Adv}^{\text{NIZK-sim-snd}}_{\text{NIZK},\mathcal{A}'} + \mathsf{Adv}^{\text{Sig-EUF-CMA}}_{\text{Sig},\mathcal{A}''} + 2\cdot q^2\cdot 2^{-\kappa}\big),\end{aligned}$$

where the adversaries are defined in the respective lemmata, and $\bar{\mathcal{B}}$ is a mixture of the adversaries $\mathcal{B}_i$ of Lemma 5.19.

The theorem implies in particular, that if the underlying cryptographic primitives are secure, then the scheme $\mathcal{E}$ is IK-CCA secure.

6 Construction of an ACE Scheme

6.1 Construction for Equality

Following Fuchsbauer et al. [FGKO17], we first construct an ACE scheme for the equality policy, i.e., $P(i,j) = 1 \Leftrightarrow i = j$, and then use such a scheme in another construction for richer policies.

Let sPKE be a sanitizable public-key encryption scheme, let Sig be a signature scheme, and let F be a PRF. Further let NIZK be a NIZK proof of knowledge system for the language $L := \{x \mid \exists w\ (x,w) \in R\}$, where the relation R is defined as follows: for $x = \big(vk^{\mathsf{Sig}}, \tilde{c}\big)$ and $w = \big(ek_i^{\mathsf{sPKE}}, m, r, vk_i^{\mathsf{Sig}}, \sigma_i^{\mathsf{Sig}}, \sigma_c^{\mathsf{Sig}}\big)$, $(x,w) \in R$ if and only if

$$\begin{aligned}\tilde{c} = \mathsf{sPKE.Enc}\big(ek_i^{\mathsf{sPKE}}, m; r\big) \ \wedge\ \mathsf{Sig.Ver}\big(vk^{\mathsf{Sig}}, \big[ek_i^{\mathsf{sPKE}}, vk_i^{\mathsf{Sig}}\big], \sigma_i^{\mathsf{Sig}}\big) = 1 \\ \wedge\ \mathsf{Sig.Ver}\big(vk_i^{\mathsf{Sig}}, \tilde{c}, \sigma_c^{\mathsf{Sig}}\big) = 1.\end{aligned}$$

We define an ACE with modification detection scheme ACE as follows:

Setup: On input a security parameter 1^κ and a policy $P\colon [n]\times[n] \to \{0,1\}$ with $P(i,j) = 1 \Leftrightarrow i = j$, the algorithm $\mathsf{ACE.Setup}$ picks a random PRF key K for a PRF F, and runs

$$\begin{aligned}\big(sp^{\mathsf{sPKE}}, msk^{\mathsf{sPKE}}\big) &\leftarrow \mathsf{sPKE.Setup}(1^\kappa),\\ \big(vk^{\mathsf{Sig}}, sk^{\mathsf{Sig}}\big) &\leftarrow \mathsf{Sig.Gen}(1^\kappa),\\ crs^{\mathsf{NIZK}} &\leftarrow \mathsf{NIZK.Gen}(1^\kappa).\end{aligned}$$

It outputs the master secret key $msk^{\mathsf{ACE}} := \big(K, msk^{\mathsf{sPKE}}, vk^{\mathsf{Sig}}, sk^{\mathsf{Sig}}, crs^{\mathsf{NIZK}}\big)$ and the sanitizer parameters $sp^{\mathsf{ACE}} := \big(sp^{\mathsf{sPKE}}, vk^{\mathsf{Sig}}, crs^{\mathsf{NIZK}}\big)$.

Key Generation: The algorithm $\mathsf{ACE.Gen}$ on input a master secret key $msk^{\mathsf{ACE}} = \big(K, msk^{\mathsf{sPKE}}, vk^{\mathsf{Sig}}, sk^{\mathsf{Sig}}, crs^{\mathsf{NIZK}}\big)$, a role $i \in [n]$, and a type $t \in \{\mathtt{sen}, \mathtt{rec}\}$, computes

$$\big(ek_i^{\mathsf{sPKE}}, dk_i^{\mathsf{sPKE}}\big) \leftarrow \mathsf{sPKE.Gen}\big(msk^{\mathsf{sPKE}}; F_K([i,0])\big).$$

If $t = \mathtt{sen}$, it further computes

$$\begin{aligned}\big(vk_i^{\mathsf{Sig}}, sk_i^{\mathsf{Sig}}\big) &\leftarrow \mathsf{Sig.Gen}\big(1^\kappa; F_K([i,1])\big),\\ \sigma_i^{\mathsf{Sig}} &\leftarrow \mathsf{Sig.Sign}\big(sk^{\mathsf{Sig}}, \big[ek_i^{\mathsf{sPKE}}, vk_i^{\mathsf{Sig}}\big]; F_K([i,2])\big).\end{aligned}$$

If $t = \mathtt{sen}$, it outputs the encryption key $ek_i^{\mathsf{ACE}} := \big(vk^{\mathsf{Sig}}, ek_i^{\mathsf{sPKE}}, vk_i^{\mathsf{Sig}}, sk_i^{\mathsf{Sig}}, \sigma_i^{\mathsf{Sig}}, crs^{\mathsf{NIZK}}\big)$; if $t = \mathtt{rec}$, it outputs the decryption key $dk_i^{\mathsf{ACE}} := dk_i^{\mathsf{sPKE}}$.

Encrypt: On input an encryption key $ek_i^{\mathsf{ACE}} = \big(vk^{\mathsf{Sig}}, ek_i^{\mathsf{sPKE}}, vk_i^{\mathsf{Sig}}, sk_i^{\mathsf{Sig}}, \sigma_i^{\mathsf{Sig}}, crs^{\mathsf{NIZK}}\big)$ and a message $m \in \mathcal{M}^{\mathsf{ACE}}$, the algorithm $\mathsf{ACE.Enc}$ samples randomness r and computes

$$\begin{aligned}\tilde{c} &\leftarrow \mathsf{sPKE.Enc}\big(ek_i^{\mathsf{sPKE}}, m; r\big),\\ \sigma_c^{\mathsf{Sig}} &\leftarrow \mathsf{Sig.Sign}\big(sk_i^{\mathsf{Sig}}, \tilde{c}\big),\\ \pi^{\mathsf{NIZK}} &\leftarrow \mathsf{NIZK.Prove}\big(crs^{\mathsf{NIZK}}, x := \big(vk^{\mathsf{Sig}}, \tilde{c}\big), w := \big(ek_i^{\mathsf{sPKE}}, m, r, vk_i^{\mathsf{Sig}}, \sigma_i^{\mathsf{Sig}}, \sigma_c^{\mathsf{Sig}}\big)\big).\end{aligned}$$

It outputs the ciphertext $c := \big(\tilde{c}, \pi^{\mathsf{NIZK}}\big)$.

Sanitizer: On input sanitizer parameters $sp^{\mathsf{ACE}} = \big(sp^{\mathsf{sPKE}}, vk^{\mathsf{Sig}}, crs^{\mathsf{NIZK}}\big)$ and a ciphertext $c = \big(\tilde{c}, \pi^{\mathsf{NIZK}}\big)$, the algorithm $\mathsf{ACE.San}$ outputs the sanitized ciphertext $c' \leftarrow \mathsf{sPKE.San}\big(sp^{\mathsf{sPKE}}, \tilde{c}\big)$ if $\mathsf{NIZK.Ver}\big(crs^{\mathsf{NIZK}}, x := \big(vk^{\mathsf{Sig}}, \tilde{c},\big), \pi^{\mathsf{NIZK}}\big) = 1$; otherwise, it outputs $\perp$.

Decrypt: The algorithm $\mathsf{ACE.Dec}$ on input a decryption key dk_j^{ACE} and a sanitized ciphertext c', outputs the message $m \leftarrow \mathsf{sPKE.Dec}(dk_j^{\mathsf{ACE}}, c')$.

Modification detection: The algorithm $\mathsf{ACE.DMod}$ on input sp^{ACE}, $c_1 = \big(\tilde{c}_1, \pi_1^{\mathsf{NIZK}}\big)$, and $c_2 = \big(\tilde{c}_2, \pi_2^{\mathsf{NIZK}}\big)$, outputs 1 if $\tilde{c}_1 = \tilde{c}_2$, and 0 otherwise.

Our scheme enjoys perfect correctness since the underlying sPKE and signature schemes are perfectly correct and the NIZK is perfectly complete, i.e.,

$$\mathsf{Adv}_{\mathsf{ACE},\mathcal{A}}^{\mathsf{ACE\text{-}corr}} = 0$$

for all $\mathcal{A}$. The scheme is also detectable as the following lemma shows.

Lemma 6.1. *Let* ACE *be the scheme from above and let* $\mathcal{A}$ *be a probabilistic algorithm. Then, there exist probabilistic algorithms* $\mathcal{A}_{\mathsf{PRF}}$ *and* $\mathcal{A}_{\mathsf{rob}}$ *(roughly as efficient as emulating an execution of* $\mathsf{Exp}_{\mathsf{ACE},\mathcal{A}}^{\mathsf{ACE\text{-}dtct}}$*) such that*

$$\mathsf{Adv}_{\mathsf{ACE},\mathcal{A}}^{\mathsf{ACE\text{-}dtct}} = \mathsf{Adv}_{F,\mathcal{A}_{\mathsf{PRF}}}^{\mathsf{PRF}} + \mathsf{Adv}_{\mathsf{sPKE},\mathcal{A}_{\mathsf{rob}}}^{\mathsf{sPKE\text{-}USROB}}.$$

Proof. We assume without loss of generality that $\mathcal{A}$ returns (m, i, j) with $P(i, j) = 0$ since doing otherwise can only reduce the advantage. Let $H_0 := \mathsf{Exp}_{\mathsf{ACE},\mathcal{A}}^{\mathsf{ACE\text{-}dtct}}$, let H_1 be as H_0 where F_K is replaced by a truly uniform function U, and let W be the event that $\mathcal{A}$ wins the detectability game, i.e.,

$$W := \big[\mathsf{ACE.Dec}\big(dk_j^{\mathsf{ACE}}, \mathsf{ACE.San}(sp^{\mathsf{ACE}}, \mathsf{ACE.Enc}(ek_i^{\mathsf{ACE}}, m))\big) \neq \perp\big].$$

We first show that the difference in the winning probability in H_0 and H_1 is bounded by the PRF advantage.

Claim 1. *There exists a probabilistic algorithm* $\mathcal{A}_{\mathsf{PRF}}^{\mathcal{O}(\cdot)}$ *such that*

$$\Pr^{H_0}[W] - \Pr^{H_1}[W] = \mathsf{Adv}_{F,\mathcal{A}_{\mathsf{PRF}}}^{\mathsf{PRF}}.$$

Proof of claim. Consider $\mathcal{A}_{\mathsf{PRF}}^{\mathcal{O}(\cdot)}$ that emulates an execution of H_0, where all invocations of $F_K(\cdot)$ are replaced by a call to the oracle $\mathcal{O}(\cdot)$. When $\mathcal{A}$ wins, $\mathcal{A}_{\mathsf{PRF}}$ outputs 1, and 0 otherwise. In case $\mathcal{O}(\cdot)$ corresponds to $F_K(\cdot)$, $\mathcal{A}_{\mathsf{PRF}}$ perfectly emulates H_0, if it corresponds to $U(\cdot)$, it perfectly emulates H_1. Hence,

$$\Pr^{H_0}[W] - \Pr^{H_1}[W] = \Pr\Big[\mathcal{A}_{\mathsf{PRF}}^{F_K(\cdot)}(1^\kappa) = 1\Big] - \Pr\Big[\mathcal{A}_{\mathsf{PRF}}^{U(\cdot)}(1^\kappa) = 1\Big] = \mathsf{Adv}_{F,\mathcal{A}_{\mathsf{PRF}}}^{\mathsf{PRF}}. \qquad \Diamond$$

We now construct a winner for the robustness game for sPKE. The algorithm $\mathcal{A}_{\mathsf{rob}}$ on input sp^{sPKE} emulates an execution of H_1. To answer queries of $\mathcal{A}$ to the key-generation oracle, $\mathcal{A}_{\mathsf{rob}}$ uses the oracle $\mathcal{O}_G$ to obtain encryption and decryption keys for sPKE; the required signature keys are generated internally. For each query (i,t), $\mathcal{A}_{\mathsf{rob}}$ remembers the generated keys ek_i^{ACE} and dk_i^{ACE}, and returns the same keys for subsequent queries with the same i. When $\mathcal{A}$ returns (m,i,j), $\mathcal{A}_{\mathsf{rob}}$ first checks whether i and j have been queried by $\mathcal{A}$ to the key-generation oracle. If not, $\mathcal{A}_{\mathsf{rob}}$ now generates these keys as above. Then, $\mathcal{A}_{\mathsf{rob}}$ computes $c \leftarrow \mathsf{sPKE.Enc}\big(ek_i^{\mathsf{sPKE}}, m\big)$, and returns (c, i_0, i_1), such that the i_0-th query and the i_1-th query to the key-generation oracle were for the roles i and j, respectively. Since P is the equality predicate, $P(i,j)=0$ is equivalent to $i_0 \neq i_1$. We further have by the perfect correctness of sPKE that $\mathsf{sPKE.Dec}\big(dk_i^{\mathsf{sPKE}}, \mathsf{sPKE.San}\big(sp^{\mathsf{sPKE}}, c\big)\big) \neq \perp$. Hence, $\mathcal{A}_{\mathsf{rob}}$ wins the robustness game if and only if $\mathcal{A}$ wins the detectability game in H_1. Using Claim 1, we can therefore conclude

$$\mathsf{Adv}^{\mathsf{ACE\text{-}dtct}}_{\mathsf{ACE},\mathcal{A}} = \Pr^{H_0}[W] = \mathsf{Adv}^{\mathsf{PRF}}_{F,\mathcal{A}_{\mathsf{PRF}}} + \Pr^{H_1}[W] = \mathsf{Adv}^{\mathsf{PRF}}_{F,\mathcal{A}_{\mathsf{PRF}}} + \mathsf{Adv}^{\mathsf{sPKE\text{-}USROB}}_{\mathsf{sPKE},\mathcal{A}_{\mathsf{rob}}}. \qquad \square$$

In the following, we prove the security of our scheme, which is summarized by the theorem below.

Theorem 6.2 (Informal)**.** *The above ACE scheme for equality is secure, i.e., all efficient adversaries have only negligible advantage in breaking the privacy, (strong) anonymity, sanitization, role-respecting, uniform-decryption, or ciphertext-unpredictability properties, if the underlying sPKE scheme is secure, the signature scheme is unforgeable, the proof system provides zero-knowledge and extractability, and if the function F is pseudo-random.*

We first show that our scheme satisfies the privacy definition from Definition 4.2 if the underlying sanitizable public-key encryption scheme is IND-CCA secure, the PRF is secure, and the NIZK is zero knowledge.

Theorem 6.3. *Let ACE be the scheme from above, let $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ be an attacker on the privacy such that $\mathcal{A}_1$ makes at most q_S queries of the form $(\cdot, \mathtt{sen})$ to the oracle $\mathcal{O}_G$, and at most q_D queries to $\mathcal{O}_{SD}$. Then, there exist probabilistic algorithms $\mathcal{A}_{\mathsf{PRF}}$, $\mathcal{A}_{\mathsf{ZK}}$, and $\mathcal{A}_{\mathsf{sPKE}}$ (which are all roughly as efficient as emulating an execution of $\mathsf{Exp}^{\mathsf{ACE\text{-}privAnon\text{-}CCA}}_{\mathsf{ACE},\mathcal{A}}$) such that*

$$\mathsf{Adv}^{\mathsf{ACE\text{-}priv\text{-}CCA}}_{\mathsf{ACE},\mathcal{A}} = 2 \cdot \mathsf{Adv}^{\mathsf{PRF}}_{F,\mathcal{A}_{\mathsf{PRF}}} + 2 \cdot \mathsf{Adv}^{\mathsf{NIZK\text{-}ZK}}_{\mathsf{NIZK},\mathcal{A}_{\mathsf{ZK}}} + (q_S + q_D + 1) \cdot \mathsf{Adv}^{\mathsf{sPKE\text{-}IND\text{-}CCA}}_{\mathsf{sPKE},\mathcal{A}_{\mathsf{sPKE}}}.$$

Proof. We assume without loss of generality that $\mathcal{A}$ ensures $i_0 = i_1$ and $P(i_0, j) = 0$ for all $j \in J$, since doing otherwise can only decrease the advantage. Let $H_0 := \mathsf{Exp}^{\mathsf{ACE\text{-}privAnon\text{-}CCA}}_{\mathsf{ACE},\mathcal{A}}$ and H_1 be as H_0 where F_K is replaced by a truly uniform random function U. The following can be proven as Claim 1 in the proof of Lemma 6.1.

Claim 1. *There exists a probabilistic algorithm $\mathcal{A}^{\mathcal{O}(\cdot)}_{\mathsf{PRF}}$ such that*

$$\Pr^{H_0}[b' = b] - \Pr^{H_1}[b' = b] = \mathsf{Adv}^{\mathsf{PRF}}_{F,\mathcal{A}_{\mathsf{PRF}}}.$$

Now let H_2 be as H_1, where we replace $crs^{\mathsf{NIZK}} \leftarrow \mathsf{NIZK.Gen}(1^\kappa)$ by $\big(crs^{\mathsf{NIZK}}, \tau^{\mathsf{NIZK}}\big) \leftarrow S_1^{\mathsf{NIZK}}(1^\kappa)$ in $\mathsf{ACE.Setup}$, and for the generation of the challenge ciphertext c^*, we replace $\pi^{\mathsf{NIZK}} \leftarrow \mathsf{NIZK.Prove}\big(crs^{\mathsf{NIZK}}, x, w\big)$ in $\mathsf{ACE.Enc}$ by $\pi^{\mathsf{NIZK}} \leftarrow S_2^{\mathsf{NIZK}}\big(crs^{\mathsf{NIZK}}, \tau^{\mathsf{NIZK}}, x\big)$.

Claim 2. *There exists a probabilistic algorithm $\mathcal{A}^{\mathcal{O}(\cdot,\cdot)}_{\mathsf{ZK}}$ such that*

$$\Pr^{H_1}[b' = b] - \Pr^{H_2}[b' = b] = \mathsf{Adv}^{\mathsf{NIZK\text{-}ZK}}_{\mathsf{NIZK},\mathcal{A}_{\mathsf{ZK}}}.$$

Proof of claim. The algorithm $\mathcal{A}_{\mathsf{ZK}}^{\mathcal{O}(\cdot,\cdot)}$ on input crs^{NIZK} proceeds as follows. It emulates an execution of H_1, where in $\mathsf{ACE.Setup}$, crs^{NIZK} is used instead of generating it, and for the generation of c^*, $\mathsf{NIZK.Prove}\big(crs^{\mathsf{NIZK}}, x, w\big)$ in $\mathsf{ACE.Enc}$ is replaced by the oracle query (x, w). Finally, $\mathcal{A}_{\mathsf{ZK}}^{\mathcal{O}(\cdot,\cdot)}$ outputs $\tilde{b} = 1$ if $\mathcal{A}_2$ returns $b' = b$, and $\tilde{b} = 0$ otherwise. Note that if crs^{NIZK} is generated by $\mathsf{NIZK.Gen}$ and $\mathcal{O}(\cdot,\cdot)$ corresponds to $\mathsf{NIZK.Prove}\big(crs^{\mathsf{NIZK}}, \cdot, \cdot\big)$, $\mathcal{A}_{\mathsf{ZK}}^{\mathcal{O}(\cdot,\cdot)}$ perfectly emulates H_1. Moreover, if crs^{NIZK} is generated together with τ^{NIZK} by S_1^{NIZK} and $\mathcal{O}(x, w)$ returns $S_2^{\mathsf{NIZK}}\big(crs^{\mathsf{NIZK}}, \tau^{\mathsf{NIZK}}, x\big)$, $\mathcal{A}_{\mathsf{ZK}}^{\mathcal{O}(\cdot,\cdot)}$ perfectly emulates H_2. Thus, the claim follows. $\Diamond$

We finally show how to transform any winner $\mathcal{A}$ for H_2 to a winner $\mathcal{A}_{\mathsf{sPKE}}$ for the IND-CCA game for the scheme sPKE. The strategy of our reduction is to guess which oracle queries of $\mathcal{A}_1$ are for the role i_0, use the key from the sPKE-scheme for these queries, and generate all other keys as H_2. Details follow. On input $(sp^{\mathsf{sPKE}}, ek^{\mathsf{sPKE}})$, $\mathcal{A}_{\mathsf{sPKE}}$ initializes $i_{q_0} \leftarrow \perp$, $k_q \leftarrow 1$, chooses $q_0 \twoheadleftarrow \{0, \ldots, q_S + q_D\}$ uniformly at random, runs $\big(vk^{\mathsf{Sig}}, sk^{\mathsf{Sig}}\big) \leftarrow \mathsf{Sig.Gen}(1^\kappa)$, and $\big(crs^{\mathsf{NIZK}}, \tau^{\mathsf{NIZK}}\big) \leftarrow S_1^{\mathsf{NIZK}}(1^\kappa)$, and gives $sp^{\mathsf{ACE}} := \big(sp^{\mathsf{sPKE}}, vk^{\mathsf{Sig}}, crs^{\mathsf{NIZK}}\big)$ to $\mathcal{A}_1$. It emulates the oracles for $\mathcal{A}_1$ as follows.

$\mathcal{O}_G(\cdot,\cdot)$**:** On query $(i, \mathtt{sen})$, if $k_q \neq q_0$ and $i \neq i_{q_0}$, then generate an encryption key $ek_i^{\mathsf{ACE}} := \big(vk^{\mathsf{Sig}}, ek_i^{\mathsf{sPKE}}, vk_i^{\mathsf{Sig}}, sk_i^{\mathsf{Sig}}, \sigma_i^{\mathsf{Sig}}, crs^{\mathsf{NIZK}}\big)$ as H_2 does, where $\big(ek_i^{\mathsf{sPKE}}, dk_i^{\mathsf{sPKE}}\big)$ is obtained via $\mathcal{O}_G$ and remembered for future queries. If $k_q = q_0$ or $i = i_{q_0}$, replace ek_i^{sPKE} by ek^{sPKE} and set $i_{q_0} \leftarrow i$. In both cases, set $k_q \leftarrow k_q + 1$ at the end. On query $(j, \mathtt{rec})$, obtain a decryption key via $\mathcal{O}_G$.

$\mathcal{O}_{SD}(\cdot,\cdot)$**:** On query $\big(j, c = \big(\tilde{c}, \pi^{\mathsf{NIZK}}\big)\big)$, if $k_q \neq q_0$ and $j \neq i_{q_0}$, run $c' \leftarrow \mathsf{ACE.San}(sp^{\mathsf{ACE}}, c)$, generate a decryption key dk_j^{ACE} as above, decrypt c' using dk_j^{ACE}, and return the resulting message. If $k_q = q_0$ or $j = i_{q_0}$, set $i_{q_0} \leftarrow j$ and use the oracle $\mathcal{O}_{SD}$ of the IND-CCA experiment to obtain a decryption m of $\tilde{c}$. If $\mathsf{NIZK.Ver}\big(crs^{\mathsf{NIZK}}, x := \big(vk^{\mathsf{Sig}}, \tilde{c}, \big), \pi^{\mathsf{NIZK}}\big) = 1$, return m, otherwise, return $\perp$. In all cases, set $k_q \leftarrow k_q + 1$ at the end.

When $\mathcal{A}_1$ returns (m_0, m_1, i_0, i_1, st), output (m_0, m_1) to the challenger of the IND-CCA experiment to obtain a challenge ciphertext $\tilde{c}^*$. Then run $\pi^{\mathsf{NIZK}} \leftarrow S_2^{\mathsf{NIZK}}\big(crs^{\mathsf{NIZK}}, \tau^{\mathsf{NIZK}}, x := \big(vk^{\mathsf{Sig}}, \tilde{c}^*\big)\big)$, and give st and the ciphertext $c^* := \big(\tilde{c}^*, \pi^{\mathsf{NIZK}}\big)$ to $\mathcal{A}_2$. Emulate the oracles for $\mathcal{A}_2$ as follows.

$\mathcal{O}_G(\cdot,\cdot)$**:** On query $(i, \mathtt{sen})$, if $i \neq i_0$, then generate an encryption key $ek_i^{\mathsf{ACE}} := \big(vk^{\mathsf{Sig}}, ek_i^{\mathsf{sPKE}}, vk_i^{\mathsf{Sig}}, sk_i^{\mathsf{Sig}}, \sigma_i^{\mathsf{Sig}}, crs^{\mathsf{NIZK}}\big)$ as H_2 does, where $\big(ek_i^{\mathsf{sPKE}}, dk_i^{\mathsf{sPKE}}\big)$ is obtained via $\mathcal{O}_G$ and remembered for future queries. If $i = i_0$, replace ek_i^{sPKE} by ek^{sPKE}. On query $(j, \mathtt{rec})$, obtain a decryption key from $\mathcal{O}_G$.

$\mathcal{O}_{SD^*}(\cdot,\cdot)$**:** On query $\big(j, c = \big(\tilde{c}, \pi^{\mathsf{NIZK}}\big)\big)$, run $\mathsf{ACE.DMod}(sp^{\mathsf{ACE}}, c^*, c)$. If the output is 1, return $\mathtt{test}$. Otherwise, if $j \neq i_0$, run $c' \leftarrow \mathsf{ACE.San}(sp^{\mathsf{ACE}}, c)$, generate a decryption key dk_j^{ACE} as above, decrypt c' using dk_j^{ACE}, and return the resulting message. If $j = i_0$, use the oracle $\mathcal{O}_{SD}$ of the IND-CCA experiment to obtain a decryption m of $\tilde{c}$. If $\mathsf{NIZK.Ver}\big(crs^{\mathsf{NIZK}}, x := \big(vk^{\mathsf{Sig}}, \tilde{c}, \big), \pi^{\mathsf{NIZK}}\big) = 1$, return m, otherwise, return $\perp$.

Note that we never query the decryption oracle of the IND-CCA experiment on $\tilde{c}^*$ because we return $\mathtt{test}$ whenever this would be necessary. Denote by Q the event that either $i_{q_0} = i_0$, or $q_0 = 0$ and $\mathcal{A}_1$ does not make the query $(i_0, \mathtt{sen})$ to $\mathcal{O}_G$ and no queries for role i_0 to $\mathcal{O}_{SD}$.

When $\mathcal{A}_2$ returns a bit b' and Q holds, $\mathcal{A}_{\mathsf{sPKE}}$ returns the same bit $b'' \leftarrow b'$, if $\neg Q$, $\mathcal{A}_{\mathsf{sPKE}}$ returns a uniform bit $b'' \twoheadleftarrow \{0,1\}$.

Let $\tilde{b}$ be the bit chosen by the IND-CCA challenger. Note that by our assumption on $\mathcal{A}$, $i_0 = i_1$ and $\mathcal{A}$ does not query $(i_0, \mathtt{rec})$ to $\mathcal{O}_G$, i.e., $i_0 \notin J$, since $P(i_0, i_0) = 1$. Hence, if Q occurs, the view of $\mathcal{A}$ is identical to the one in H_2 with $b = \tilde{b}$. This implies

$$\Pr^{\mathsf{Exp}^{\mathsf{sPKE\text{-}IND\text{-}CCA}}_{\mathsf{sPKE},\mathcal{A}_{\mathsf{sPKE}}}}\big[b'' = \tilde{b} \mid Q\big] = \Pr^{H_2}\big[b' = b\big],$$

and therefore

$$\begin{aligned}\Pr^{\mathsf{Exp}^{\mathsf{sPKE\text{-}IND\text{-}CCA}}_{\mathsf{sPKE},\mathcal{A}_{\mathsf{sPKE}}}}\big[b'' = \tilde{b}\big] &= \Pr^{\mathsf{Exp}^{\mathsf{sPKE\text{-}IND\text{-}CCA}}_{\mathsf{sPKE},\mathcal{A}_{\mathsf{sPKE}}}}\big[b'' = \tilde{b} \mid Q\big] \cdot \Pr^{\mathsf{Exp}^{\mathsf{sPKE\text{-}IND\text{-}CCA}}_{\mathsf{sPKE},\mathcal{A}_{\mathsf{sPKE}}}}\big[Q\big] \\ &\quad + \Pr^{\mathsf{Exp}^{\mathsf{sPKE\text{-}IND\text{-}CCA}}_{\mathsf{sPKE},\mathcal{A}_{\mathsf{sPKE}}}}\big[b'' = \tilde{b} \mid \neg Q\big] \cdot \Pr^{\mathsf{Exp}^{\mathsf{sPKE\text{-}IND\text{-}CCA}}_{\mathsf{sPKE},\mathcal{A}_{\mathsf{sPKE}}}}\big[\neg Q\big] \\ &= \Pr^{H_2}\big[b' = b\big] \cdot \Pr^{\mathsf{Exp}^{\mathsf{sPKE\text{-}IND\text{-}CCA}}_{\mathsf{sPKE},\mathcal{A}_{\mathsf{sPKE}}}}\big[Q\big] + \frac{1}{2}\Pr^{\mathsf{Exp}^{\mathsf{sPKE\text{-}IND\text{-}CCA}}_{\mathsf{sPKE},\mathcal{A}_{\mathsf{sPKE}}}}\big[\neg Q\big].\end{aligned}$$

Using that the probability of Q is $1/(q_S + q_D + 1)$, this yields

$$\begin{aligned}&\Pr^{H_2}\big[b' = b\big] \\ &= \frac{1}{\Pr^{\mathsf{Exp}^{\mathsf{sPKE\text{-}IND\text{-}CCA}}_{\mathsf{sPKE},\mathcal{A}_{\mathsf{sPKE}}}}\big[Q\big]} \cdot \left(\Pr^{\mathsf{Exp}^{\mathsf{sPKE\text{-}IND\text{-}CCA}}_{\mathsf{sPKE},\mathcal{A}_{\mathsf{sPKE}}}}\Big[b'' = \tilde{b}\Big] - \frac{1}{2} \cdot \Big(1 - \Pr^{\mathsf{Exp}^{\mathsf{sPKE\text{-}IND\text{-}CCA}}_{\mathsf{sPKE},\mathcal{A}_{\mathsf{sPKE}}}}\big[Q\big]\Big)\right) \\ &= (q_S + q_D + 1) \cdot \left(\Pr^{\mathsf{Exp}^{\mathsf{sPKE\text{-}IND\text{-}CCA}}_{\mathsf{sPKE},\mathcal{A}_{\mathsf{sPKE}}}}\Big[b'' = \tilde{b}\Big] - \frac{1}{2}\right) + \frac{1}{2}.\end{aligned}$$

Combining this with Claims 1 and 2, we can conclude

$$\begin{aligned}&\mathsf{Adv}^{\mathsf{ACE\text{-}priv\text{-}CCA}}_{\mathsf{ACE},\mathcal{A}} \\ &= 2 \cdot \Pr^{H_0}[b' = b] - 1 \\ &= 2 \cdot \Big(\Pr^{H_0}[b' = b] - \Pr^{H_1}[b' = b] + \Pr^{H_1}[b' = b] - \Pr^{H_2}[b' = b] + \Pr^{H_2}[b' = b]\Big) - 1 \\ &= 2 \cdot \left[\mathsf{Adv}^{\mathsf{PRF}}_{F,\mathcal{A}_{\mathsf{PRF}}} + \mathsf{Adv}^{\mathsf{NIZK\text{-}ZK}}_{\mathsf{NIZK},\mathcal{A}_{\mathsf{ZK}}} + (q_S + q_D + 1)\left[\Pr^{\mathsf{Exp}^{\mathsf{sPKE\text{-}IND\text{-}CCA}}_{\mathsf{sPKE},\mathcal{A}_{\mathsf{sPKE}}}}\Big[b'' = \tilde{b}\Big] - \frac{1}{2}\right] + \frac{1}{2}\right] - 1 \\ &= 2 \cdot \mathsf{Adv}^{\mathsf{PRF}}_{F,\mathcal{A}_{\mathsf{PRF}}} + 2 \cdot \mathsf{Adv}^{\mathsf{NIZK\text{-}ZK}}_{\mathsf{NIZK},\mathcal{A}_{\mathsf{ZK}}} + (q_S + q_D + 1) \cdot \mathsf{Adv}^{\mathsf{sPKE\text{-}IND\text{-}CCA}}_{\mathsf{sPKE},\mathcal{A}_{\mathsf{sPKE}}}.\end{aligned}$$

□

We next consider anonymity, which can be shown similarly. We provide a proof for strong anonymity. Note, however, that for the equality policy, strong anonymity does not provide more guarantees than weak anonymity because anyone who can decrypt directly learns that the sender role is equal to the receiver role.

Theorem 6.4. *Let* ACE *be the scheme from above, let* $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ *be an attacker on the anonymity such that* $\mathcal{A}_1$ *makes at most* q_S *queries of the form* $(\cdot, \mathtt{sen})$ *to the oracle* $\mathcal{O}_G$*, and at most* q_D *queries to* $\mathcal{O}_{SD}$*. Then, there exist probabilistic algorithms* $\mathcal{A}_{\mathsf{PRF}}$*,* $\mathcal{A}_{\mathsf{ZK}}$*, and* $\mathcal{A}_{\mathsf{sPKE}}$ *(which are all roughly as efficient as emulating an execution of* $\mathsf{Exp}^{\mathsf{ACE\text{-}privAnon\text{-}CCA}}_{\mathsf{ACE},\mathcal{A}}$*) such that*

$$\mathsf{Adv}^{\mathsf{ACE\text{-}sAnon\text{-}CCA}}_{\mathsf{ACE},\mathcal{A}} = 2 \cdot \mathsf{Adv}^{\mathsf{PRF}}_{F,\mathcal{A}_{\mathsf{PRF}}} + 2 \cdot \mathsf{Adv}^{\mathsf{NIZK\text{-}ZK}}_{\mathsf{NIZK},\mathcal{A}_{\mathsf{ZK}}} + (q_S + q_D + 1)^2 \cdot \mathsf{Adv}^{\mathsf{sPKE\text{-}IK\text{-}CCA}}_{\mathsf{sPKE},\mathcal{A}_{\mathsf{sPKE}}}.$$

Proof. We assume without loss of generality that $\mathcal{A}$ ensures $m_0 = m_1$ and $P(i_0, j) = P(i_1, j)$ for all $j \in J$, since doing otherwise can only decrease the advantage. Since we have $P(i, j) = 1 \Leftrightarrow i = j$, the latter condition implies that if $i_0 \in J$ or $i_1 \in J$, then $i_0 = i_1$. In case $i_0 = i_1$ and $m_0 = m_1$, $\mathcal{A}$ cannot have positive advantage. Hence, we can further assume without loss of generality that $i_0 \notin J$ and $i_1 \notin J$. As in the proof of Theorem 6.3, let $H_0 := \mathsf{Exp}^{\mathsf{ACE\text{-}privAnon\text{-}CCA}}_{\mathsf{ACE},\mathcal{A}}$, let H_1 be as H_0 where F_K is replaced by a truly uniform random function U, and let H_2 be as H_1, where $crs^{\mathsf{NIZK}} \leftarrow \mathsf{NIZK.Gen}(1^\kappa)$ in $\mathsf{ACE.Setup}$ is replaced by $\big(crs^{\mathsf{NIZK}}, \tau^{\mathsf{NIZK}}\big) \leftarrow S_1^{\mathsf{NIZK}}(1^\kappa)$ and for the generation of the challenge ciphertext c^*, $\pi^{\mathsf{NIZK}} \leftarrow \mathsf{NIZK.Prove}\big(crs^{\mathsf{NIZK}}, x, w\big)$ in $\mathsf{ACE.Enc}$ is replaced by $\pi^{\mathsf{NIZK}} \leftarrow S_2^{\mathsf{NIZK}}\big(crs^{\mathsf{NIZK}}, \tau^{\mathsf{NIZK}}, x\big)$. An identical proof as the one in the proof of Theorem 6.3 shows that there exist $\mathcal{A}_{\mathsf{PRF}}$ and $\mathcal{A}_{\mathsf{ZK}}$ such that

$$\Pr^{H_0}\big[b' = b\big] - \Pr^{H_2}\big[b' = b\big] = \mathsf{Adv}^{\mathsf{PRF}}_{F,\mathcal{A}_{\mathsf{PRF}}} + \mathsf{Adv}^{\mathsf{NIZK\text{-}ZK}}_{\mathsf{NIZK},\mathcal{A}_{\mathsf{ZK}}}.$$

We now transform $\mathcal{A}$ to a winner $\mathcal{A}_{\mathsf{sPKE}}$ for the anonymity game for the scheme sPKE. The reduction is similar to the one in the proof of Theorem 6.3, but $\mathcal{A}_{\mathsf{sPKE}}$ has to guess both i_0 and i_1, which is why we loose the quadratic factor $(q_S + q_D + 1)^2$. On input $(sp^{\mathsf{sPKE}}, ek_0^{\mathsf{sPKE}}, ek_1^{\mathsf{sPKE}})$, $\mathcal{A}_{\mathsf{sPKE}}$ initializes $i_{q_0}, i_{q_1} \leftarrow \bot$, $k_q \leftarrow 1$, chooses $q_0, q_1 \twoheadleftarrow \{0, \ldots, q_S + q_D\}$ uniformly at random, runs $\big(vk^{\mathsf{Sig}}, sk^{\mathsf{Sig}}\big) \leftarrow \mathsf{Sig.Gen}(1^\kappa)$, and $\big(crs^{\mathsf{NIZK}}, \tau^{\mathsf{NIZK}}\big) \leftarrow S_1^{\mathsf{NIZK}}(1^\kappa)$, and gives $sp^{\mathsf{ACE}} := \big(sp^{\mathsf{sPKE}}, vk^{\mathsf{Sig}}, crs^{\mathsf{NIZK}}\big)$ to $\mathcal{A}_1$. It emulates the oracles for $\mathcal{A}_1$ as follows.

$\mathcal{O}_G(\cdot,\cdot)$**:** On query $(i, \mathtt{sen})$, if $k_q \notin \{q_0, q_1\}$ and $i \notin \{i_{q_0}, i_{q_1}\}$, then generate an encryption key $ek_i^{\mathsf{ACE}} := \big(vk^{\mathsf{Sig}}, ek_i^{\mathsf{sPKE}}, vk_i^{\mathsf{Sig}}, sk_i^{\mathsf{Sig}}, \sigma_i^{\mathsf{Sig}}, crs^{\mathsf{NIZK}}\big)$ as H_2 does, where $\big(ek_i^{\mathsf{sPKE}}, dk_i^{\mathsf{sPKE}}\big)$ is obtained via $\mathcal{O}_G$ and remembered for future queries. If $k_q = q_l$ or $i = i_{q_l}$ for some $l \in \{0,1\}$, replace ek_i^{sPKE} by ek_l^{sPKE} (by ek_0^{sPKE} if $q_0 = q_1$) and set $i_{q_l} \leftarrow i$. In both cases, set $k_q \leftarrow k_q + 1$ at the end. On query $(j, \mathtt{rec})$, obtain a decryption key from $\mathcal{O}_G$ and remember it for later.

$\mathcal{O}_{SD}(\cdot,\cdot)$**:** On query $\big(j, c = \big(\tilde{c}, \pi^{\mathsf{NIZK}}\big)\big)$, if $k_q \notin \{q_0, q_1\}$ and $j \notin \{i_{q_0}, i_{q_1}\}$, then execute $c' \leftarrow \mathsf{ACE.San}(sp^{\mathsf{ACE}}, c)$, generate a decryption key dk_j^{ACE} as above, decrypt c' using dk_j^{ACE}, and return the resulting message. If $k_q = q_l$ or $j = i_{q_l}$ for some $l \in \{0,1\}$, set $i_{q_l} \leftarrow j$ and use the oracle $\mathcal{O}_{SD_l}$ of the IK-CCA experiment to obtain a decryption m of $\tilde{c}$. If $\mathsf{NIZK.Ver}\big(crs^{\mathsf{NIZK}}, x := \big(vk^{\mathsf{Sig}}, \tilde{c}, \big), \pi^{\mathsf{NIZK}}\big) = 1$, return m, otherwise, return $\bot$. In all cases, set $k_q \leftarrow k_q + 1$ at the end.

When $\mathcal{A}_1$ returns (m_0, m_1, i_0, i_1, st), $\mathcal{A}_{\mathsf{sPKE}}$ outputs m_0 to the challenger of the anonymity experiment to obtain a challenge ciphertext $\tilde{c}^*$. It then runs $S_2^{\mathsf{NIZK}}\big(crs^{\mathsf{NIZK}}, \tau^{\mathsf{NIZK}}, x := \big(vk^{\mathsf{Sig}}, \tilde{c}^*\big)\big)$, and gives st and the ciphertext $c^* := \big(\tilde{c}^*, \pi^{\mathsf{NIZK}}\big)$ to $\mathcal{A}_2$. It emulates the oracles for $\mathcal{A}_2$ as follows:

$\mathcal{O}_G(\cdot,\cdot)$**:** On query $(i, \mathtt{sen})$, if $i \notin \{i_0, i_1\}$, then generate an encryption key $ek_i^{\mathsf{ACE}} := \big(vk^{\mathsf{Sig}}, ek_i^{\mathsf{sPKE}}, vk_i^{\mathsf{Sig}}, sk_i^{\mathsf{Sig}}, \sigma_i^{\mathsf{Sig}}, crs^{\mathsf{NIZK}}\big)$ as H_2 does, where $\big(ek_i^{\mathsf{sPKE}}, dk_i^{\mathsf{sPKE}}\big)$ is obtained via $\mathcal{O}_G$ and remembered for future queries. If $i = i_{q_l}$ for some $l \in \{0,1\}$, replace ek_i^{sPKE} by ek_l^{sPKE}. On query $(j, \mathtt{rec})$, obtain a decryption key as before.

$\mathcal{O}_{SD^*}(\cdot,\cdot)$**:** On query $\big(j, c = \big(\tilde{c}, \pi^{\mathsf{NIZK}}\big)\big)$, run $\mathsf{ACE.DMod}(sp^{\mathsf{ACE}}, c^*, c)$. If the output is 1, return $\mathtt{test}$. Otherwise, if $j \notin \{i_0, i_1\}$, run $c' \leftarrow \mathsf{ACE.San}(sp^{\mathsf{ACE}}, c)$, generate a decryption key dk_j^{ACE} as above, decrypt c' using dk_j^{ACE}, and return the resulting message. If $j = i_{q_l}$ for some $l \in \{0,1\}$, use the oracle $\mathcal{O}_{SD_l}$ of the IK-CCA experiment to obtain a decryption m of $\tilde{c}$. If $\mathsf{NIZK.Ver}\big(crs^{\mathsf{NIZK}}, x := \big(vk^{\mathsf{Sig}}, \tilde{c}, \big), \pi^{\mathsf{NIZK}}\big) = 1$, return m, otherwise, return $\bot$.

Note that $\mathcal{A}_{\mathsf{sPKE}}$ never queries any of the decryption oracles of the IK-CCA experiment on $\tilde{c}^*$ because we return $\mathtt{test}$ whenever this would be necessary. Denote by Q the event that for all

$l \in \{0,1\}$ we have either $i_{q_l} = i_l$, or $q_l = 0$ and $\mathcal{A}_1$ does not make the query $(i_l, \mathtt{sen})$ to $\mathcal{O}_G$ and no queries for role i_l to $\mathcal{O}_{SD}$. When $\mathcal{A}_2$ returns a bit b' and Q holds, $\mathcal{A}_{\mathsf{sPKE}}$ returns the same bit $b'' \leftarrow b'$, if $\neg Q$, $\mathcal{A}_{\mathsf{sPKE}}$ returns a uniform bit $b'' \twoheadleftarrow \{0,1\}$.

Let $\tilde{b}$ be the bit chosen by the IK-CCA experiment. Note that if Q occurs, the view of $\mathcal{A}$ is identical to the one in H_2 with $b = \tilde{b}$. This implies

$$\Pr^{\mathsf{Exp}^{\mathsf{sPKE\text{-}IK\text{-}CCA}}_{\mathsf{sPKE},\mathcal{A}_{\mathsf{sPKE}}}}\big[b'' = \tilde{b} \mid Q\big] = \Pr^{H_2}\big[b' = b\big].$$

Using that the probability of Q is $1/(q_S + q_D + 1)^2$, it follows as in the proof of Theorem 6.3 that

$$\mathsf{Adv}^{\mathsf{ACE\text{-}sAnon\text{-}CCA}}_{\mathsf{ACE},\mathcal{A}} = 2 \cdot \mathsf{Adv}^{\mathsf{PRF}}_{F,\mathcal{A}_{\mathsf{PRF}}} + 2 \cdot \mathsf{Adv}^{\mathsf{NIZK\text{-}ZK}}_{\mathsf{NIZK},\mathcal{A}_{\mathsf{ZK}}} + (q_S + q_D + 1)^2 \cdot \mathsf{Adv}^{\mathsf{sPKE\text{-}IK\text{-}CCA}}_{\mathsf{sPKE},\mathcal{A}_{\mathsf{sPKE}}}. \qquad \square$$

We next prove the sanitization security of our scheme.

Theorem 6.5. *Let* ACE *be the scheme from above, let* $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ *be an attacker on the sanitization security such that* $\mathcal{A}_1$ *makes at most* q_{S_1} *queries of the form* $(\cdot, \mathtt{sen})$ *and at most* q_{R_1} *queries of the form* $(\cdot, \mathtt{rec})$ *to the oracle* $\mathcal{O}_G$*, and at most* q_{D_1} *queries to* $\mathcal{O}_{SD}$*, and* $\mathcal{A}_2$ *makes at most* q_{R_2} *queries of the form* $(\cdot, \mathtt{rec})$ *to the oracle* $\mathcal{O}_G$*. Then, there exist probabilistic algorithms* $\mathcal{A}_{\mathsf{PRF}}$*,* $\mathcal{A}_{\mathsf{ZK}_1}$*,* $\mathcal{A}_{\mathsf{ZK}_2}$*,* $\mathcal{A}_{\mathsf{Sig}}$*,* $\mathcal{A}_{\mathsf{sPKE}}$*, and* $\mathcal{A}_{\mathsf{rob}}$ *(which are all roughly as efficient as emulating an execution of* $\mathsf{Exp}^{\mathsf{ACE\text{-}san\text{-}CCA}}_{\mathsf{ACE},\mathcal{A}}$*) such that*

$$\begin{aligned}\mathsf{Adv}^{\mathsf{ACE\text{-}san\text{-}CCA}}_{\mathsf{ACE},\mathcal{A}} \le{}& 2 \cdot \mathsf{Adv}^{\mathsf{PRF}}_{F,\mathcal{A}_{\mathsf{PRF}}} + 2 \cdot \mathsf{Adv}^{\mathsf{NIZK\text{-}ext_1}}_{\mathsf{NIZK},\mathcal{A}_{\mathsf{ZK}_1}} + 4 \cdot \mathsf{Adv}^{\mathsf{NIZK\text{-}ext_2}}_{\mathsf{NIZK},\mathcal{A}_{\mathsf{ZK}_2}} + 4 \cdot \mathsf{Adv}^{\mathsf{Sig\text{-}EUF\text{-}CMA}}_{\mathsf{Sig},\mathcal{A}_{\mathsf{Sig}}} \\ &+ (q_{S_1} + q_{R_1} + q_{D_1})^2 \cdot \mathsf{Adv}^{\mathsf{sPKE\text{-}san\text{-}CCA}}_{\mathsf{sPKE},\mathcal{A}_{\mathsf{sPKE}}} + 4(q_{R_1} + q_{R_2}) \cdot \mathsf{Adv}^{\mathsf{sPKE\text{-}USROB}}_{\mathsf{sPKE},\mathcal{A}_{\mathsf{rob}}}.\end{aligned}$$

Proof. Let $H_0 := \mathsf{Exp}^{\mathsf{ACE\text{-}san\text{-}CCA}}_{\mathsf{ACE},\mathcal{A}}$, let H_1 be as H_0 where F_K is replaced by a truly uniform random function U, and let H_2 be as H_1, where $crs^{\mathsf{NIZK}} \leftarrow \mathsf{NIZK.Gen}(1^\kappa)$ in $\mathsf{ACE.Setup}$ is replaced by $\big(crs^{\mathsf{NIZK}}, \xi^{\mathsf{NIZK}}\big) \leftarrow E_1^{\mathsf{NIZK}}(1^\kappa)$. Let W_{ACE} denote the event that $\mathcal{A}$ wins, i.e.,

$$W_{\mathsf{ACE}} := \big[b' = b \ \wedge\ c_0' \neq \bot \neq c_1' \ \wedge\ \forall j \in J\ m_{0,j} = m_{1,j}\big].$$

Similarly as in the proof of Theorem 6.3, it can be shown that there exist $\mathcal{A}_{\mathsf{PRF}}$ and $\mathcal{A}_{\mathsf{ZK}_1}$ such that

$$\Pr^{H_0}[W_{\mathsf{ACE}}] - \Pr^{H_2}[W_{\mathsf{ACE}}] = \mathsf{Adv}^{\mathsf{PRF}}_{F,\mathcal{A}_{\mathsf{PRF}}} + \mathsf{Adv}^{\mathsf{NIZK\text{-}ext_1}}_{\mathsf{NIZK},\mathcal{A}_{\mathsf{ZK}_1}}. \tag{2}$$

Let H_3 be identical to H_2 except that after $\mathcal{A}_1$ returns $\big(c_0 = \big(\tilde{c}_0, \pi_0^{\mathsf{NIZK}}\big), c_1 = \big(\tilde{c}_1, \pi_1^{\mathsf{NIZK}}\big), st\big)$, H_3 executes for $\tilde{b} \in \{0,1\}$

$$w_{\tilde{b}} := \big(ek^{\mathsf{sPKE}}_{i_{\tilde{b}}}, m_{\tilde{b}}, r_{\tilde{b}}, vk^{\mathsf{Sig}}_{i_{\tilde{b}}}, \sigma^{\mathsf{Sig}}_{i_{\tilde{b}}}, \sigma^{\mathsf{Sig}}_{c_{\tilde{b}}}\big) \leftarrow E_2^{\mathsf{NIZK}}\big(crs^{\mathsf{NIZK}}, \xi^{\mathsf{NIZK}}, x_{\tilde{b}} := \big(vk^{\mathsf{Sig}}, \tilde{c}_{\tilde{b}}\big), \pi^{\mathsf{NIZK}}_{\tilde{b}}\big).$$

We clearly have

$$\Pr^{H_3}[W_{\mathsf{ACE}}] = \Pr^{H_2}[W_{\mathsf{ACE}}]. \tag{3}$$

Let $V_{\tilde{b}} := \big[\mathsf{NIZK.Ver}\big(crs^{\mathsf{NIZK}}, x_{\tilde{b}}, \pi^{\mathsf{NIZK}}_{\tilde{b}}\big) = 1\big]$ and let B_E be the event that (at least) one of the extractions fail, i.e.,

$$B_E := \big[(V_0 \ \wedge\ (x_0, w_0) \notin R) \ \vee\ (V_1 \ \wedge\ (x_1, w_1) \notin R)\big].$$

If B_E occurs, the knowledge extraction of NIZK is broken. To prove this, we define $\mathcal{A}_{\mathsf{ZK}_2}$ as follows. On input crs^{NIZK}, it emulates an execution of H_3, where in $\mathsf{ACE.Setup}$, crs^{NIZK} is used instead of generating it. When $\mathcal{A}_1$ returns (c_0, c_1, st), $\mathcal{A}_{\mathsf{ZK}_2}$ flips a coin $\tilde{b} \twoheadleftarrow \{0,1\}$ and returns $\big(x_{\tilde{b}}, \pi_{\tilde{b}}^{\mathsf{NIZK}}\big)$. If the $\tilde{b}$'s extraction fails, $\mathcal{A}_{\mathsf{ZK}_2}$ wins the extraction game. Hence,

$$\Pr^{H_3}[B_E] \leq 2 \cdot \mathsf{Adv}_{\mathsf{NIZK},\mathcal{A}_{\mathsf{ZK}_2}}^{\mathsf{NIZK\text{-}ext}_2}. \tag{4}$$

For $\tilde{b} \in \{0,1\}$, let $B_{S,\tilde{b}}$ be the event that $(x_{\tilde{b}}, w_{\tilde{b}}) \in R$ and $ek_{i_{\tilde{b}}}^{\mathsf{sPKE}}$ is not contained in an answer from $\mathcal{O}_G$ to $\mathcal{A}_1$, and let B_S be the union of $B_{S,0}$ and $B_{S,1}$. We next show that if B_S occurs, the adversary found a forgery for the signature scheme.

Claim 1. *There exists a probabilistic algorithm $\mathcal{A}_{\mathsf{Sig}}$ such that*

$$\Pr^{H_3}[B_S] \leq 2 \cdot \mathsf{Adv}_{\mathsf{Sig},\mathcal{A}_{\mathsf{Sig}}}^{\mathsf{Sig\text{-}EUF\text{-}CMA}}. \tag{5}$$

Proof of claim. On input vk^{Sig}, $\mathcal{A}_{\mathsf{Sig}}$ emulate an execution of H_3, where vk^{Sig} is used in msk^{ACE} and sp^{ACE}. Queries $(i, \mathtt{sen})$ by $\mathcal{A}_1$ to the oracle $\mathcal{O}_G$ are answered by executing $\mathsf{ACE.Gen}$ (with F_K replaced by U) where σ_i^{Sig} is generated using the signing oracle of $\mathsf{Exp}_{\mathcal{E},\mathcal{A}}^{\mathsf{Sig\text{-}EUF\text{-}CMA}}$. After extracting w_0 and w_1, $\mathcal{A}_{\mathsf{Sig}}$ flips a coin $\tilde{b} \twoheadleftarrow \{0,1\}$ and returns $\Big(\big[ek_{i_{\tilde{b}}}^{\mathsf{sPKE}}, vk_{i_{\tilde{b}}}^{\mathsf{Sig}}\big], \sigma_{i_{\tilde{b}}}^{\mathsf{Sig}}\Big)$. If $B_{S,\tilde{b}}$ occurs, $\big[ek_{i_{\tilde{b}}}^{\mathsf{sPKE}}, vk_{i_{\tilde{b}}}^{\mathsf{Sig}}\big]$ was not queried to the signing oracle and $(x_{\tilde{b}}, w_{\tilde{b}}) \in R$. The latter implies that $\sigma_{i_{\tilde{b}}}^{\mathsf{Sig}}$ is a valid signature and hence $\mathcal{A}_{\mathsf{Sig}}$ successfully forged a signature. We conclude

$$\Pr^{H_3}[B_S] \leq 2 \cdot \left(\frac{1}{2}\Pr^{H_3}[B_{S,0}] + \frac{1}{2}\Pr^{H_3}[B_{S,1}]\right) = 2 \cdot \mathsf{Adv}_{\mathsf{Sig},\mathcal{A}_{\mathsf{Sig}}}^{\mathsf{Sig\text{-}EUF\text{-}CMA}}. \qquad \Diamond$$

Let H_4 be identical to H_3 with the difference that we replace for $k \in \{0,1\}$ and $j \in J$, $m_{k,j} \leftarrow \mathsf{ACE.Dec}\big(\mathsf{ACE.Gen}(msk, j, \mathtt{rec}), c'_k\big)$ by

$$m_{k,j} \leftarrow \begin{cases} m_k, & ek_j^{\mathsf{sPKE}} = ek_{i_k}^{\mathsf{sPKE}} \text{ for } \big(ek_j^{\mathsf{sPKE}}, dk_j^{\mathsf{sPKE}}\big) = \mathsf{sPKE.Gen}\big(msk^{\mathsf{sPKE}}; U([j,0])\big), \\ \bot, & \text{else}, \end{cases} \tag{6}$$

where $ek_{i_k}^{\mathsf{sPKE}}$ are the extracted keys. Note that if V_k, $\neg B_E$, and $\neg B_S$ occur, we have $c'_k = \mathsf{San}(sp^{\mathsf{sPKE}}, \tilde{c}_k)$, $\tilde{c}_k = \mathsf{sPKE.Enc}\big(ek_{i_k}^{\mathsf{sPKE}}, m_k; r_k\big)$, and $ek_{i_k}^{\mathsf{sPKE}}$ was generated by $\mathcal{O}_G$. Hence, for $j \in J$ with $ek_j^{\mathsf{sPKE}} = ek_{i_k}^{\mathsf{sPKE}}$, we have by the correctness of the sPKE scheme that $\mathsf{ACE.Dec}\big(\mathsf{ACE.Gen}(msk, j, \mathtt{rec}), c'_k\big) = m_k$, i.e., $m_{k,j} = m_k$ in both H_3 and H_4. For other $j \in J$, decryption only yields a message different from $\bot$ if robustness of the sPKE scheme is violated. Since $|J| \leq q_{R_1} + q_{R_2}$, this implies for $V \coloneqq V_0 \cap V_1$,

$$\Pr^{H_3}[W_{\mathsf{ACE}} \mid V \cap \neg B_E \cap \neg B_S] - \Pr^{H_4}[W_{\mathsf{ACE}} \mid V \cap \neg B_E \cap \neg B_S] \leq 2(q_{R_1} + q_{R_2})\mathsf{Adv}_{\mathsf{sPKE},\mathcal{A}_{\mathsf{rob}}}^{\mathsf{sPKE\text{-}USROB}}, \tag{7}$$

where $\mathcal{A}_{\mathsf{rob}}$ emulates the experiment and outputs $\tilde{c}_k$ for a uniformly chosen $k \in \{0,1\}$, i such that the i-th query to the key-generation oracle yields $ek_{i_k}^{\mathsf{sPKE}}$, and a uniformly chosen j.[7]

We finally construct an adversary $\mathcal{A}_{\mathsf{sPKE}}$ against the sanitization security of sPKE. On input $(sp^{\mathsf{sPKE}}, ek_0^{\mathsf{sPKE}}, ek_1^{\mathsf{sPKE}})$, $\mathcal{A}_{\mathsf{sPKE}}$ initializes $i_{q_0}, i_{q_1} \leftarrow \bot$, $k_q \leftarrow 1$, chooses distinct $q_0, q_1 \twoheadleftarrow \{1, \ldots, q_{S_1} + q_{R_1} + q_{D_1}\}$ uniformly at random, executes $\big(vk^{\mathsf{Sig}}, sk^{\mathsf{Sig}}\big) \leftarrow \mathsf{Sig.Gen}(1^\kappa)$, and $\big(crs^{\mathsf{NIZK}}, \xi^{\mathsf{NIZK}}\big) \leftarrow E_1^{\mathsf{NIZK}}(1^\kappa)$, and gives $sp^{\mathsf{ACE}} \coloneqq \big(sp^{\mathsf{sPKE}}, vk^{\mathsf{Sig}}, crs^{\mathsf{NIZK}}\big)$ to $\mathcal{A}_1$. It emulates the oracles for $\mathcal{A}_1$ as follows.

[7]Note that robustness is only defined for encryption and decryption keys generated by $\mathsf{sPKE.Gen}$. Hence, it is important to also condition on $\neg B_S$.

$\mathcal{O}_G(\cdot,\cdot)$**:** On query $(i, \texttt{sen})$, if $k_q \notin \{q_0, q_1\}$ and $i \notin \{i_{q_0}, i_{q_1}\}$, generate an encryption key $\bigl(vk^{\mathsf{Sig}}, ek_i^{\mathsf{sPKE}}, sk_i^{\mathsf{Sig}}, \sigma_i^{\mathsf{Sig}}, crs^{\mathsf{NIZK}}\bigr)$ as H_4, where $\bigl(ek_i^{\mathsf{sPKE}}, dk_i^{\mathsf{sPKE}}\bigr)$ is obtained via $\mathcal{O}_G$ and remembered for future queries. If $k_q = q_l$ or $i = i_{q_l}$ for some $l \in \{0,1\}$, replace ek_i^{sPKE} by ek_l^{sPKE} and set $i_{q_l} \leftarrow i$. In both cases, set $k_q \leftarrow k_q + 1$ at the end.

On query $(j, \texttt{rec})$, if $k_q \notin \{q_0, q_1\}$ and $j \notin \{i_{q_0}, i_{q_1}\}$, obtain a decryption key from $\mathcal{O}_G$, remember it, and set $k_q \leftarrow k_q + 1$. If $k_q = q_l$ or $j = i_{q_l}$ for some $l \in \{0,1\}$, then return $\perp$ and set $k_q \leftarrow k_q + 1$.

$\mathcal{O}_{SD}(\cdot,\cdot)$**:** On query $\bigl(j, c = \bigl(\tilde{c}, \pi^{\mathsf{NIZK}}\bigr)\bigr)$, if $k_q \notin \{q_0, q_1\}$ and $j \notin \{i_{q_0}, i_{q_1}\}$, then execute $c' \leftarrow \mathsf{ACE.San}(sp^{\mathsf{ACE}}, c)$, generate a decryption key dk_j^{ACE} as above, decrypt c' using dk_j^{ACE}, and return the resulting message. If $k_q = q_l$ or $j = i_{q_l}$ for some $l \in \{0,1\}$, set $i_{q_l} \leftarrow j$, if $\mathsf{NIZK.Ver}\bigl(crs^{\mathsf{NIZK}}, x := \bigl(vk^{\mathsf{Sig}}, \tilde{c},\bigr), \pi^{\mathsf{NIZK}}\bigr) = 0$, return $\perp$, otherwise, use the oracle $\mathcal{O}_{SD_l}$ of the sPKE-sanitization experiment to obtain a decryption of $\tilde{c}$ and return it. In all cases, set $k_q \leftarrow k_q + 1$ at the end.

When $\mathcal{A}_1$ returns $\bigl(c_0 = \bigl(\tilde{c}_0, \pi_0^{\mathsf{NIZK}}\bigr), c_1 = \bigl(\tilde{c}_1, \pi_1^{\mathsf{NIZK}}\bigr), st\bigr)$, $\mathcal{A}_{\mathsf{sPKE}}$ verifies the proofs π_0^{NIZK} and π_1^{NIZK} and extracts the witnesses to check the events V, B_E, and B_S. Denote by Q the event that $ek_{i_0}^{\mathsf{sPKE}}, ek_{i_1}^{\mathsf{sPKE}} \in \{ek_0^{\mathsf{sPKE}}, ek_1^{\mathsf{sPKE}}\}$, where $ek_{i_0}^{\mathsf{sPKE}}, ek_{i_1}^{\mathsf{sPKE}}$ are the extracted keys. Note that if V, $\neg B_E$, and $\neg B_S$ occur, both $ek_{i_0}^{\mathsf{sPKE}}$ and $ek_{i_1}^{\mathsf{sPKE}}$ have been returned by $\mathcal{O}_G$ to $\mathcal{A}_1$. This implies

$$\Pr^{\mathsf{Exp}^{\mathsf{sPKE\text{-}san\text{-}CCA}}_{\mathsf{sPKE},\mathcal{A}_{\mathsf{sPKE}}}}[Q \mid V \cap \neg B_E \cap \neg B_S] \geq 1/(q_{S_1} + q_{R_1} + q_{D_1})^2. \tag{8}$$

If Q, V, $\neg B_E$, and $\neg B_S$ occur, $\mathcal{A}_{\mathsf{sPKE}}$ returns $(\tilde{c}_0, \tilde{c}_1)$ to the challenger of the sPKE-sanitization experiment to obtain the sanitized ciphertext $c'_{\tilde{b}}$. It then gives $\bigl(st, c'_{\tilde{b}}\bigr)$ to $\mathcal{A}_2$ and emulates the oracles as above. After $\mathcal{A}_2$ returned the bit b', $\mathcal{A}_{\mathsf{sPKE}}$ returns $b'' \leftarrow b'$. If $Q \cap V \cap \neg B_E \cap \neg B_S$ does not occur, $\mathcal{A}_{\mathsf{sPKE}}$ runs $\bar{c} \leftarrow \mathsf{sPKE.Enc}(ek_0^{\mathsf{sPKE}}, \bar{m})$ for an arbitrary fixed message $\bar{m}$ and returns $(c_0 := \bar{c}, c_1 := \bar{c})$ to the challenger. After receiving back a sanitized ciphertext $c'_{\tilde{b}}$, it returns a uniform bit $b'' \twoheadleftarrow \{0,1\}$.

Let W_{sPKE} be the event that $\mathcal{A}_{\mathsf{sPKE}}$ wins, i.e.,

$$W_{\mathsf{sPKE}} := \bigl[b'' = \tilde{b} \;\wedge\; \exists j, j' \in \{0,1\}\; m_{0,j}^{\mathsf{sPKE}} \neq \perp \neq m_{1,j'}^{\mathsf{sPKE}}))\bigr],$$

where the messages refer to the ones generated by $\mathsf{Exp}^{\mathsf{sPKE\text{-}san\text{-}CCA}}_{\mathsf{sPKE},\mathcal{A}_{\mathsf{sPKE}}}$. Note that if $Q \cap V \cap \neg B_E \cap \neg B_S$ does not occur, we have $m_{0,0}^{\mathsf{sPKE}} = m_{1,0}^{\mathsf{sPKE}} = \bar{m} \neq \perp$ by the correctness of sPKE, and thus

$$\Pr^{\mathsf{Exp}^{\mathsf{sPKE\text{-}san\text{-}CCA}}_{\mathsf{sPKE},\mathcal{A}_{\mathsf{sPKE}}}}[W_{\mathsf{sPKE}} \mid \neg(Q \cap V \cap \neg B_E \cap \neg B_S)] = \frac{1}{2}. \tag{9}$$

Next consider the case that $Q \cap V \cap \neg B_E \cap \neg B_S$ occurs. In this case, the view of $\mathcal{A}$ is identical to the one in H_4 with $b = \tilde{b}$, as long as the emulated $\mathcal{O}_G$ never returns $\perp$. Moreover, if $\mathcal{A}$ wins, we have $m_{0,j}^{H_4} = m_{1,j}^{H_4} = \perp$ for all $j \in J^{H_4}$, where the messages here refer to the ones in H_4, generated according to (6), and J^{H_4} is the set of all j such that $\mathcal{A}_1$ or $\mathcal{A}_2$ issued the query $(j, \texttt{rec})$ to the oracle $\mathcal{O}_G$. Therefore, $\mathcal{O}_G$ is never gets a query for which it returns $\perp$ in this case. The event $Q \cap V \cap \neg B_E$ implies that the ciphertexts are encryptions of some message under ek_0^{sPKE} or ek_1^{sPKE}. Correctness of sPKE now implies that $m_{0,0}^{\mathsf{sPKE}} \neq \perp \neq m_{1,0}^{\mathsf{sPKE}}$, i.e., the winning condition for $\mathcal{A}_{\mathsf{sPKE}}$ is satisfied. We can conclude that

$$\Pr^{\mathsf{Exp}^{\mathsf{sPKE\text{-}san\text{-}CCA}}_{\mathsf{sPKE},\mathcal{A}_{\mathsf{sPKE}}}}[W_{\mathsf{sPKE}} \mid Q \cap V \cap \neg B_E \cap \neg B_S] \geq \Pr^{H_4}[W_{\mathsf{ACE}} \mid V \cap \neg B_E \cap \neg B_S]. \tag{10}$$

Let

$$p_G := \Pr^{\mathsf{Exp}^{\mathsf{sPKE\text{-}san\text{-}CCA}}_{\mathsf{sPKE},\mathcal{A}_{\mathsf{sPKE}}}}[Q \cap V \cap \neg B_E \cap \neg B_S].$$

Putting our results together, we obtain

$$\begin{aligned}\Pr^{\mathsf{Exp}^{\mathsf{sPKE\text{-}san\text{-}CCA}}_{\mathsf{sPKE},\mathcal{A}_{\mathsf{sPKE}}}}[W_{\mathsf{sPKE}}] &= \Pr^{\mathsf{Exp}^{\mathsf{sPKE\text{-}san\text{-}CCA}}_{\mathsf{sPKE},\mathcal{A}_{\mathsf{sPKE}}}}[W_{\mathsf{sPKE}} \mid Q \cap V \cap \neg B_E \cap \neg B_S] \cdot p_G \\ &\quad + \Pr^{\mathsf{Exp}^{\mathsf{sPKE\text{-}san\text{-}CCA}}_{\mathsf{sPKE},\mathcal{A}_{\mathsf{sPKE}}}}[W_{\mathsf{sPKE}} \mid \neg(Q \cap V \cap \neg B_E \cap \neg B_S)] \cdot (1 - p_G) \\ &\overset{(9)}{=} \Pr^{\mathsf{Exp}^{\mathsf{sPKE\text{-}san\text{-}CCA}}_{\mathsf{sPKE},\mathcal{A}_{\mathsf{sPKE}}}}[W_{\mathsf{sPKE}} \mid Q \cap V \cap \neg B_E \cap \neg B_S] \cdot p_G + \frac{1}{2}(1 - p_G).\end{aligned}$$

This implies

$$\begin{aligned}\Pr^{\mathsf{Exp}^{\mathsf{sPKE\text{-}san\text{-}CCA}}_{\mathsf{sPKE},\mathcal{A}_{\mathsf{sPKE}}}}[W_{\mathsf{sPKE}} \mid Q \cap V \cap \neg B_E \cap \neg B_S] &= \frac{1}{p_G}\left[\Pr^{\mathsf{Exp}^{\mathsf{sPKE\text{-}san\text{-}CCA}}_{\mathsf{sPKE},\mathcal{A}_{\mathsf{sPKE}}}}[W_{\mathsf{sPKE}}] - \frac{1}{2}(1 - p_G)\right] \\ &= \frac{1}{2\,p_G} \cdot \mathsf{Adv}^{\mathsf{sPKE\text{-}san\text{-}CCA}}_{\mathsf{sPKE},\mathcal{A}_{\mathsf{sPKE}}} + \frac{1}{2}.\end{aligned} \tag{11}$$

Furthermore,

$$\mathsf{Adv}^{\mathsf{ACE\text{-}san\text{-}CCA}}_{\mathsf{ACE},\mathcal{A}} = 2 \cdot \Pr^{H_0}[W_{\mathsf{ACE}}] - 1 \overset{(2),(3)}{=} 2 \cdot \left(\mathsf{Adv}^{\mathsf{PRF}}_{F,\mathcal{A}_{\mathsf{PRF}}} + \mathsf{Adv}^{\mathsf{NIZK\text{-}ext_1}}_{\mathsf{NIZK},\mathcal{A}_{\mathsf{ZK}_1}} + \Pr^{H_3}[W_{\mathsf{ACE}}]\right) - 1.$$

Since B_E, $\neg B_E \cap B_S$, and $\neg B_E \cap \neg B_S$ partition the sample space, the law of total probability implies

$$\begin{aligned}\Pr^{H_3}[W_{\mathsf{ACE}}] &= \Pr^{H_3}[W_{\mathsf{ACE}} \cap B_E] + \Pr^{H_3}[W_{\mathsf{ACE}} \cap \neg B_E \cap B_S] \\ &\qquad + \Pr^{H_3}[W_{\mathsf{ACE}} \cap \neg B_E \cap \neg B_S] \\ &\le \Pr^{H_3}[B_E] + \Pr^{H_3}[B_S] + \Pr^{H_3}[W_{\mathsf{ACE}} \cap \neg B_E \cap \neg B_S] \\ &\overset{(4),(5)}{\le} 2 \cdot \mathsf{Adv}^{\mathsf{NIZK\text{-}ext_2}}_{\mathsf{NIZK},\mathcal{A}_{\mathsf{ZK}_2}} + 2 \cdot \mathsf{Adv}^{\mathsf{Sig\text{-}EUF\text{-}CMA}}_{\mathsf{Sig},\mathcal{A}_{\mathsf{Sig}}} + \Pr^{H_3}[W_{\mathsf{ACE}} \cap \neg B_E \cap \neg B_S].\end{aligned}$$

Note that W_{ACE} implies $c_0' \neq \perp \neq c_1'$ and thus also V because if the verification fails, $\mathsf{ACE.San}$ returns $\perp$. Hence,

$$\begin{aligned}&\Pr^{H_3}[W_{\mathsf{ACE}} \cap \neg B_E \cap \neg B_S] = \Pr^{H_3}[W_{\mathsf{ACE}} \cap V \cap \neg B_E \cap \neg B_S] \\ &= \Pr^{H_3}[W_{\mathsf{ACE}} \mid V \cap \neg B_E \cap \neg B_S] \cdot \Pr^{H_3}[V \cap \neg B_E \cap \neg B_S] \\ &\overset{(7)}{\le} \Big(\underbrace{\Pr^{H_4}[W_{\mathsf{ACE}} \mid V \cap \neg B_E \cap \neg B_S]}_{\overset{(10)}{\le} \Pr^{\mathsf{Exp}^{\mathsf{sPKE\text{-}san\text{-}CCA}}_{\mathsf{sPKE},\mathcal{A}_{\mathsf{sPKE}}}}[W_{\mathsf{sPKE}} \mid Q \cap V \cap \neg B_E \cap \neg B_S]} + 2(q_{R_1} + q_{R_2}) \cdot \mathsf{Adv}^{\mathsf{sPKE\text{-}USROB}}_{\mathsf{sPKE},\mathcal{A}_{\mathsf{rob}}}\Big) \cdot \Pr^{H_3}[V \cap \neg B_E \cap \neg B_S] \\ &\overset{(11)}{\le} \left(\frac{1}{2\,p_G} \cdot \mathsf{Adv}^{\mathsf{sPKE\text{-}san\text{-}CCA}}_{\mathsf{sPKE},\mathcal{A}_{\mathsf{sPKE}}} + \frac{1}{2} + 2(q_{R_1} + q_{R_2}) \cdot \mathsf{Adv}^{\mathsf{sPKE\text{-}USROB}}_{\mathsf{sPKE},\mathcal{A}_{\mathsf{rob}}}\right) \cdot \Pr^{H_3}[V \cap \neg B_E \cap \neg B_S].\end{aligned}$$

Since $\Pr^{H_3}[V \cap \neg B_E \cap \neg B_S] = \Pr^{\mathsf{Exp}^{\mathsf{sPKE\text{-}san\text{-}CCA}}_{\mathsf{sPKE},\mathcal{A}_{\mathsf{sPKE}}}}[V \cap \neg B_E \cap \neg B_S]$, we have

$$\begin{aligned}\frac{\Pr^{H_3}[V \cap \neg B_E \cap \neg B_S]}{p_G} &= \frac{\Pr^{\mathsf{Exp}^{\mathsf{sPKE\text{-}san\text{-}CCA}}_{\mathsf{sPKE},\mathcal{A}_{\mathsf{sPKE}}}}[V \cap \neg B_E \cap \neg B_S]}{\Pr^{\mathsf{Exp}^{\mathsf{sPKE\text{-}san\text{-}CCA}}_{\mathsf{sPKE},\mathcal{A}_{\mathsf{sPKE}}}}[Q \cap V \cap \neg B_E \cap \neg B_S]} \\ &= 1\Big/\Big(\Pr^{\mathsf{Exp}^{\mathsf{sPKE\text{-}san\text{-}CCA}}_{\mathsf{sPKE},\mathcal{A}_{\mathsf{sPKE}}}}[Q \mid V \cap \neg B_E \cap \neg B_S]\Big) \\ &\overset{(8)}{\le} (q_{S_1} + q_{R_1} + q_{D_1})^2.\end{aligned}$$

Therefore,

$$\Pr^{H_3}[W_{\mathsf{ACE}} \cap \neg B_E \cap \neg B_S] \le \frac{1}{2} \cdot (q_{S_1} + q_{R_1} + q_{D_1})^2 \cdot \mathsf{Adv}^{\mathsf{sPKE\text{-}san\text{-}CCA}}_{\mathsf{sPKE},\mathcal{A}_{\mathsf{sPKE}}} + \frac{1}{2} + 2(q_{R_1} + q_{R_2}) \cdot \mathsf{Adv}^{\mathsf{sPKE\text{-}USROB}}_{\mathsf{sPKE},\mathcal{A}_{\mathsf{rob}}}.$$

This implies

$$\begin{aligned}\mathsf{Adv}^{\mathsf{ACE\text{-}san\text{-}CCA}}_{\mathsf{ACE},\mathcal{A}} \le 2 \cdot \mathsf{Adv}^{\mathsf{PRF}}_{F,\mathcal{A}_{\mathsf{PRF}}} + 2 \cdot \mathsf{Adv}^{\mathsf{NIZK\text{-}ext_1}}_{\mathsf{NIZK},\mathcal{A}_{\mathsf{ZK}_1}} + 4 \cdot \mathsf{Adv}^{\mathsf{NIZK\text{-}ext_2}}_{\mathsf{NIZK},\mathcal{A}_{\mathsf{ZK}_2}} + 4 \cdot \mathsf{Adv}^{\mathsf{Sig\text{-}EUF\text{-}CMA}}_{\mathsf{Sig},\mathcal{A}_{\mathsf{Sig}}} \\ + (q_{S_1} + q_{R_1} + q_{D_1})^2 \cdot \mathsf{Adv}^{\mathsf{sPKE\text{-}san\text{-}CCA}}_{\mathsf{sPKE},\mathcal{A}_{\mathsf{sPKE}}} + 4(q_{R_1} + q_{R_2}) \cdot \mathsf{Adv}^{\mathsf{sPKE\text{-}USROB}}_{\mathsf{sPKE},\mathcal{A}_{\mathsf{rob}}}\end{aligned}$$

and concludes the proof. □

We next prove ciphertext unpredictability, which directly follows from ciphertext unpredictability of the underlying sPKE scheme.

Theorem 6.6. *Let* ACE *be the scheme from above and let $\mathcal{A}$ be an attacker on the ciphertext unpredictability that makes at most q queries to the oracle $\mathcal{O}_G$. Then, there exist probabilistic algorithms $\mathcal{A}_{\mathsf{PRF}}$ and $\mathcal{A}_{\mathsf{sPKE}}$ (which are both roughly as efficient as emulating an execution of $\mathsf{Exp}^{\mathsf{ACE\text{-}ctxt\text{-}unpred}}_{\mathsf{ACE},\mathcal{A}}$) such that*

$$\mathsf{Adv}^{\mathsf{ACE\text{-}ctxt\text{-}unpred}}_{\mathsf{ACE},\mathcal{A}} \le \mathsf{Adv}^{\mathsf{PRF}}_{F,\mathcal{A}_{\mathsf{PRF}}} + (q+1) \cdot \mathsf{Adv}^{\mathsf{sPKE\text{-}ctxt\text{-}unpred}}_{\mathsf{sPKE},\mathcal{A}_{\mathsf{sPKE}}}.$$

Proof. Let $H_0 := \mathsf{Exp}^{\mathsf{ACE\text{-}ctxt\text{-}unpred}}_{\mathsf{ACE},\mathcal{A}}$ and H_1 be as H_0 where F_K is replaced by a truly uniform random function U. As in the proof of Theorem 6.3, one can show that there exists $\mathcal{A}_{\mathsf{PRF}}$ such that

$$\Pr^{H_0}[b = 1] - \Pr^{H_1}[b = 1] = \mathsf{Adv}^{\mathsf{PRF}}_{F,\mathcal{A}_{\mathsf{PRF}}}.$$

The adversary $\mathcal{A}_{\mathsf{sPKE}}$ on input $\big(sp^{\mathsf{sPKE}}, ek^{\mathsf{sPKE}}, dk^{\mathsf{sPKE}}\big)$, sets $i_{q_0} \leftarrow \perp$, $k_q \leftarrow 1$, chooses $q_0 \twoheadleftarrow \{0, \dots, q\}$ uniformly at random, runs $\big(vk^{\mathsf{Sig}}, sk^{\mathsf{Sig}}\big) \leftarrow \mathsf{Sig.Gen}(1^\kappa)$, $crs^{\mathsf{NIZK}} \leftarrow \mathsf{NIZK.Gen}(1^\kappa)$, and gives $sp^{\mathsf{ACE}} := \big(sp^{\mathsf{sPKE}}, vk^{\mathsf{Sig}}, crs^{\mathsf{NIZK}}\big)$ to $\mathcal{A}$. It emulates the oracle $\mathcal{O}_G$ for $\mathcal{A}_1$ as follows. On query (i, t), if $k_q \ne q_0$ and $i \ne i_{q_0}$, then generate an encryption key $ek_i^{\mathsf{ACE}} := \big(vk^{\mathsf{Sig}}, ek_i^{\mathsf{sPKE}}, vk_i^{\mathsf{Sig}}, sk_i^{\mathsf{Sig}}, \sigma_i^{\mathsf{Sig}}, crs^{\mathsf{NIZK}}\big)$ and a decryption key $dk_i^{\mathsf{ACE}} := dk_i^{\mathsf{sPKE}}$ as H_1 does, where $\big(ek_i^{\mathsf{sPKE}}, dk_i^{\mathsf{sPKE}}\big)$ is obtained via $\mathcal{O}_G$ and remembered for future queries. Return ek_i^{ACE} if $t = \mathtt{sen}$, and dk_i^{ACE} if $t = \mathtt{rec}$. If $k_q = q_0$ or $i = i_{q_0}$, replace ek_i^{sPKE} and dk_i^{sPKE} by ek^{sPKE} and dk^{sPKE}, respectively, and set $i_{q_0} \leftarrow i$. In both cases, set $k_q \leftarrow k_q + 1$ at the end. When $\mathcal{A}$ returns $\big(m, i, c = \big(\tilde{c}, \pi^{\mathsf{NIZK}}\big)\big)$, $\mathcal{A}_{\mathsf{sPKE}}$ returns $\big(m, \tilde{c}\big)$.

Let Q be the event that $i_{q_0} = i$, or $q_0 = 0$ and $\mathcal{A}$ does not make the query $(i, \mathtt{sen})$ or $(i, \mathtt{rec})$ to $\mathcal{O}_G$. Note that the probability of Q is $1/(q+1)$ and since $b = \mathsf{ACE.DMod}\big(sp^{\mathsf{ACE}}, \big(\tilde{c}^*, \pi^{\mathsf{NIZK}*}\big), \big(\tilde{c}, \pi^{\mathsf{NIZK}}\big)\big) = 1$ if and only if $\tilde{c}^* = \tilde{c}$, we have

$$\Pr^{\mathsf{Exp}^{\mathsf{sPKE\text{-}ctxt\text{-}unpred}}_{\mathsf{sPKE},\mathcal{A}_{\mathsf{sPKE}}}}[c = c^* \mid Q] = \Pr^{H_1}[b = 1].$$

Hence, we can conclude

$$\begin{aligned}\mathsf{Adv}^{\mathsf{ACE\text{-}ctxt\text{-}unpred}}_{\mathsf{ACE},\mathcal{A}} &= \Pr^{H_0}\big[b = 1\big] = \mathsf{Adv}^{\mathsf{PRF}}_{F,\mathcal{A}_{\mathsf{PRF}}} + \Pr^{H_1}[b = 1] \\ &= \mathsf{Adv}^{\mathsf{PRF}}_{F,\mathcal{A}_{\mathsf{PRF}}} + \Pr^{\mathsf{Exp}^{\mathsf{sPKE\text{-}ctxt\text{-}unpred}}_{\mathsf{sPKE},\mathcal{A}_{\mathsf{sPKE}}}}[c = c^* \mid Q] \\ &\le \mathsf{Adv}^{\mathsf{PRF}}_{F,\mathcal{A}_{\mathsf{PRF}}} + (q+1) \cdot \Pr^{\mathsf{Exp}^{\mathsf{sPKE\text{-}ctxt\text{-}unpred}}_{\mathsf{sPKE},\mathcal{A}_{\mathsf{sPKE}}}}[c = c^*] \\ &= \mathsf{Adv}^{\mathsf{PRF}}_{F,\mathcal{A}_{\mathsf{PRF}}} + (q+1) \cdot \mathsf{Adv}^{\mathsf{sPKE\text{-}ctxt\text{-}unpred}}_{\mathsf{sPKE},\mathcal{A}_{\mathsf{sPKE}}}.\end{aligned}$$

□

We finally prove the uniform decryption and role-respecting properties.

Theorem 6.7. *Let* ACE *be the scheme from above and let* $\mathcal{A}$ *be an attacker on the uniform-decryption security that makes at most* q_R *queries of the form* $(\cdot, \mathtt{rec})$ *to the oracle* $\mathcal{O}_G$*. Then, there exist probabilistic algorithms* $\mathcal{A}_{\mathsf{PRF}}$*,* $\mathcal{A}_{\mathsf{ZK}_1}$*,* $\mathcal{A}_{\mathsf{ZK}_2}$*,* $\mathcal{A}_{\mathsf{Sig}}$*, and* $\mathcal{A}_{\mathsf{rob}}$ *(which are all roughly as efficient as emulating an execution of* $\mathsf{Exp}^{\mathsf{ACE\text{-}URR}}_{\mathsf{ACE},\mathcal{A}}$*) such that*

$$\mathsf{Adv}^{\mathsf{ACE\text{-}uDec}}_{\mathsf{ACE},\mathcal{A}} \leq \mathsf{Adv}^{\mathsf{PRF}}_{F,\mathcal{A}_{\mathsf{PRF}}} + \mathsf{Adv}^{\mathsf{NIZK\text{-}ext_1}}_{\mathsf{NIZK},\mathcal{A}_{\mathsf{ZK}_1}} + \mathsf{Adv}^{\mathsf{NIZK\text{-}ext_2}}_{\mathsf{NIZK},\mathcal{A}_{\mathsf{ZK}_2}} + \mathsf{Adv}^{\mathsf{Sig\text{-}EUF\text{-}CMA}}_{\mathsf{Sig},\mathcal{A}_{\mathsf{Sig}}} + q_R \cdot \mathsf{Adv}^{\mathsf{sPKE\text{-}USROB}}_{\mathsf{sPKE},\mathcal{A}_{\mathsf{rob}}}.$$

Proof. Note that we can assume without loss of generality that $\mathcal{A}$ does not use the oracle $\mathcal{O}_E$ since obtaining encryption keys from $\mathcal{O}_G$ does not decrease the advantage. Let $H_0 := \mathsf{Exp}^{\mathsf{ACE\text{-}URR}}_{\mathsf{ACE},\mathcal{A}}$ and let W_{UDec} be the event that $\mathcal{A}$ wins the uniform-decryption game:

$$W_{\mathsf{UDec}} := \big[\exists j, j' \in J \ m_j \neq \bot \neq m_{j'} \ \wedge \ m_j \neq \ m_{j'}\big].$$

As in the proof of Theorem 6.5, let H_1 be as H_0 with F_K replaced by a uniform random function U, let H_2 be as H_1 with crs^{NIZK} being generated by E_1^{NIZK}, and let H_3 be as H_2, but after $\mathcal{A}$ returns $c = \big(\tilde{c}, \pi^{\mathsf{NIZK}}\big)$, a witness

$$w := \big(ek^{\mathsf{sPKE}}_{i_w}, m_w, r_w, vk^{\mathsf{Sig}}_{i_w}, \sigma^{\mathsf{Sig}}_{i_w}, \sigma^{\mathsf{Sig}}_{c,w}\big)$$

for the statement $x := \big(vk^{\mathsf{Sig}}, \tilde{c}\big)$ is extracted from the proof π^{NIZK} by E_2^{NIZK}. We define the events $V := \big[\mathsf{NIZK.Ver}\big(crs^{\mathsf{NIZK}}, x, \pi^{\mathsf{NIZK}}\big) = 1\big]$, $B_E := \big[V \ \wedge \ (x, w) \notin R\big]$, and B_S as the event that $(x, w) \in R$ and $ek^{\mathsf{sPKE}}_{i_w}$ is not contained in an answer from $\mathcal{O}_G$ to $\mathcal{A}$. Is can be shown as in the proof of Theorem 6.5 that there exist $\mathcal{A}_{\mathsf{PRF}}$, $\mathcal{A}_{\mathsf{ZK}_1}$, $\mathcal{A}_{\mathsf{ZK}_2}$, and $\mathcal{A}_{\mathsf{Sig}}$ such that

$$\begin{aligned}
\Pr^{H_0}[W_{\mathsf{UDec}}] - \Pr^{H_3}[W_{\mathsf{UDec}}] &= \mathsf{Adv}^{\mathsf{PRF}}_{F,\mathcal{A}_{\mathsf{PRF}}} + \mathsf{Adv}^{\mathsf{NIZK\text{-}ext_1}}_{\mathsf{NIZK},\mathcal{A}_{\mathsf{ZK}_1}}, \\
\Pr^{H_3}[B_E] &\leq \mathsf{Adv}^{\mathsf{NIZK\text{-}ext_2}}_{\mathsf{NIZK},\mathcal{A}_{\mathsf{ZK}_2}}, \\
\Pr^{H_3}[B_S] &\leq \mathsf{Adv}^{\mathsf{Sig\text{-}EUF\text{-}CMA}}_{\mathsf{Sig},\mathcal{A}_{\mathsf{Sig}}},
\end{aligned}$$

where the last inequality uses that $\mathcal{A}$ does not query the oracle $\mathcal{O}_E$. Now let H_4 be as H_3 where for $j \in J$, $m_j \leftarrow \mathsf{ACE.Dec}\big(\mathsf{ACE.Gen}(msk, j, \mathtt{rec}), c'\big)$ is replaced by

$$m_j \leftarrow \begin{cases} m_w, & ek^{\mathsf{sPKE}}_j = ek^{\mathsf{sPKE}}_{i_w} \text{ for } \big(ek^{\mathsf{sPKE}}_j, dk^{\mathsf{sPKE}}_j\big) = \mathsf{sPKE.Gen}\big(msk^{\mathsf{sPKE}}; U([j, 0])\big), \\ \bot, & \text{else.} \end{cases}$$

One can show as in the proof of Theorem 6.5 that there exists a probabilistic algorithm $\mathcal{A}_{\mathsf{rob}}$ such that

$$\Pr^{H_3}[W_{\mathsf{UDec}} \mid V \cap \neg B_E \cap \neg B_S] - \Pr^{H_4}[W_{\mathsf{UDec}} \mid V \cap \neg B_E \cap \neg B_S] \leq q_R \cdot \mathsf{Adv}^{\mathsf{sPKE\text{-}USROB}}_{\mathsf{sPKE},\mathcal{A}_{\mathsf{rob}}}.$$

Note that $\mathcal{A}$ cannot win in H_4 since if $m_j \neq \bot \neq m_{j'}$, then $m_j = m_w = m_{j'}$. This implies that $\Pr^{H_3}[W_{\mathsf{UDec}} \mid V \cap \neg B_E \cap \neg B_S] \leq q_R \, \mathsf{Adv}^{\mathsf{sPKE\text{-}USROB}}_{\mathsf{sPKE},\mathcal{A}_{\mathsf{rob}}}$. Note that $\mathcal{A}$ can only win in H_3 if V occurs since otherwise $c' = \bot$ and consequently $m_j = \bot$ for all $j \in J$. We therefore obtain

$$\begin{aligned}
\Pr^{H_3}[W_{\mathsf{UDec}}] &= \Pr^{H_3}[W_{\mathsf{UDec}} \cap V \cap B_E] + \Pr^{H_3}[W_{\mathsf{UDec}} \cap V \cap \neg B_E \cap B_S] \\
&\qquad + \Pr^{H_3}[W_{\mathsf{UDec}} \cap V \cap \neg B_E \cap \neg B_S] \\
&\leq \Pr^{H_3}[B_E] + \Pr^{H_3}[B_S] + \Pr^{H_3}[W_{\mathsf{UDec}} \mid V \cap \neg B_E \cap \neg B_S] \\
&\leq \mathsf{Adv}^{\mathsf{NIZK\text{-}ext_2}}_{\mathsf{NIZK},\mathcal{A}_{\mathsf{ZK}_2}} + \mathsf{Adv}^{\mathsf{Sig\text{-}EUF\text{-}CMA}}_{\mathsf{Sig},\mathcal{A}_{\mathsf{Sig}}} + q_R \cdot \mathsf{Adv}^{\mathsf{sPKE\text{-}USROB}}_{\mathsf{sPKE},\mathcal{A}_{\mathsf{rob}}}.
\end{aligned}$$

Together with $\Pr^{H_0}[W_{\mathsf{UDec}}] - \Pr^{H_3}[W_{\mathsf{UDec}}] = \mathsf{Adv}^{\mathsf{PRF}}_{F,\mathcal{A}_{\mathsf{PRF}}} + \mathsf{Adv}^{\mathsf{NIZK\text{-}ext_1}}_{\mathsf{NIZK},\mathcal{A}_{\mathsf{ZK}_1}}$, this concludes the proof. ☐

Theorem 6.8. *Let* ACE *be the scheme from above and let $\mathcal{A}$ be an attacker on the role-respecting security that makes at most q_S queries of the form $(\cdot, \mathtt{sen})$ and at most q_R queries of the form $(\cdot, \mathtt{rec})$ to the oracle $\mathcal{O}_G$, and at most q_E queries to the oracle $\mathcal{O}_E$. Then, there exist probabilistic algorithms $\mathcal{A}_{\mathsf{PRF}}$, $\mathcal{A}_{\mathsf{ZK}_1}$, $\mathcal{A}_{\mathsf{ZK}_2}$, $\mathcal{A}_{\mathsf{Sig}}$, and $\mathcal{A}_{\mathsf{rob}}$ (which are all roughly as efficient as emulating an execution of $\mathsf{Exp}^{\mathsf{ACE\text{-}URR}}_{\mathsf{ACE},\mathcal{A}}$) such that*

$$\mathsf{Adv}^{\mathsf{ACE\text{-}RR}}_{\mathsf{ACE},\mathcal{A}} \leq \mathsf{Adv}^{\mathsf{PRF}}_{F,\mathcal{A}_{\mathsf{PRF}}} + \mathsf{Adv}^{\mathsf{NIZK\text{-}ext_1}}_{\mathsf{NIZK},\mathcal{A}_{\mathsf{ZK}_1}} + \mathsf{Adv}^{\mathsf{NIZK\text{-}ext_2}}_{\mathsf{NIZK},\mathcal{A}_{\mathsf{ZK}_2}} + (q_E+1)\cdot \mathsf{Adv}^{\mathsf{Sig\text{-}EUF\text{-}CMA}}_{\mathsf{Sig},\mathcal{A}_{\mathsf{Sig}}} \\ + q_R \cdot \mathsf{Adv}^{\mathsf{sPKE\text{-}USROB}}_{\mathsf{sPKE},\mathcal{A}_{\mathsf{rob}}} + (q_S+q_R+q_E)^2 \cdot \mathsf{Col}^{\mathsf{ek}}_{\mathsf{sPKE}}.$$

Proof. Let $H_0, \dots, H_4$, $V := \big[\mathsf{NIZK.Ver}\big(crs^{\mathsf{NIZK}}, x, \pi^{\mathsf{NIZK}}\big) = 1\big]$, and $B_E := \big[V \wedge (x,w) \notin R\big]$ for the statement $x := \big(vk^{\mathsf{Sig}}, \tilde{c}\big)$ and the extracted witness $w := \big(ek^{\mathsf{sPKE}}_{i_w}, m_w, r_w, vk^{\mathsf{Sig}}_{i_w}, \sigma^{\mathsf{Sig}}_{i_w}, \sigma^{\mathsf{Sig}}_{c,w}\big)$ be defined as in the proof of Theorem 6.7, and let W_{RR} be the event that $\mathcal{A}$ wins the role-respecting game:

$$W_{\mathsf{RR}} := \big[c' \neq \bot \ \wedge\ \mathsf{dct} = \mathtt{false} \ \wedge\ \neg\big(\exists i \in I\ \forall j \in J\ (m_j \neq \bot \leftrightarrow P(i,j) = 1)\big)\big].$$

As in that proof, there exist $\mathcal{A}_{\mathsf{PRF}}$, $\mathcal{A}_{\mathsf{ZK}_1}$, and $\mathcal{A}_{\mathsf{ZK}_2}$ such that

$$\Pr^{H_0}[W_{\mathsf{RR}}] - \Pr^{H_3}[W_{\mathsf{RR}}] = \mathsf{Adv}^{\mathsf{PRF}}_{F,\mathcal{A}_{\mathsf{PRF}}} + \mathsf{Adv}^{\mathsf{NIZK\text{-}ext_1}}_{\mathsf{NIZK},\mathcal{A}_{\mathsf{ZK}_1}}, \tag{12}$$

and

$$\Pr^{H_3}[B_E] \leq \mathsf{Adv}^{\mathsf{NIZK\text{-}ext_2}}_{\mathsf{NIZK},\mathcal{A}_{\mathsf{ZK}_2}}. \tag{13}$$

Let E_G be the event that the extracted key $ek^{\mathsf{sPKE}}_{i_w}$ is contained in an answer from $\mathcal{O}_G$ to $\mathcal{A}$. One can show similarly as in the proof of Theorem 6.5 that there exists an algorithm $\mathcal{A}_{\mathsf{rob}}$ such that

$$\Pr^{H_3}[W_{\mathsf{RR}} \cap V \cap \neg B_E \cap E_G] - \Pr^{H_4}[W_{\mathsf{RR}} \cap V \cap \neg B_E \cap E_G] \leq q_R \cdot \mathsf{Adv}^{\mathsf{sPKE\text{-}USROB}}_{\mathsf{sPKE},\mathcal{A}_{\mathsf{rob}}}. \tag{14}$$

We first show that if V, $\neg B_E$, and E_G occur in H_4, $\mathcal{A}$ can only win if two encryption keys generated by sPKE.Gen are equal, which happens only with small probability.

Claim 1. *We have*

$$\Pr^{H_4}[W_{\mathsf{RR}} \cap V \cap \neg B_E \cap E_G] \leq (q_S+q_R+q_E)^2 \cdot \mathsf{Col}^{\mathsf{ek}}_{\mathsf{sPKE}}.$$

Proof of claim. If V, $\neg B_E$, and E_G occur, then there is an $i_0 \in I$ such that $ek^{\mathsf{sPKE}}_{i_0} = ek^{\mathsf{sPKE}}_{i_w}$ for $\big(ek^{\mathsf{sPKE}}_{i_0}, dk^{\mathsf{sPKE}}_{i_0}\big) = \mathsf{sPKE.Gen}\big(msk^{\mathsf{sPKE}}; U([i_0,0])\big)$. Using $P(i,j) = 1 \leftrightarrow i = j$, we have that $\mathcal{A}$ only wins if there exists $j \in J \setminus \{i_0\}$ such that $m_j \neq \bot$ or if $i_0 \in J$ and $m_{i_0} = \bot$. Because in H_4, m_j for $j \in J$ is equal to m_w if $ek^{\mathsf{sPKE}}_j = ek^{\mathsf{sPKE}}_{i_w}$ for $\big(ek^{\mathsf{sPKE}}_j, dk^{\mathsf{sPKE}}_j\big) = \mathsf{sPKE.Gen}\big(msk^{\mathsf{sPKE}}; U([j,0])\big)$, and $\bot$ otherwise, we have $m_{i_0} \neq \bot$ if $i_0 \in J$. Moreover, for $i_0 \neq j \in J$, we have $m_j = \bot$ unless $ek^{\mathsf{sPKE}}_j = ek^{\mathsf{sPKE}}_{i_0}$. This means that $\mathcal{A}$ can only win if sPKE.Gen generates the same encryption key for the randomness values $U([i_0,0]$ and $U([j,0]$ for some $i_0 \neq j \in J$. Since at most $q_S + q_R + q_E$ key pairs are generated in the experiment, there are at most $(q_S+q_R+q_E)^2$ pairs of encryption keys that could collide. For each such pair, the collision probability is bounded by $\mathsf{Col}^{\mathsf{ek}}_{\mathsf{sPKE}}$ because for $i \neq i'$, $U([i,0])$ and $U([i',0])$ are independent and uniformly distributed. Hence, the claim follows. ◊

Now let E_E be the event that $\mathcal{A}$ made a query $(i,\cdot)$ to $\mathcal{O}_E$ such that $ek_i^{\mathsf{sPKE}} = ek_{i_w}^{\mathsf{sPKE}}$ and $vk_i^{\mathsf{Sig}} = vk_{i_w}^{\mathsf{Sig}}$ for $\big(ek_i^{\mathsf{sPKE}}, dk_i^{\mathsf{sPKE}}\big) = \mathsf{sPKE.Gen}\big(msk^{\mathsf{sPKE}}; U([i,0])\big)$ and $\big(vk_i^{\mathsf{Sig}}, sk_i^{\mathsf{Sig}}\big) = \mathsf{Sig.Gen}\big(1^\kappa; U([i,1])\big)$. We next show that if $\mathcal{A}$ wins and $V \cap \neg B_E \cap \neg E_G \cap E_E$ occurs, $\mathcal{A}$ forged a signature on $\tilde{c}$.

Claim 2. *There exists a probabilistic algorithm $\mathcal{A}_{\mathsf{Sig}_1}$ such that*

$$\Pr^{H_3}[W_{\mathsf{RR}} \cap V \cap \neg B_E \cap \neg E_G \cap E_E] \leq q_E \cdot \mathsf{Adv}^{\mathsf{Sig\text{-}EUF\text{-}CMA}}_{\mathsf{Sig},\mathcal{A}_{\mathsf{Sig}_1}}.$$

Proof of claim. On input $vk^{\mathsf{Sig}*}$, $\mathcal{A}_{\mathsf{Sig}_1}$ initializes $i_{q_0} \leftarrow \bot$, $k_q \leftarrow 1$, chooses $q_0 \twoheadleftarrow \{1,\ldots,q_E\}$ uniformly at random, generates $\big(sp^{\mathsf{sPKE}}, msk^{\mathsf{sPKE}}\big) \leftarrow \mathsf{sPKE.Setup}(1^\kappa)$, $\big(vk^{\mathsf{Sig}}, sk^{\mathsf{Sig}}\big) \leftarrow \mathsf{Sig.Gen}(1^\kappa)$, and $\big(crs^{\mathsf{NIZK}}, \xi^{\mathsf{NIZK}}\big) \leftarrow E_1^{\mathsf{NIZK}}(1^\kappa)$ as H_3, and gives $sp^{\mathsf{ACE}} := \big(sp^{\mathsf{sPKE}}, vk^{\mathsf{Sig}}, crs^{\mathsf{NIZK}}\big)$ to $\mathcal{A}$. It emulates the oracles for $\mathcal{A}$ as follows.

$\mathcal{O}_G(\cdot,\cdot)$**:** Generate the requested key exactly as H_3 does and return it.

$\mathcal{O}_E(\cdot,\cdot)$**:** On query (i,m), if $k_q \neq q_0$ and $i \neq i_{q_0}$, generate an encryption key ek_i^{ACE} as H_3, encrypt m using ek_i^{ACE}, and return the resulting ciphertext. If $k_q = q_0$ or $i = i_{q_0}$, set $i_{q_0} \leftarrow i$, execute $\big(ek_i^{\mathsf{sPKE}}, dk_i^{\mathsf{sPKE}}\big) \leftarrow \mathsf{sPKE.Gen}\big(msk^{\mathsf{sPKE}}; U([i,0])\big)$, $\sigma_i^{\mathsf{Sig}} \leftarrow \mathsf{Sig.Sign}\big(sk^{\mathsf{Sig}}, \big[ek_i^{\mathsf{sPKE}}, vk_i^{\mathsf{Sig}}\big]; U([i,2])\big)$, and set $vk_i^{\mathsf{Sig}} := vk^{\mathsf{Sig}*}$. Then, sample randomness r and compute $\tilde{c} \leftarrow \mathsf{sPKE.Enc}\big(ek_i^{\mathsf{sPKE}}, m; r\big)$, query the signing oracle on $\tilde{c}$ to obtain a signature σ_c^{Sig}, and run

$$\pi^{\mathsf{NIZK}} \leftarrow \mathsf{NIZK.Prove}\big(crs^{\mathsf{NIZK}}, x := \big(vk^{\mathsf{Sig}}, \tilde{c}\big), w := \big(ek_i^{\mathsf{sPKE}}, m, r, vk_i^{\mathsf{Sig}}, \sigma_i^{\mathsf{Sig}}, \sigma_c^{\mathsf{Sig}}\big)\big).$$

Finally, return the ciphertext $c := \big(\tilde{c}, \pi^{\mathsf{NIZK}}\big)$. In all cases, set $k_q \leftarrow k_q + 1$ at the end.

When $\mathcal{A}$ returns $c = \big(\tilde{c}, \pi^{\mathsf{NIZK}}\big)$, $\mathcal{A}_{\mathsf{Sig}_1}$ extracts a witness

$$w := \big(ek_{i_w}^{\mathsf{sPKE}}, m_w, r_w, vk_{i_w}^{\mathsf{Sig}}, \sigma_{i_w}^{\mathsf{Sig}}, \sigma_{c,w}^{\mathsf{Sig}}\big) \leftarrow E_2^{\mathsf{NIZK}}\big(crs^{\mathsf{NIZK}}, \xi^{\mathsf{NIZK}}, x := \big(vk^{\mathsf{Sig}}, \tilde{c}\big), \pi^{\mathsf{NIZK}}\big).$$

It finally returns the forgery attempt $\big(\tilde{c}, \sigma_{c,w}^{\mathsf{Sig}}\big)$.

Note that if $\mathcal{A}$ wins the role-respecting game, $\mathsf{ACE.DMod}\big(sp^{\mathsf{ACE}}, \hat{c}, c\big) = 0$ for all $\hat{c}$ that $\mathcal{O}_E$ has returned. Since $\mathsf{ACE.DMod}$ checks for equality of sPKE ciphertexts, this means that $\mathcal{A}_{\mathsf{Sig}_1}$ has not issued the query $\tilde{c}$ to its signing oracle. Furthermore, if the extraction and verification succeed, $\sigma_{c,w}^{\mathsf{Sig}}$ is a valid signature for $\tilde{c}$. Let Q be the event that $ek_{i_{q_0}}^{\mathsf{sPKE}} = ek_{i_w}^{\mathsf{sPKE}}$ and $vk_{i_{q_0}}^{\mathsf{Sig}} = vk_{i_w}^{\mathsf{Sig}}$. If Q and $V \cap \neg B_E \cap \neg E_G \cap E_E$ occur, $\mathcal{A}$ has not requested $ek_{i_{q_0}}^{\mathsf{ACE}}$ from $\mathcal{O}_G$ and hence $\mathcal{A}_{\mathsf{Sig}_1}$ perfectly emulates H_3. This implies

$$\Pr^{\mathsf{Exp}^{\mathsf{Sig\text{-}EUF\text{-}CMA}}_{\mathsf{Sig},\mathcal{A}_{\mathsf{Sig}_1}}}[W_{\mathsf{Sig}} \mid V \cap \neg B_E \cap \neg E_G \cap E_E \cap Q] \geq \Pr^{H_3}[W_{\mathsf{RR}} \mid V \cap \neg B_E \cap \neg E_G \cap E_E],$$

where W_{Sig} denotes the event that $\mathcal{A}_{\mathsf{Sig}_1}$ wins in the signature forgery game. We further have

$$\Pr^{\mathsf{Exp}^{\mathsf{Sig\text{-}EUF\text{-}CMA}}_{\mathsf{Sig},\mathcal{A}_{\mathsf{Sig}_1}}}[Q \mid V \cap \neg B_E \cap \neg E_G \cap E_E] = 1/q_E.$$

This implies for $p_G := \Pr^{\mathsf{Exp}^{\mathsf{Sig\text{-}EUF\text{-}CMA}}_{\mathsf{Sig},\mathcal{A}_{\mathsf{Sig}_1}}}[V \cap \neg B_E \cap \neg E_G \cap E_E \cap Q]$,

$$\begin{aligned}
\mathsf{Adv}^{\mathsf{Sig\text{-}EUF\text{-}CMA}}_{\mathsf{Sig},\mathcal{A}_{\mathsf{Sig}_1}} &= \Pr^{\mathsf{Exp}^{\mathsf{Sig\text{-}EUF\text{-}CMA}}_{\mathsf{Sig},\mathcal{A}_{\mathsf{Sig}_1}}}[W_{\mathsf{Sig}}] \geq \Pr^{\mathsf{Exp}^{\mathsf{Sig\text{-}EUF\text{-}CMA}}_{\mathsf{Sig},\mathcal{A}_{\mathsf{Sig}_1}}}[W_{\mathsf{Sig}} \mid V \cap \neg B_E \cap \neg E_G \cap E_E \cap Q] \cdot p_G \\
&\geq \Pr^{H_3}[W_{\mathsf{RR}} \mid V \cap \neg B_E \cap \neg E_G \cap E_E] \cdot p_G \\
&= \Pr^{H_3}[W_{\mathsf{RR}} \cap V \cap \neg B_E \cap \neg E_G \cap E_E] \cdot \frac{p_G}{\Pr^{H_3}[V \cap \neg B_E \cap \neg E_G \cap E_E]}.
\end{aligned}$$

Since $[V \cap \neg B_E \cap \neg E_G \cap E_E]$ in H_3 has the same probability as in $\mathsf{Exp}^{\mathsf{Sig\text{-}EUF\text{-}CMA}}_{\mathsf{Sig},\mathcal{A}_{\mathsf{Sig}_1}}$, we have

$$\frac{p_G}{\Pr^{H_3}[V \cap \neg B_E \cap \neg E_G \cap E_E]} = \Pr^{\mathsf{Exp}^{\mathsf{Sig\text{-}EUF\text{-}CMA}}_{\mathsf{Sig},\mathcal{A}_{\mathsf{Sig}_1}}}[Q \mid V \cap \neg B_E \cap \neg E_G \cap E_E] = \frac{1}{q_E},$$

which implies the claim. $\Diamond$

Finally, we show that if $\mathcal{A}$ wins and $V \cap \neg B_E \cap \neg E_G \cap \neg E_E$ occurs, $\mathcal{A}$ forged a signature on $\left[ek^{\mathsf{sPKE}}_{i_w}, vk^{\mathsf{Sig}}_{i_w}\right]$.

Claim 3. *There exists a probabilistic algorithm $\mathcal{A}_{\mathsf{Sig}_2}$ such that*

$$\Pr^{H_3}[W_{\mathsf{RR}} \cap V \cap \neg B_E \cap \neg E_G \cap \neg E_E] \le \mathsf{Adv}^{\mathsf{Sig\text{-}EUF\text{-}CMA}}_{\mathsf{Sig},\mathcal{A}_{\mathsf{Sig}_2}}.$$

Proof of claim. The algorithm $\mathcal{A}_{\mathsf{Sig}_2}$ on input $vk^{\mathsf{Sig}*}$ runs $\left(sp^{\mathsf{sPKE}}, msk^{\mathsf{sPKE}}\right) \leftarrow \mathsf{sPKE.Setup}(1^\kappa)$ and $\left(crs^{\mathsf{NIZK}}, \xi^{\mathsf{NIZK}}\right) \leftarrow E_1^{\mathsf{NIZK}}(1^\kappa)$, and gives $sp^{\mathsf{ACE}} := \left(sp^{\mathsf{sPKE}}, vk^{\mathsf{Sig}*}, crs^{\mathsf{NIZK}}\right)$ to $\mathcal{A}$. It emulates the oracles for $\mathcal{A}$ as follows.

$\mathcal{O}_G(\cdot,\cdot)$**:** Generate the requested key as H_3, but obtain the signature σ_i^{Sig} via a query to the signing oracle. Remember the signature and when asked again for the same i, reuse σ_i^{Sig} instead of issuing another query. This ensures that the oracle behaves as the one in H_3 and returns the same key for repeated queries.

$\mathcal{O}_E(\cdot,\cdot)$**:** On query (i, m), generate an encryption key as for a query $(i, \mathtt{sen})$ to $\mathcal{O}_G$, encrypt m using that key, and return the resulting ciphertext.

When $\mathcal{A}$ returns $c = \left(\tilde{c}, \pi^{\mathsf{NIZK}}\right)$, $\mathcal{A}_{\mathsf{Sig}_1}$ extracts a witness

$$w := \left(ek^{\mathsf{sPKE}}_{i_w}, m_w, r_w, vk^{\mathsf{Sig}}_{i_w}, \sigma^{\mathsf{Sig}}_{i_w}, \sigma^{\mathsf{Sig}}_{c,w}\right) \leftarrow E_2^{\mathsf{NIZK}}\left(crs^{\mathsf{NIZK}}, \xi^{\mathsf{NIZK}}, x := \left(vk^{\mathsf{Sig}}, \tilde{c}\right), \pi^{\mathsf{NIZK}}\right).$$

It finally returns the forgery attempt $\left(\left[ek^{\mathsf{sPKE}}_{i_w}, vk^{\mathsf{Sig}}_{i_w}\right], \sigma^{\mathsf{Sig}}_{i_w}\right)$. Note that if $W_{\mathsf{RR}} \cap V \cap \neg B_E \cap \neg E_G \cap \neg E_E$ occurs, $\sigma^{\mathsf{Sig}}_{i_w}$ is a valid signature for $\left[ek^{\mathsf{sPKE}}_{i_w}, vk^{\mathsf{Sig}}_{i_w}\right]$ and $\mathcal{A}_{\mathsf{Sig}_2}$ has not requested a signature for this value from the signing oracle. Therefore, $\mathcal{A}_{\mathsf{Sig}_2}$ wins the forgery game and thus the probability of that event is bounded by $\mathsf{Adv}^{\mathsf{Sig\text{-}EUF\text{-}CMA}}_{\mathsf{Sig},\mathcal{A}_{\mathsf{Sig}_2}}$. $\Diamond$

Combining Claims 2 and 3, we obtain

$$\Pr^{H_3}[W_{\mathsf{RR}} \cap V \cap \neg B_E \cap \neg E_G] \le q_E \cdot \mathsf{Adv}^{\mathsf{Sig\text{-}EUF\text{-}CMA}}_{\mathsf{Sig},\mathcal{A}_{\mathsf{Sig}_1}} + \mathsf{Adv}^{\mathsf{Sig\text{-}EUF\text{-}CMA}}_{\mathsf{Sig},\mathcal{A}_{\mathsf{Sig}_2}}.$$

Let $\mathcal{A}_{\mathsf{Sig}}$ be the algorithm that runs $\mathcal{A}_{\mathsf{Sig}_1}$ with probability $\frac{q_E}{q_E+1}$ and $\mathcal{A}_{\mathsf{Sig}_2}$ with probability $\frac{1}{q_E+1}$. We then have

$$\begin{aligned} \mathsf{Adv}^{\mathsf{Sig\text{-}EUF\text{-}CMA}}_{\mathsf{Sig},\mathcal{A}_{\mathsf{Sig}}} &= \frac{q_E}{q_E+1} \cdot \mathsf{Adv}^{\mathsf{Sig\text{-}EUF\text{-}CMA}}_{\mathsf{Sig},\mathcal{A}_{\mathsf{Sig}_1}} + \frac{1}{q_E+1} \cdot \mathsf{Adv}^{\mathsf{Sig\text{-}EUF\text{-}CMA}}_{\mathsf{Sig},\mathcal{A}_{\mathsf{Sig}_2}} \\ &\ge \frac{1}{q_E+1} \cdot \Pr^{H_3}[W_{\mathsf{RR}} \cap V \cap \neg B_E \cap \neg E_G]. \end{aligned} \tag{15}$$

Note that W_{RR} implies $c' \neq \perp$ and therefore V, i.e., the events W_{RR} and $W_{\mathsf{RR}} \cap V$ are equal. Thus,

$$\begin{aligned}\Pr^{H_3}[W_{\mathsf{RR}}] &= \Pr^{H_3}[W_{\mathsf{RR}} \cap B_E] + \Pr^{H_3}[W_{\mathsf{RR}} \cap V \cap \neg B_E \cap E_G] + \Pr^{H_3}[W_{\mathsf{RR}} \cap V \cap \neg B_E \cap \neg E_G] \\ &\overset{(13),(14),(15)}{\leq} \mathsf{Adv}^{\mathsf{NIZK\text{-}ext_2}}_{\mathsf{NIZK},\mathcal{A}_{\mathsf{ZK}_2}} + q_R \cdot \mathsf{Adv}^{\mathsf{sPKE\text{-}USROB}}_{\mathsf{sPKE},\mathcal{A}_{\mathsf{rob}}} + \Pr^{H_4}[W_{\mathsf{RR}} \cap V \cap \neg B_E \cap E_G] \\ &\qquad + (q_E + 1) \cdot \mathsf{Adv}^{\mathsf{Sig\text{-}EUF\text{-}CMA}}_{\mathsf{Sig},\mathcal{A}_{\mathsf{Sig}}}.\end{aligned}$$

Combined with Claim 1 and equation (12), this concludes the proof. □

6.2 Lifting Equality to Disjunction of Equalities

We finally show how an ACE scheme for equality, as the one from Section 6.1, can be used to construct a scheme for the policy $P_{\mathsf{DEq}} \colon \mathcal{D}^\ell \times \mathcal{D}^\ell \to \{0,1\}$ with

$$P_{\mathsf{DEq}}\big(\mathbf{x} = (x_1, \dots, x_\ell), \mathbf{y} = (y_1, \dots, y_\ell)\big) = 1 \ :\Longleftrightarrow\ \bigvee_{i=1}^{\ell} x_i = y_i,$$

where $\mathcal{D}$ is some finite set and $\ell \in \mathbb{N}$.[8] This policy can for example be used to implement the no read-up and now write-down principle ($P(i,j) = 1 \Leftrightarrow i \leq j$) from the Bell–LaPadula model [BL73] via an appropriate encoding of the roles [FGKO17].

The intuition of our construction is as follows. A key for a role $\mathbf{x} = (x_1, \dots, x_\ell)$ contains one key of the ACE scheme for equality for each component x_i of the role vector. To encrypt a message, this message is encrypted with each of these keys. To decrypt, one tries to decrypt each ciphertext component with the corresponding key. If at least one component of the sender and receiver roles match (i.e., if the policy is satisfied), one of the decryptions is successful. So far, the construction is identical to the one by Fuchsbauer et al. [FGKO17]. That construction is, however, not role-respecting, since a dishonest sender with keys for more than one role can arbitrarily mix the components of the keys for the encryption. Moreover, the construction does not guarantee uniform decryption, because different messages can be encrypted in different components. We fix these issues using the same techniques we used in our construction of the scheme for equality, i.e., we add a signature of the key vector to the encryption keys, sign the ciphertexts, and require a zero-knowledge proof that a valid key combination was used to encrypt the same message for each component and that all signatures a valid.

Our construction. Let $\mathsf{ACE}_=$ be an ACE with modification detection scheme for the equality predicate on $\mathcal{D} \times [\ell]$, let Sig be a signature scheme, let F be a PRF, and let NIZK be a NIZK proof of knowledge system for the language $L := \{x \mid \exists w\ (x, w) \in R\}$, where the relation R is defined as follows: for $x = \big(vk^{\mathsf{Sig}}, c_1, \dots, c_\ell\big)$ and $w = \big(ek^{\mathsf{ACE}_=}_{(x_1,1)}, \dots, ek^{\mathsf{ACE}_=}_{(x_\ell,\ell)}, m, r_1, \dots, r_\ell, vk^{\mathsf{Sig}}_{\mathbf{x}}, \sigma^{\mathsf{Sig}}_{\mathbf{x}}, \sigma^{\mathsf{Sig}}_{c}\big)$, $(x, w) \in R$ if and only if

$$\begin{aligned}\bigwedge_{i=1}^{\ell} c_i = \mathsf{ACE}_=.\mathsf{Enc}\big(ek^{\mathsf{ACE}_=}_{(x_i,i)}, m; r_i\big) \ \wedge\ \mathsf{Sig.Ver}\big(vk^{\mathsf{Sig}}_{\mathbf{x}}, [c_1, \dots, c_\ell], \sigma^{\mathsf{Sig}}_c\big) = 1 \\ \wedge\ \mathsf{Sig.Ver}\big(vk^{\mathsf{Sig}}, \big[ek^{\mathsf{ACE}_=}_{(x_1,1)}, \dots, ek^{\mathsf{ACE}_=}_{(x_\ell,\ell)}, vk^{\mathsf{Sig}}_{\mathbf{x}}\big], \sigma^{\mathsf{Sig}}_{\mathbf{x}}\big) = 1\end{aligned}$$

[8] In this section, we denote roles by $\mathbf{x}$ and $\mathbf{y}$ instead of i and j. To be compatible with our definitions that consider policies $[n] \times [n] \to \{0,1\}$, one needs to identify elements of $\mathcal{D}^\ell$ with numbers in $[n]$. We will ignore this technicality to simplify the presentation.

We define an ACE scheme $\mathsf{ACE}_{\mathsf{DEq}}$ as follows:

Setup: On input a security parameter 1^κ and the policy P_{DEq}, the algorithm $\mathsf{ACE}_{\mathsf{DEq}}.\mathsf{Setup}$ picks a random key K for F and runs

$$\begin{aligned}
\big(msk^{\mathsf{ACE}_=}, sp^{\mathsf{ACE}_=}\big) &\leftarrow \mathsf{ACE}_=.\mathsf{Setup}(1^\kappa),\\
\big(vk^{\mathsf{Sig}}, sk^{\mathsf{Sig}}\big) &\leftarrow \mathsf{Sig}.\mathsf{Gen}(1^\kappa),\\
crs^{\mathsf{NIZK}} &\leftarrow \mathsf{NIZK}.\mathsf{Gen}(1^\kappa).
\end{aligned}$$

It outputs the master secret key $msk^{\mathsf{ACE}_{\mathsf{DEq}}} := \big(K, msk^{\mathsf{ACE}_=}, vk^{\mathsf{Sig}}, sk^{\mathsf{Sig}}, crs^{\mathsf{NIZK}}\big)$ and the sanitizer parameters $sp^{\mathsf{ACE}_{\mathsf{DEq}}} := \big(sp^{\mathsf{ACE}_=}, vk^{\mathsf{Sig}}, crs^{\mathsf{NIZK}}\big)$.

Key Generation: The algorithm $\mathsf{ACE}_{\mathsf{DEq}}.\mathsf{Gen}$ on input a master secret key $msk^{\mathsf{ACE}_{\mathsf{DEq}}} = \big(K, msk^{\mathsf{ACE}_=}, vk^{\mathsf{Sig}}, sk^{\mathsf{Sig}}, crs^{\mathsf{NIZK}}\big)$, a role $\mathbf{x} \in \mathcal{D}^\ell$, and the type $\mathtt{sen}$, generates

$$\begin{aligned}
ek^{\mathsf{ACE}_=}_{(x_i,i)} &\leftarrow \mathsf{ACE}_=.\mathsf{Gen}\big(msk^{\mathsf{ACE}_=}, (x_i,i), \mathtt{sen}\big) \quad (\text{for } i \in [\ell]),\\
\big(vk^{\mathsf{Sig}}_{\mathbf{x}}, sk^{\mathsf{Sig}}_{\mathbf{x}}\big) &\leftarrow \mathsf{Gen}(1^\kappa; F_K([(x_1,1),0])),\\
\sigma^{\mathsf{Sig}}_{\mathbf{x}} &\leftarrow \mathsf{Sig}.\mathsf{Sign}\big(sk^{\mathsf{Sig}}, \big[ek^{\mathsf{ACE}_=}_{(x_1,1)}, \ldots, ek^{\mathsf{ACE}_=}_{(x_\ell,\ell)}, vk^{\mathsf{Sig}}_{\mathbf{x}}\big]; F_K([(x_1,1),1])\big),
\end{aligned}$$

and outputs the encryption key $ek^{\mathsf{ACE}_{\mathsf{DEq}}}_{\mathbf{x}} := \big(vk^{\mathsf{Sig}}, ek^{\mathsf{ACE}_=}_{(x_1,1)}, \ldots, ek^{\mathsf{ACE}_=}_{(x_\ell,\ell)}, vk^{\mathsf{Sig}}_{\mathbf{x}}, sk^{\mathsf{Sig}}_{\mathbf{x}}, \sigma^{\mathsf{Sig}}_{\mathbf{x}}, crs^{\mathsf{NIZK}}\big)$; on input $msk^{\mathsf{ACE}_{\mathsf{DEq}}}$, a role $\mathbf{y} \in \mathcal{D}^\ell$, and the type $\mathtt{rec}$, it generates for $i \in [\ell]$,

$$dk^{\mathsf{ACE}_=}_{(y_i,i)} \leftarrow \mathsf{ACE}_=.\mathsf{Gen}\big(msk^{\mathsf{ACE}_=}, (y_i,i), \mathtt{rec}\big),$$

and outputs the decryption key $dk^{\mathsf{ACE}_{\mathsf{DEq}}}_{\mathbf{y}} := \big(dk^{\mathsf{ACE}_=}_{(y_1,1)}, \ldots, dk^{\mathsf{ACE}_=}_{(y_\ell,\ell)}\big)$.

Encrypt: On input an encryption key $ek^{\mathsf{ACE}_{\mathsf{DEq}}}_{\mathbf{x}} = \big(vk^{\mathsf{Sig}}, ek^{\mathsf{ACE}_=}_{(x_1,1)}, \ldots, ek^{\mathsf{ACE}_=}_{(x_\ell,\ell)}, vk^{\mathsf{Sig}}_{\mathbf{x}}, sk^{\mathsf{Sig}}_{\mathbf{x}}, \sigma^{\mathsf{Sig}}_{\mathbf{x}}, crs^{\mathsf{NIZK}}\big)$ and a message $m \in \mathcal{M}^{\mathsf{ACE}_{\mathsf{DEq}}}$, the algorithm $\mathsf{ACE}_{\mathsf{DEq}}.\mathsf{Enc}$ samples randomness $r_1, \ldots, r_\ell$ and computes

$$\begin{aligned}
c_i &\leftarrow \mathsf{ACE}_=.\mathsf{Enc}\big(ek^{\mathsf{ACE}_=}_{(x_i,i)}, m; r_i\big) \quad (\text{for } i \in [\ell]),\\
\sigma^{\mathsf{Sig}}_c &\leftarrow \mathsf{Sig}.\mathsf{Sign}\big(sk^{\mathsf{Sig}}_{\mathbf{x}}, [c_1, \ldots, c_\ell]\big),\\
\pi^{\mathsf{NIZK}} &\leftarrow \mathsf{NIZK}.\mathsf{Prove}\big(crs^{\mathsf{NIZK}}, x := \big(vk^{\mathsf{Sig}}, c_1, \ldots, c_\ell\big),\\
&\qquad w := \big(ek^{\mathsf{ACE}_=}_{(x_1,1)}, \ldots, ek^{\mathsf{ACE}_=}_{(x_\ell,\ell)}, m, r_1, \ldots, r_\ell, vk^{\mathsf{Sig}}_{\mathbf{x}}, \sigma^{\mathsf{Sig}}_{\mathbf{x}}, \sigma^{\mathsf{Sig}}_c\big)\big).
\end{aligned}$$

It outputs the ciphertext $c := \big(c_1, \ldots, c_\ell, \pi^{\mathsf{NIZK}}\big)$.

Sanitizer: On input sanitizer parameters $sp^{\mathsf{ACE}_{\mathsf{DEq}}} = \big(sp^{\mathsf{ACE}_=}, vk^{\mathsf{Sig}}, crs^{\mathsf{NIZK}}\big)$ and a ciphertext $c = \big(c_1, \ldots, c_\ell, \pi^{\mathsf{NIZK}}\big)$, the algorithm $\mathsf{ACE}_{\mathsf{DEq}}.\mathsf{San}$ first checks whether $\mathsf{NIZK}.\mathsf{Ver}\big(crs^{\mathsf{NIZK}}, x := \big(vk^{\mathsf{Sig}}, c_1, \ldots, c_\ell\big), \pi^{\mathsf{NIZK}}\big) = 1$. If this is the case, it runs $c'_i \leftarrow \mathsf{ACE}_=.\mathsf{San}\big(c_i\big)$ for $i \in [\ell]$. If $c'_i \neq \bot$ for all $i \in [\ell]$, it outputs the sanitized ciphertext $c' := \big(c'_1, \ldots, c'_\ell\big)$. If the verification fails or any of the sanitized ciphertexts is $\bot$, it outputs $\bot$.

Decrypt: On input a decryption key $dk^{\mathsf{ACE}_{\mathsf{DEq}}}_{\mathbf{y}} = \big(dk^{\mathsf{ACE}_=}_{(y_1,1)}, \ldots, dk^{\mathsf{ACE}_=}_{(y_\ell,\ell)}\big)$ and a sanitized ciphertext $c' := \big(c'_1, \ldots, c'_\ell\big)$, the algorithm $\mathsf{ACE}_{\mathsf{DEq}}.\mathsf{Dec}$ computes for $i \in [\ell]$ the message $m_i \leftarrow \mathsf{ACE}_=.\mathsf{Dec}\big(dk^{\mathsf{ACE}_=}_{(y_i,i)}, c'_i\big)$. If $m_i \neq \bot$ for some $i \in [\ell]$, $\mathsf{ACE}_{\mathsf{DEq}}.\mathsf{Dec}$ outputs the first such m_i; otherwise it outputs $\bot$.

Modification detection: On input sanitizer parameters $sp^{\mathsf{ACE}_{\mathsf{DEq}}} := \big(sp^{\mathsf{ACE}_=}, vk^{\mathsf{Sig}}, crs^{\mathsf{NIZK}}\big)$ and two ciphertexts $c = \big(c_1, \dots, c_\ell, \pi^{\mathsf{NIZK}}\big)$ and $\tilde{c} := \big(\tilde{c}_1, \dots, \tilde{c}_\ell, \tilde{\pi}^{\mathsf{NIZK}}\big)$, the algorithm $\mathsf{ACE}_{\mathsf{DEq}}.\mathsf{DMod}$ checks for $i \in [\ell]$ whether $\mathsf{ACE}_=.\mathsf{DMod}\big(sp^{\mathsf{ACE}_=}, c_i, \tilde{c}_i\big) = 1$. If this is the case for some $i \in [\ell]$, it outputs 1; otherwise, it outputs 0.

Weak and strong anonymity. As we show below, our scheme enjoys weak anonymity. It is easy to see that it does not have strong anonymity: Given a decryption key for the role $(1, 2)$, one can decrypt ciphertexts encrypted under a key for the roles $(1, 1)$ and $(2, 2)$. One does, however, also learn which of the two components decrypted successfully. If it is the first one, the sender role must be $(1, 1)$, if it is the second one, the sender role must be $(2, 2)$.

A similar construction can be used to achieve strong anonymity for less expressive policies: If a sender role still corresponds to a vector $(x_1, \dots, x_\ell) \in \mathcal{D}^\ell$ but a receiver role only to one component $(j, y) \in [\ell] \times \mathcal{D}$, one can consider the policy that allows to receive if $x_j = y$. Now, we do not need several components for the decryption key and the problem sketched above disappears.

We first show that correctness and detectability of our scheme is implied by the correctness and detectability of the underlying schemes.

Lemma 6.9. *Let $\mathsf{ACE}_{\mathsf{DEq}}$ be the scheme from above and let $\mathcal{A}$ be a probabilistic algorithm. Then, there exist probabilistic algorithms $\mathcal{A}_{\mathsf{corr}}$, $\mathcal{A}_{\mathsf{dtct}}$, and $\mathcal{A}'_{\mathsf{dtct}}$ such that*

$$\mathsf{Adv}^{\mathsf{ACE\text{-}corr}}_{\mathsf{ACE}_{\mathsf{DEq}},\mathcal{A}} \leq \mathsf{Adv}^{\mathsf{ACE\text{-}corr}}_{\mathsf{ACE}_=,\mathcal{A}_{\mathsf{corr}}} + (\ell - 1) \cdot \mathsf{Adv}^{\mathsf{ACE\text{-}dtct}}_{\mathsf{ACE}_=,\mathcal{A}_{\mathsf{dtct}}},$$
$$\mathsf{Adv}^{\mathsf{ACE\text{-}dtct}}_{\mathsf{ACE}_{\mathsf{DEq}},\mathcal{A}} \leq \ell \cdot \mathsf{Adv}^{\mathsf{ACE\text{-}dtct}}_{\mathsf{ACE}_=,\mathcal{A}'_{\mathsf{dtct}}}.$$

Proof. We first prove correctness. Let $(m, \mathbf{x}, \mathbf{y})$ with $P_{\mathsf{DEq}}(\mathbf{x}, \mathbf{y}) = 1$ be the output of $\mathcal{A}$ in an execution of $\mathsf{Exp}^{\mathsf{ACE\text{-}corr}}_{\mathsf{ACE}_{\mathsf{DEq}},\mathcal{A}}$. Correctness of the signature scheme and completeness of the NIZK imply that the verification in the sanitizer algorithm succeeds with probability 1. Note that $P_{\mathsf{DEq}}(\mathbf{x}, \mathbf{y}) = 1$ implies that there exists $i \in [\ell]$ with $x_i = y_i$. Let i_0 be the first such i. Then, $\mathcal{A}$ only wins the correctness game if either $\mathsf{ACE}_=.\mathsf{Dec}\big(dk^{\mathsf{ACE}_=}_{(y_{i_0}, i_0)}, c'_{i_0}\big) \neq m$, or $\mathsf{ACE}_=.\mathsf{Dec}\big(dk^{\mathsf{ACE}_=}_{(y_i, i)}, c'_i\big) \neq \perp$ for some $i < i_0$. The probability of the former event is bounded by $\mathsf{Adv}^{\mathsf{ACE\text{-}corr}}_{\mathsf{ACE}_=,\mathcal{A}_{\mathsf{corr}}}$ where $\mathcal{A}_{\mathsf{corr}}$ emulates this experiment and returns $\big(m, (x_{i_0}, i_0), (y_{i_0}, i_0)\big)$. For the latter event, note that there are at most $\ell - 1$ such i, so the probability that $\mathsf{ACE}_=.\mathsf{Dec}$ returns a message different from $\perp$ for any of them can be bounded by $(\ell - 1) \cdot \mathsf{Adv}^{\mathsf{ACE\text{-}dtct}}_{\mathsf{ACE}_{\mathsf{DEq}},\mathcal{A}_{\mathsf{dtct}}}$ for the adversary $\mathcal{A}_{\mathsf{dtct}}$ that emulates the experiment and returns $\big(m, (x_i, i), (y_i, i)\big)$ for a uniformly chosen $i < i_0$.

For detectability, the adversary $\mathcal{A}'_{\mathsf{dtct}}$ emulates an execution of $\mathsf{Exp}^{\mathsf{ACE\text{-}dtct}}_{\mathsf{ACE}_{\mathsf{DEq}},\mathcal{A}}$ and when $\mathcal{A}$ returns $(m, \mathbf{x}, \mathbf{y})$, $\mathcal{A}'_{\mathsf{dtct}}$ outputs $(m, (x_i, i), (y_i, i))$ for a uniformly chosen $i \in \{1, \dots, \ell\}$. Note that $\mathcal{A}$ only wins if $P_{\mathsf{DEq}}(\mathbf{x}, \mathbf{y}) = 0$, which implies that $x_i \neq y_i$ for all $i \in [\ell]$. In this case, $\mathcal{A}$ wins if any of the ciphertext components decrypt to something different from $\perp$. Thus, $\mathcal{A}'_{\mathsf{dtct}}$ also wins if the component i was guesses correctly, which happens with probability $1/\ell$. □

The following theorem summarizes the security properties we prove for our scheme.

Theorem 6.10 (Informal)**.** *The lifted ACE scheme is secure, i.e., all efficient adversaries have only negligible advantage in breaking the privacy, (weak) anonymity, sanitization, role-respecting, uniform decryption, or ciphertext-unpredictability properties, if the underlying ACE scheme for equality is secure, the signature scheme is unforgeable, the proof system provides zero-knowledge and extractability, and if the function F is pseudo-random.*

We prove this theorem in a sequence of theorems proving the individual properties. We begin by showing that privacy and weak anonymity of the scheme follow from the corresponding properties of the underlying scheme for equality and the zero-knowledge property of the NIZK. Note that security of the PRF is not needed for this step since it is only used for the signatures, which are irrelevant here.

Theorem 6.11. *Let $\mathsf{ACE}_{\mathsf{DEq}}$, be the scheme from above, let $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ be a probabilistic algorithm. Then, there exist probabilistic algorithms $\mathcal{A}_{\mathsf{PRF}}$, $\mathcal{A}_{\mathsf{ZK}}$, $\mathcal{A}_{\mathsf{ACE}}$, $\mathcal{A}'_{\mathsf{PRF}}$, $\mathcal{A}'_{\mathsf{ZK}}$, and $\mathcal{A}'_{\mathsf{ACE}}$ (which are all roughly as efficient as emulating an execution of $\mathsf{Exp}^{\mathsf{ACE\text{-}privAnon\text{-}CCA}}_{\mathsf{ACE}_{\mathsf{DEq}},\mathcal{A}}$) such that*

$$\mathsf{Adv}^{\mathsf{ACE\text{-}priv\text{-}CCA}}_{\mathsf{ACE}_{\mathsf{DEq}},\mathcal{A}} = 2 \cdot \mathsf{Adv}^{\mathsf{NIZK\text{-}ZK}}_{\mathsf{NIZK},\mathcal{A}_{\mathsf{ZK}}} + \ell \cdot \mathsf{Adv}^{\mathsf{ACE\text{-}priv\text{-}CCA}}_{\mathsf{ACE}_{=},\mathcal{A}_{\mathsf{ACE}}},$$
$$\mathsf{Adv}^{\mathsf{ACE\text{-}wAnon\text{-}CCA}}_{\mathsf{ACE}_{\mathsf{DEq}},\mathcal{A}} = 2 \cdot \mathsf{Adv}^{\mathsf{NIZK\text{-}ZK}}_{\mathsf{NIZK},\mathcal{A}'_{\mathsf{ZK}}} + \ell \cdot \mathsf{Adv}^{\mathsf{ACE\text{-}wAnon\text{-}CCA}}_{\mathsf{ACE}_{=},\mathcal{A}'_{\mathsf{ACE}}}.$$

Proof. We only prove the statement about the privacy advantage. The proof for weak anonymity is completely analogous. We assume without loss of generality that $\mathcal{A}$ ensures $\mathbf{x}^0 = \mathbf{x}^1$ and $P(\mathbf{x}^0, \mathbf{y}) = 0$ for all $\mathbf{y} \in J$, since doing otherwise can only decrease the privacy advantage. Let $H_0 := \mathsf{Exp}^{\mathsf{ACE\text{-}privAnon\text{-}CCA}}_{\mathsf{ACE}_{\mathsf{DEq}},\mathcal{A}}$ and let H_1 be as H_0 where we replace $crs^{\mathsf{NIZK}} \leftarrow \mathsf{NIZK.Gen}(1^\kappa)$ by $\big(crs^{\mathsf{NIZK}}, \tau^{\mathsf{NIZK}}\big) \leftarrow S_1^{\mathsf{NIZK}}(1^\kappa)$ in $\mathsf{ACE}_{\mathsf{DEq}}.\mathsf{Setup}$, and for the generation of the challenge ciphertext c^*, we replace $\pi^{\mathsf{NIZK}} \leftarrow \mathsf{NIZK.Prove}\big(crs^{\mathsf{NIZK}}, x, w\big)$ in $\mathsf{ACE}_{\mathsf{DEq}}.\mathsf{Enc}$ by $\pi^{\mathsf{NIZK}} \leftarrow S_2^{\mathsf{NIZK}}\big(crs^{\mathsf{NIZK}}, \tau^{\mathsf{NIZK}}, x\big)$. It can be shown as in the proof of Theorem 6.3 that there exist probabilistic algorithms $\mathcal{A}_{\mathsf{PRF}}$ and $\mathcal{A}_{\mathsf{ZK}}$ such that

$$\Pr^{H_0}[b' = b] - \Pr^{H_1}[b' = b] = \mathsf{Adv}^{\mathsf{NIZK\text{-}ZK}}_{\mathsf{NIZK},\mathcal{A}_{\mathsf{ZK}}}. \tag{16}$$

For $k \in \{0, \dots, \ell\}$, we define $H_{2,k}$ as follows. It is identical to H_1 except that after $\mathcal{A}$ returns $(m_0, m_1, \mathbf{x}^0, \mathbf{x}^1, st)$, we replace the ciphertext components in c^*

$$c_i \leftarrow \mathsf{ACE}_{=}.\mathsf{Enc}\big(ek^{\mathsf{ACE}_=}_{(x_i^0, i)}, m_0; r_i\big) \quad (\text{for } 1 \le i \le k),$$
$$c_i \leftarrow \mathsf{ACE}_{=}.\mathsf{Enc}\big(ek^{\mathsf{ACE}_=}_{(x_i^1, i)}, m_1; r_i\big) \quad (\text{for } k < i \le \ell).$$

Note that $H_{2,0}$ corresponds to H_1 with $b = 1$ and $H_{2,\ell}$ corresponds to H_1 with $b = 0$. Now consider the adversary $\mathcal{A}_{\mathsf{ACE}}$ that on input sp chooses $k_0 \twoheadleftarrow \{1, \dots, \ell\}$ uniformly at random and emulates an execution of H_1. It emulates the oracle $\mathcal{O}_G$ by obtaining all the required sub-keys from its own oracle $\mathcal{O}_G$. To emulate the oracle $\mathcal{O}_{SD}$, it first checks the NIZK proof as $\mathsf{ACE}_{\mathsf{DEq}}.\mathsf{San}$ and if the verification succeeds, it uses its oracle $\mathcal{O}_{SD}$ to sanitize and decrypt all ciphertext components. As $\mathsf{ACE}_{\mathsf{DEq}}.\mathsf{Dec}$, it outputs the first message different from $\perp$, or $\perp$ if no such message exists.

When $\mathcal{A}$ returns $(m_0, m_1, \mathbf{x}^0, \mathbf{x}^1, st)$, $\mathcal{A}_{\mathsf{ACE}}$ generates the challenge ciphertext c^* by encrypting m_0 under the key $ek^{\mathsf{ACE}_=}_{(x_i^0, i)}$ to obtain c_i for $1 \le i < k_0$, and by encrypting m_1 under the key $ek^{\mathsf{ACE}_=}_{(x_i^1, i)}$ for $k_0 < i \le \ell$, where these keys can obtain from $\mathcal{O}_G$ without changing the advantage. For the k_0-th component, it returns $\big(m_0, m_1, x^0_{k_0}, x^1_{k_0}\big)$ to the challenger and uses the obtained challenge ciphertext as c_{k_0}. It then proceeds with the emulation of H_1. It emulates the oracle $\mathcal{O}_G$ as above and the oracle $\mathcal{O}_{SD^*}$ as $\mathcal{O}_{SD}$ with the difference that if its own oracle returns `test` for any of the components, it returns `test` as well. Finally, when $\mathcal{A}_2$ returns b', $\mathcal{A}_{\mathsf{ACE}}$ returns $b'' \leftarrow b'$. Note that if $b = 0$ or $b = 1$, $\mathcal{A}_{\mathsf{ACE}}$ perfectly emulates an execution of H_{2,k_0} or H_{2,k_0-1}, respectively. Further note that since $\mathcal{A}$ by assumption does not query $\mathcal{O}_G$ on a decryption key for any $\mathbf{y}$ with $P(\mathbf{x}^0, \mathbf{y}) = 1$, $\mathcal{A}_{\mathsf{ACE}}$ also does not ask for a decryption that could decrypt the challenger ciphertext. Hence, $\mathcal{A}_{\mathsf{ACE}}$ wins if $b'' = b$ and we have

$$\begin{aligned}
\mathsf{Adv}^{\mathsf{ACE\text{-}priv\text{-}CCA}}_{\mathsf{ACE}_{=},\mathcal{A}_{\mathsf{ACE}}} &= 2 \cdot \Pr^{\mathsf{Exp}^{\mathsf{ACE\text{-}privAnon\text{-}CCA}}_{\mathsf{ACE}_{=},\mathcal{A}_{\mathsf{ACE}}}}[b'' = b] - 1 \\
&= \Pr^{\mathsf{Exp}^{\mathsf{ACE\text{-}privAnon\text{-}CCA}}_{\mathsf{ACE}_{=},\mathcal{A}_{\mathsf{ACE}}}}[b'' = 1 \mid b = 1] - \Pr^{\mathsf{Exp}^{\mathsf{ACE\text{-}privAnon\text{-}CCA}}_{\mathsf{ACE}_{=},\mathcal{A}_{\mathsf{ACE}}}}[b'' = 1 \mid b = 0] \\
&= \sum_{k=1}^{\ell} \frac{1}{\ell} \Pr^{H_{2,k-1}}[b' = 1] - \sum_{k=1}^{\ell} \frac{1}{\ell} \Pr^{H_{2,k}}[b' = 1] \\
&= \big(\Pr^{H_{2,0}}[b' = 1] - \Pr^{H_{2,\ell}}[b' = 1]\big)/\ell \\
&= \big(\Pr^{H_1}[b' = 1 \mid b = 1] - \Pr^{H_1}[b' = 1 \mid b = 0]\big)/\ell.
\end{aligned}$$

We therefore have that $2 \cdot \Pr^{H_1}[b' = 1] - 1 = \ell \cdot \mathsf{Adv}^{\mathsf{ACE\text{-}priv\text{-}CCA}}_{\mathsf{ACE}_{=},\mathcal{A}_{\mathsf{ACE}}}$. Combining this with equation (16) concludes the proof. $\square$

Next, we prove sanitization security, which directly follows from the sanitization security of the underlying scheme for equality.

Theorem 6.12. *Let $\mathsf{ACE}_{\mathsf{DEq}}$, be the scheme from above and let $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ be an attacker on the the sanitization security. Then, there exists a probabilistic algorithm $\mathcal{A}'$ (which is roughly as efficient as emulating an execution of $\mathsf{Exp}^{\mathsf{ACE\text{-}san\text{-}CCA}}_{\mathsf{ACE}_{\mathsf{DEq}},\mathcal{A}}$) such that*

$$\mathsf{Adv}^{\mathsf{ACE\text{-}san\text{-}CCA}}_{\mathsf{ACE}_{\mathsf{DEq}},\mathcal{A}} \leq \ell \cdot \mathsf{Adv}^{\mathsf{ACE\text{-}san\text{-}CCA}}_{\mathsf{ACE}_{=},\mathcal{A}'}.$$

Proof. For $k \in \{0, \ldots, \ell\}$, we define the hybrid H_k as follows. It is identical to $\mathsf{Exp}^{\mathsf{ACE\text{-}san\text{-}CCA}}_{\mathsf{ACE}_{\mathsf{DEq}},\mathcal{A}}$ until $\mathcal{A}_1$ returns $c^0 = \big(c^0_1, \ldots, c^0_\ell, \pi^{\mathsf{NIZK}}_0\big)$, $c^1 = \big(c^1_1, \ldots, c^1_\ell, \pi^{\mathsf{NIZK}}_1\big)$, and st. Then, c'^0 and c'^1 are obtained as before by sanitizing the given ciphertexts, but $\mathcal{A}_2$ instead of c'^b receives $\big(c'_1, \ldots, c'_\ell\big)$ with $c'_i \leftarrow \mathsf{ACE}_{=}.\mathsf{San}\big(c^0_i\big)$ for $1 \leq i \leq k$ and $c'_i \leftarrow \mathsf{ACE}_{=}.\mathsf{San}\big(c^1_i\big)$ for $k < i \leq \ell$. Note that H_0 is equal to $\mathsf{Exp}^{\mathsf{ACE\text{-}san\text{-}CCA}}_{\mathsf{ACE}_{\mathsf{DEq}},\mathcal{A}}$ with $b = 1$ if $c'^1 \neq \perp$, and H_ℓ is equal to $\mathsf{Exp}^{\mathsf{ACE\text{-}san\text{-}CCA}}_{\mathsf{ACE}_{\mathsf{DEq}},\mathcal{A}}$ with $b = 0$ if $c'^0 \neq \perp$.

Now consider the adversary $\mathcal{A}'$ that on input $sp^{\mathsf{ACE}_{=}}$ chooses $k_0 \twoheadleftarrow \{1, \ldots \ell\}$ uniformly at random and emulates an execution of $\mathsf{Exp}^{\mathsf{ACE\text{-}san\text{-}CCA}}_{\mathsf{ACE}_{\mathsf{DEq}},\mathcal{A}}$. The oracle queries by $\mathcal{A}$ are answered using the oracles of $\mathcal{A}'$. It gives the sanitized ciphertext $\big(c'_1, \ldots, c'_\ell\big)$ to $\mathcal{A}_2$ where $c'_i \leftarrow \mathsf{ACE}_{=}.\mathsf{San}\big(c^0_i\big)$ for $1 \leq i < k_0$, $c'_i \leftarrow \mathsf{ACE}_{=}.\mathsf{San}\big(c^1_i\big)$ for $k_0 < i \leq \ell$, and c'_{k_0} is obtained from the challenger by submitting $\big(c^0_{k_0}, c^1_{k_0}\big)$. When $\mathcal{A}_2$ returns a bit b', $\mathcal{A}'$ returns $b'' \leftarrow b'$. Note that if $b = 0$, $\mathcal{A}'$ perfectly emulates H_{k_0}, and if $b = 1$, $\mathcal{A}'$ perfectly emulates H_{k_0-1}. Further note that if $\mathcal{A}$ wins, then $c'^0 \neq \perp \neq c'^1$ and $m_{0,\mathbf{y}} = m_{1,\mathbf{y}} = \perp$ for all $\mathbf{y} \in J$. Since a sanitized ciphertext is only not $\perp$ if all components do not sanitize to $\perp$, and a message is $\perp$ if all components are, this means that the ciphertext components submitted by $\mathcal{A}'$ also satisfy the winning condition if the ciphertexts from $\mathcal{A}$ do. Hence, we can conclude that $\mathsf{Adv}^{\mathsf{ACE\text{-}san\text{-}CCA}}_{\mathsf{ACE}_{\mathsf{DEq}},\mathcal{A}} \leq \ell \cdot \mathsf{Adv}^{\mathsf{ACE\text{-}san\text{-}CCA}}_{\mathsf{ACE}_{=},\mathcal{A}'}$. $\square$

Ciphertext unpredictability directly follows from ciphertext unpredictability of the underlying ACE scheme.

Theorem 6.13. *Let $\mathsf{ACE}_{\mathsf{DEq}}$, be the scheme from above and let $\mathcal{A}$ be an attacker on the the ciphertext unpredictability. Then, there exists a probabilistic algorithm $\mathcal{A}'$ (which is roughly as efficient as emulating an execution of $\mathsf{Exp}^{\mathsf{ACE\text{-}ctxt\text{-}unpred}}_{\mathsf{ACE}_{\mathsf{DEq}},\mathcal{A}}$) such that*

$$\mathsf{Adv}^{\mathsf{ACE\text{-}ctxt\text{-}unpred}}_{\mathsf{ACE}_{\mathsf{DEq}},\mathcal{A}} \leq \ell \cdot \mathsf{Adv}^{\mathsf{ACE\text{-}ctxt\text{-}unpred}}_{\mathsf{ACE}_{=},\mathcal{A}'}.$$

Proof. Let $\mathcal{A}'$ emulate an execution of $\mathsf{Exp}^{\mathsf{ACE\text{-}ctxt\text{-}unpred}}_{\mathsf{ACE}_{\mathsf{DEq}},\mathcal{A}}$, using $\mathcal{O}_G$ to answer oracle queries from $\mathcal{A}$. When $\mathcal{A}$ returns $\big(m, \mathbf{x}, c = \big(c_1, \ldots, c_\ell, \pi^{\mathsf{NIZK}}\big)\big)$, $\mathcal{A}'$ chooses $k \twoheadleftarrow \{1, \ldots, \ell\}$ uniformly at random, and returns $\big(m, (x_k, k), c_k\big)$. If $\mathcal{A}$ wins, c is detected as a modification of a fresh encryption of m under $\mathbf{x}$. Since encryption and modification detection are defined component-wise, this means that there exists a component k_0 such that c_{k_0} is detected to be a modification of a fresh encryption of m under (x_{k_0}, k_0). Hence, $\mathcal{A}'$ also wins if additionally $k = k_0$, which happens with probability $1/\ell$. □

We finally prove role-respecting and uniform decryption security.

Theorem 6.14. *Let $\mathsf{ACE}_{\mathsf{DEq}}$, be the scheme from above and let $\mathcal{A}$ be a probabilistic algorithm that makes at most at most q_E queries to the oracle $\mathcal{O}_E$. Then, there exist probabilistic algorithms $\mathcal{A}_{\mathsf{PRF}}$, $\mathcal{A}_{\mathsf{ZK}_1}$, $\mathcal{A}_{\mathsf{ZK}_2}$, $\mathcal{A}_{\mathsf{Sig}}$, and $\mathcal{A}_{\mathsf{ACE}}$ (which are all roughly as efficient as emulating an execution of $\mathsf{Exp}^{\mathsf{ACE\text{-}URR}}_{\mathsf{ACE}_{\mathsf{DEq}},\mathcal{A}}$) such that*

$$\begin{aligned}\mathsf{Adv}^{\mathsf{ACE\text{-}RR}}_{\mathsf{ACE}_{\mathsf{DEq}},\mathcal{A}} + \mathsf{Adv}^{\mathsf{ACE\text{-}uDec}}_{\mathsf{ACE}_{\mathsf{DEq}},\mathcal{A}} \leq 2 \cdot \mathsf{Adv}^{\mathsf{PRF}}_{F,\mathcal{A}_{\mathsf{PRF}}} + 2 \cdot \mathsf{Adv}^{\mathsf{NIZK\text{-}ext}_1}_{\mathsf{NIZK},\mathcal{A}_{\mathsf{ZK}_1}} + 2 \cdot \mathsf{Adv}^{\mathsf{NIZK\text{-}ext}_2}_{\mathsf{NIZK},\mathcal{A}_{\mathsf{ZK}_2}} \\ + 2(q_E + 1) \cdot \mathsf{Adv}^{\mathsf{Sig\text{-}EUF\text{-}CMA}}_{\mathsf{Sig},\mathcal{A}_{\mathsf{Sig}}} + 2\ell \cdot \Big(\mathsf{Adv}^{\mathsf{ACE\text{-}RR}}_{\mathsf{ACE}_{=},\mathcal{A}_{\mathsf{ACE}}} + \mathsf{Adv}^{\mathsf{ACE\text{-}uDec}}_{\mathsf{ACE}_{=},\mathcal{A}_{\mathsf{ACE}}}\Big).\end{aligned}$$

Proof Sketch. As in the proof of Theorem 6.8, we define hybrids $H_0 \coloneqq \mathsf{Exp}^{\mathsf{ACE\text{-}URR}}_{\mathsf{ACE}_{\mathsf{DEq}},\mathcal{A}}$, H_1 as H_0 where F_K is replaced by a uniform random function U, H_2 as H_1 where crs^{NIZK} is generated by E_1^{NIZK}, H_3 as H_2 where a witness $w = \big(ek^{\mathsf{ACE}_=}_{(x_1,1)}, \ldots, ek^{\mathsf{ACE}_=}_{(x_\ell,\ell)}, m, r_1, \ldots, r_\ell, vk^{\mathsf{Sig}}_{\mathbf{x}}, \sigma^{\mathsf{Sig}}_{\mathbf{x}}, \sigma^{\mathsf{Sig}}_c\big)$ is extracted from π^{NIZK} by E_2^{NIZK} after $\mathcal{A}$ returned $c \coloneqq \big(c_1, \ldots, c_\ell, \pi^{\mathsf{NIZK}}\big)$. We can bound the probability that no valid witness is extracted even though π^{NIZK} is a valid proof by the knowledge extraction advantage of a suitable adversary, and the probability that a valid witness was extracted and the contained encryption key was not obtained via an oracle call by the signature forgery advantage of another adversary as in the proof of Theorem 6.8. If these events do not occur, the ciphertext c is an encryption of the message m under a valid key that was returned by $\mathcal{O}_G$. Hence, $\mathcal{A}$ can in this case only win the role-respecting game or the uniform decryption game if some ciphertext component violates one of these properties. We can construct an adversary $\mathcal{A}_{\mathsf{ACE}}$ that emulates the execution, guesses this component, and uses the corresponding ciphertext component to win the game for the underlying scheme for equality. □

7 Conclusion and Directions for Future Work

In this paper, we have critically revisited existing notions for access control encryption, proposed stronger security definitions, and presented a new scheme that provably achieves our strong requirements. The need for stronger notions is not only a theoretical one as we have shown: In particular, we have described a practical attack based on the observation that a semi-honest sanititzer might leak an unsanitized ciphertext to a dishonest party.

An important question is whether all realistic attacks are excluded by our definitions. Furthermore, we would like to understand the fundamental limits of ACE. This includes investigating in which scenarios it can or cannot be used. To settle these questions, the authors are currently working on a theoretical model to capture the use case of ACE in a simulation-based framework. Another interesting research direction is to find more efficient schemes for useful policies.

A Standard Cryptographic Primitives and Games

A.1 Pseudorandom Functions

Definition A.1. For $\kappa \in \mathbb{N}$, let $\mathcal{K}_\kappa$, $\mathcal{X}_\kappa$, and $\mathcal{Y}_\kappa$ be finite sets and let $F_\kappa \colon \mathcal{K}_\kappa \times \mathcal{X}_\kappa \to \mathcal{Y}_\kappa$ be a function. For $K \in \mathcal{K}_\kappa$, we use the notation $F_K := F_\kappa(K, \cdot)$. Further let $\mathcal{A}$ be a probabilistic algorithm and consider the experiment in which $\mathcal{A}$ outputs a bit after interacting with an oracle that either corresponds to F_K for a uniformly chosen $K \in \mathcal{K}_\kappa$, or to a uniformly chosen function $U \colon \mathcal{X}_\kappa \to \mathcal{Y}_\kappa$. We define the *pseudorandom function advantage* of $\mathcal{A}$ as

$$\mathsf{Adv}^{\mathsf{PRF}}_{F,\mathcal{A}} := \Pr\big[\mathcal{A}^{F_K(\cdot)}(1^\kappa) = 1\big] - \Pr\big[\mathcal{A}^{U(\cdot)}(1^\kappa) = 1\big],$$

where the first probability is over the random coins of $\mathcal{A}$ and the choice of K, and the second probability is over the random coins of $\mathcal{A}$ and the choice of U.

A.2 Decisional Diffie-Hellman

Definition A.2. Let $G = \langle g \rangle$ be a prime-order group of order q and let g be its generator. Let $\mathcal{A}$ be an adversary that on input q, g, and three elements $X, Y, T \in G$ returns a bit d. Let $\mathsf{DDH}^{\mathrm{real}}_{\mathcal{A},g,q}$ be the experiment where $\mathcal{A}$ is given two random group elements $X = g^a$, $Y = g^b$, and the value $T = g^{ab}$. Let $\mathsf{DDH}^{\mathrm{rand}}_{\mathcal{A},g,q}$ be the experiment where $\mathcal{A}$ is given three random group elements $X = g^a$, $Y = g^b$, and $T = g^c$. We define the *decisional Diffie-Hellman (DDH) advantage* of $\mathcal{A}$ as

$$\mathsf{Adv}^{\mathsf{DDH}}_{\mathcal{A},g,q} := \Pr^{\mathsf{DDH}^{\mathrm{real}}_{\mathcal{A},g,q}}[d = 1] - \Pr^{\mathsf{DDH}^{\mathrm{rand}}_{\mathcal{A},g,q}}[d = 1].$$

A.3 Public-Key Encryption

Definition A.3. A *public-key encryption (PKE) scheme* consist of the following three PPT algorithms:

Key Generation: The algorithm Gen on input a security parameter 1^κ, outputs a *public key* ek and a *private key* dk.

Encryption: The algorithm Enc on input a public key ek and a message $m \in \mathcal{M}$, outputs a *ciphertext* c.

Decryption: The algorithm Dec on input a private key dk and a ciphertext c, outputs a message $m \in \mathcal{M} \cup \{\perp\}$.

We require for all (ek, dk) in the range of Gen and all $m \in \mathcal{M}$ that

$$\mathsf{Dec}\big(dk, \mathsf{Enc}(ek, m)\big) = m$$

with probability 1.

Definition A.4. Let $\mathcal{E} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ be a PKE scheme and let $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ be a pair of probabilistic algorithm.s Consider the experiment $\mathsf{Exp}^{\mathsf{PKE\text{-}IND\text{-}CPA}}_{\mathcal{E},\mathcal{A}}$ in Figure 4. We define the *ciphertext indistinguishability under chosen-plaintext attacks advantage* of $\mathcal{A}$ as

$$\mathsf{Adv}^{\mathsf{PKE\text{-}IND\text{-}CPA}}_{\mathcal{E},\mathcal{A}} := 2 \cdot \Pr\big[b' = b \ \wedge \ |m_0| = |m_1|\big] - 1$$

where the probability is over the randomness in $\mathsf{Exp}^{\mathsf{PKE\text{-}IND\text{-}CPA}}_{\mathcal{E},\mathcal{A}}$.

Experiment $\mathsf{Exp}^{\mathsf{PKE\text{-}IND\text{-}CPA}}_{\mathcal{E},\mathcal{A}}$

Input: 1^κ
$(ek, dk) \leftarrow \mathsf{Gen}(1^\kappa)$
$(m_0, m_1, st) \leftarrow \mathcal{A}_1(ek)$
$b \twoheadleftarrow \{0,1\}$
$c^* \leftarrow \mathsf{Enc}(ek, m_b)$
$b' \leftarrow \mathcal{A}_2(st, c^*)$

Experiment $\mathsf{Exp}^{\mathsf{Sig\text{-}EUF\text{-}CMA}}_{\mathcal{E},\mathcal{A}}$

Input: 1^κ
$(vk, sk) \leftarrow \mathsf{Gen}(1^\kappa)$
$(m, \sigma) \leftarrow \mathcal{A}^{\mathsf{Sign}(sk,\cdot)}(vk)$

Figure 4: Experiments for the security definitions of public-key encryption and digital signature schemes.

A.4 Digital Signature Schemes

Definition A.5. A *(digital) signature scheme* consist of the following three PPT algorithms:

Key Generation: The algorithm Gen on input a security parameter 1^κ, outputs a *public key* vk and a *private key* sk.

Signing: The algorithm Sign on input a private key sk and a message $m \in \mathcal{M}$, outputs a *signature* σ.

Verification: The algorithm Ver is deterministic and on input a public key vk, a message m, and a signature σ, outputs a bit b (where $b = 1$ means "valid" and $b = 0$ means "invalid").

We require for all (vk, sk) in the range of Gen and all $m \in \mathcal{M}$ that

$$\mathsf{Ver}\big(vk, m, \mathsf{Sign}(sk, m)\big) = 1$$

with probability 1.

Definition A.6. Let $\mathcal{E} = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Ver})$ be a signature scheme and let $\mathcal{A}$ be a probabilistic algorithm. Consider the experiment $\mathsf{Exp}^{\mathsf{Sig\text{-}EUF\text{-}CMA}}_{\mathcal{E},\mathcal{A}}$ in Figure 4 and let Q be the set of queries $\mathcal{A}$ issued to its oracle. We define the *existential unforgeability under adaptive chosen-message attacks advantage* of $\mathcal{A}$ as

$$\mathsf{Adv}^{\mathsf{Sig\text{-}EUF\text{-}CMA}}_{\mathcal{E},\mathcal{A}} := \Pr\big[\mathsf{Ver}(vk, m, \sigma) = 1 \ \wedge \ m \notin Q\big],$$

where the probability is over the randomness in $\mathsf{Exp}^{\mathsf{Sig\text{-}EUF\text{-}CMA}}_{\mathcal{E},\mathcal{A}}$.

A.5 Non-Interactive Zero-Knowledge Proofs

We define non-interactive zero-knowledge proofs following Groth [Gro06].

Definition A.7. Let R be an efficiently computable binary relation and consider the *language* $L := \{x \mid \exists w \ (x, w) \in R\}$. A *non-interactive proof system* for L consists of the following three PPT algorithms:

Key Generation: The algorithm Gen on input a security parameter 1^κ, outputs a *common reference string* crs.

Proving: The algorithm Prove on input a common reference string crs, a *statement* x, and a *witness* w, outputs a *proof* π.

Verification: The algorithm Ver on input a common reference string crs, a statement x, and a proof π, outputs a bit b (where $b = 1$ means "accept" and $b = 0$ means "reject").

We require *perfect completeness*, i.e., for all crs in the range of Gen and for all $(x, w) \in R$, we have

$$\mathsf{Ver}\big(crs, x, \mathsf{Prove}(crs, x, w)\big) = 1$$

with probability 1.

Definition A.8 (Soundness)**.** Let $\mathcal{E} = (\mathsf{Gen}, \mathsf{Prove}, \mathsf{Ver})$ be a non-interactive proof system for a language L and let $\mathcal{A}$ be a probabilistic algorithm. We define the *soundness advantage* of $\mathcal{A}$ as

$$\mathsf{Adv}^{\mathsf{NIZK\text{-}snd}}_{\mathcal{E},\mathcal{A}} := \Pr^{crs \leftarrow \mathsf{Gen}(1^\kappa);\ (x,\pi) \leftarrow \mathcal{A}(crs)}\big[x \notin L \ \wedge\ \mathsf{Ver}(crs, x, \pi) = 1\big].$$

Definition A.9 ((Unbounded) computational zero-knowledge)**.** A *non-interactive zero-knowledge (NIZK) proof system* for a relation R is a non-interactive proof system $\mathcal{E} = (\mathsf{Gen}, \mathsf{Prove}, \mathsf{Ver})$ for R together with a pair of PPT algorithms $S = (S_1, S_2)$, called *simulator.* Let $S'(crs, \tau, x, w) = S_2(crs, \tau, x)$ for $(x, w) \in R$, and $S'(crs, \tau, x, w) = \texttt{failure}$ for $(x, w) \notin R$. We define the *zero-knowledge advantage* of a probabilistic algorithm $\mathcal{A}$ as

$$\mathsf{Adv}^{\mathsf{NIZK\text{-}ZK}}_{\mathcal{E},S,\mathcal{A}} := \Pr^{crs \leftarrow \mathsf{Gen}(1^\kappa)}\big[\mathcal{A}^{\mathsf{Prove}(crs,\cdot,\cdot)}(crs) = 1\big] - \Pr^{(crs,\tau) \leftarrow S_1(1^\kappa)}\big[\mathcal{A}^{S'(crs,\tau,\cdot,\cdot)}(crs) = 1\big].$$

Definition A.10 (Knowledge extraction)**.** A *non-interactive proof of knowledge system* for a relation R is a non-interactive proof system $\mathcal{E} = (\mathsf{Gen}, \mathsf{Prove}, \mathsf{Ver})$ for R together with a pair of PPT algorithms $E = (E_1, E_2)$, called *knowledge extractor.* We define the *knowledge extraction advantages* of a probabilistic algorithm $\mathcal{A}$ as

$$\begin{aligned}
\mathsf{Adv}^{\mathsf{NIZK\text{-}ext_1}}_{\mathcal{E},E,\mathcal{A}} &:= \Pr^{crs \leftarrow \mathsf{Gen}(1^\kappa)}\big[\mathcal{A}(crs) = 1\big] - \Pr^{(crs,\xi) \leftarrow E_1(1^\kappa)}\big[\mathcal{A}(crs) = 1\big],\\
\mathsf{Adv}^{\mathsf{NIZK\text{-}ext_2}}_{\mathcal{E},E,\mathcal{A}} &:= \Pr^{(crs,\xi) \leftarrow E_1(1^\kappa);\ (x,\pi) \leftarrow \mathcal{A}(crs);\ w \leftarrow E_2(crs,\xi,x,\pi)}\big[\mathsf{Ver}(crs, x, \pi) = 1 \ \wedge\ (x, w) \notin R\big].
\end{aligned}$$

Definition A.11 ((Unbounded) simulation soundness)**.** Let $\mathcal{E} = (\mathsf{Gen}, \mathsf{Prove}, \mathsf{Ver}, S_1, S_2)$ be a NIZK proof system for a language L and let $\mathcal{A}$ be a probabilistic algorithm. Consider the experiment $\mathsf{Exp}^{\mathsf{NIZK\text{-}sim\text{-}snd}}_{\mathcal{E},\mathcal{A}}$ that executes $(crs, \tau) \leftarrow S_1(1^\kappa)$ and $(x, \pi) \leftarrow \mathcal{A}^{S_2(crs,\tau,\cdot)}(crs)$. Further let Q be set of all (x', π') such that $\mathcal{A}$ queried x' to its oracle and received π' as a response. We define the *simulation soundness advantage* of $\mathcal{A}$ as

$$\mathsf{Adv}^{\mathsf{NIZK\text{-}sim\text{-}snd}}_{\mathcal{E},\mathcal{A}} := \Pr\big[(x, \pi) \notin Q \ \wedge\ x \notin L \ \wedge\ \mathsf{Ver}(crs, x, \pi) = 1\big].$$

Note that in the above definition, $\mathcal{A}$ is allowed to issue queries $x' \notin L$ to its oracle. This means that soundness is preserved even if an adversary sees simulated proofs of false statements.

References

[ABN10] M. Abdalla, M. Bellare, and G. Neven, "Robust encryption," in *Theory of Cryptography: 7th Theory of Cryptography Conference, TCC 2010, Zurich, Switzerland, February 9-11, 2010. Proceedings*, D. Micciancio, Ed., Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 480–497. DOI: 10.1007/978-3-642-11799-2_28.

[BL73] D. E. Bell and L. J. LaPadula, "Secure computer systems: Mathematical foundations," MITRE, Tech. Rep. MTR-2547, 1973.

[BBDP01] M. Bellare, A. Boldyreva, A. Desai, and D. Pointcheval, "Key-privacy in public-key encryption," in *Advances in Cryptology — ASIACRYPT 2001: 7th International Conference on the Theory and Application of Cryptology and Information Security Gold Coast, Australia, December 9–13, 2001 Proceedings*, C. Boyd, Ed., Berlin, Heidelberg: Springer Berlin Heidelberg, 2001, pp. 566–582. DOI: 10.1007/3-540-45682-1_33.

[BSW11] D. Boneh, A. Sahai, and B. Waters, "Functional encryption: Definitions and challenges," in *Theory of Cryptography: 8th Theory of Cryptography Conference, TCC 2011, Providence, RI, USA, March 28-30, 2011. Proceedings*, Y. Ishai, Ed., Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 253–273. DOI: 10.1007/978-3-642-19571-6_16.

[CKN03] R. Canetti, H. Krawczyk, and J. B. Nielsen, "Relaxing chosen-ciphertext security," in *Advances in Cryptology - CRYPTO 2003: 23rd Annual International Cryptology Conference, Santa Barbara, California, USA, August 17-21, 2003. Proceedings*, D. Boneh, Ed., Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, pp. 565–582. DOI: 10.1007/978-3-540-45146-4_33.

[DHO16] I. Damgård, H. Haagh, and C. Orlandi, "Access control encryption: Enforcing information flow with cryptography," in *Theory of Cryptography: 14th International Conference, TCC 2016-B, Beijing, China, October 31-November 3, 2016, Proceedings, Part II*, M. Hirt and A. Smith, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2016, pp. 547–576. DOI: 10.1007/978-3-662-53644-5_21.

[Elg85] T. Elgamal, "A public key cryptosystem and a signature scheme based on discrete logarithms," *IEEE Transactions on Information Theory*, vol. 31, no. 4, pp. 469–472, Jul. 1985. DOI: 10.1109/TIT.1985.1057074.

[FLPQ13] P. Farshim, B. Libert, K. G. Paterson, and E. A. Quaglia, "Robust encryption, revisited," in *Public-Key Cryptography – PKC 2013: 16th International Conference on Practice and Theory in Public-Key Cryptography, Nara, Japan, February 26 – March 1, 2013. Proceedings*, K. Kurosawa and G. Hanaoka, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 352–368. DOI: 10.1007/978-3-642-36362-7_22.

[FGKO17] G. Fuchsbauer, R. Gay, L. Kowalczyk, and C. Orlandi, "Access control encryption for equality, comparison, and more," in *Public-Key Cryptography – PKC 2017: 20th IACR International Conference on Practice and Theory in Public-Key Cryptography, Amsterdam, The Netherlands, March 28-31, 2017, Proceedings, Part II*, S. Fehr, Ed., Berlin, Heidelberg: Springer Berlin Heidelberg, 2017, pp. 88–118. DOI: 10.1007/978-3-662-54388-7_4.

[GJJS04] P. Golle, M. Jakobsson, A. Juels, and P. Syverson, "Universal re-encryption for mixnets," in *Topics in Cryptology – CT-RSA 2004: The Cryptographers' Track at the RSA Conference 2004, San Francisco, CA, USA, February 23-27, 2004, Proceedings*, T. Okamoto, Ed., Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, pp. 163–178. DOI: 10.1007/978-3-540-24660-2_14.

[GPSW06] V. Goyal, O. Pandey, A. Sahai, and B. Waters, "Attribute-based encryption for fine-grained access control of encrypted data," in *Proceedings of the 13th ACM Conference on Computer and Communications Security*, ser. CCS '06, Alexandria, Virginia, USA: ACM, 2006, pp. 89–98. DOI: 10.1145/1180405.1180418.

[Gro04] J. Groth, "Rerandomizable and replayable adaptive chosen ciphertext attack secure cryptosystems," in *Theory of Cryptography: First Theory of Cryptography Conference, TCC 2004, Cambridge, MA, USA, February 19-21, 2004. Proceedings*, M. Naor, Ed., Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, pp. 152–170. DOI: 10.1007/978-3-540-24638-1_9.

[Gro06] ——, "Simulation-sound NIZK proofs for a practical language and constant size group signatures," in *Advances in Cryptology – ASIACRYPT 2006: 12th International Conference on the Theory and Application of Cryptology and Information Security, Shanghai, China, December 3-7, 2006. Proceedings*, X. Lai and K. Chen, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 444–459. DOI: 10.1007/11935230_29.

[Lin06] Y. Lindell, "A simpler construction of CCA2-secure public-key encryption under general assumptions," *Journal of Cryptology*, vol. 19, no. 3, pp. 359–377, 2006. DOI: 10.1007/s00145-005-0345-x.

[NY90] M. Naor and M. Yung, "Public-key cryptosystems provably secure against chosen ciphertext attacks," in *Proceedings of the Twenty-second Annual ACM Symposium on Theory of Computing*, ser. STOC '90, Baltimore, Maryland, USA: ACM, 1990, pp. 427–437. DOI: 10.1145/100216.100273.

[PR07] M. Prabhakaran and M. Rosulek, "Rerandomizable RCCA encryption," in *Advances in Cryptology - CRYPTO 2007: 27th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2007. Proceedings*, A. Menezes, Ed., Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, pp. 517–534. DOI: 10.1007/978-3-540-74143-5_29.

[Sah99] A. Sahai, "Non-malleable non-interactive zero knowledge and adaptive chosen-ciphertext security," in *40th Annual Symposium on Foundations of Computer Science*, 1999, pp. 543–553. DOI: 10.1109/SFFCS.1999.814628.

[SW05] A. Sahai and B. Waters, "Fuzzy identity-based encryption," in *Advances in Cryptology – EUROCRYPT 2005: 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Aarhus, Denmark, May 22-26, 2005. Proceedings*, R. Cramer, Ed., Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 457–473. DOI: 10.1007/11426639_27.

[TZMT17] G. Tan, R. Zhang, H. Ma, and Y. Tao, "Access control encryption based on LWE," in *Proceedings of the 4th ACM International Workshop on ASIA Public-Key Cryptography*, ser. APKC '17, Abu Dhabi, United Arab Emirates: ACM, 2017, pp. 43–50. DOI: 10.1145/3055504.3055509.