Diss. ETH No. 24607

A Constructive Treatment of Enhanced Encryption Schemes

A thesis submitted to attain the degree of

Doctor of Sciences of ETH Zurich
(Dr. sc. ETH Zurich)

presented by

Christian Matt
Dipl.-Inform. Dipl.-Math.,
Karlsruhe Institute of Technology (KIT)

born on May 12, 1986
citizen of the Federal Republic of Germany

accepted on the recommendation of

Prof. Dr. Ueli Maurer, examiner
Prof. Dr. David Basin, co-examiner
Prof. Dr. Dennis Hofheinz, co-examiner

2017

Acknowledgements

First of all, I would like to thank my advisor Ueli Maurer. He always had time for me and his abstract way of thinking led to many interesting discussions. I also thank David Basin and Dennis Hofheinz for co-refereeing this thesis.

Moreover, I want to thank Christian Badertscher, Dennis Hofheinz, and Ueli Maurer for co-authoring the papers on which this thesis is based. Further thanks go to Phil Rogaway for very helpful feedback on a draft of the paper on functional encryption.

I also thank Christian Badertscher, Daniel Jost, Chen-Da Liu Zhang, Ueli Maurer, Christopher Portmann, Renato Renner, Phil Rogaway, and Björn Tackmann for co-authoring papers outside the scope of this thesis with me during my time as a PhD student.

Last but not least, I would like to thank all the current and former members of the cryptography group at ETH that have been my colleagues for many interesting discussions and all the good times we have had together.

Abstract

Encryption is a tool that has traditionally been used to allow confidential communication between two parties. Over the years, several types of encryption have been proposed, including public-key encryption, identity-based encryption, deniable encryption, and functional encryption. These variants provide different features and security guarantees. Their security is typically defined by a game between an adversary and a challenger. Even for ordinary public-key encryption, several different security definitions have been proposed and identifying the "right" one is a nontrivial task. For more complex primitives such as functional encryption, security definitions are far more involved and it is way more difficult to evaluate whether a given definition is appropriate.

The goal of this thesis is to better understand these definitions for several types of encryption by analyzing them in the *constructive cryptography* framework. In this framework, a cryptographic primitive can be seen as providing a construction of a so-called *ideal resource* from a so-called *real resource*, for a well-defined notion of construction. The real resource formalizes what is available to the involved parties, e.g., a shared secret key and an authenticated communication channel, and the ideal resource formalizes what should be achieved by applying the encryption scheme, e.g., a secure channel that does not leak the sent messages to eavesdroppers. This paradigm makes the requirements and the achieved guarantees explicit and helps to decide whether a given scheme is suitable for a certain application.

The first and simplest encryption scheme we consider in this thesis is the one-time pad. We show that it provides a guarantee that *deniable encryption* targets, namely it remains secure if the receiver reveals the secret key to the adversary after receiving the message. We model this in constructive cryptography by allowing the receiver to become dishonest after receiving the message.

We next consider *identity-based encryption (IBE)*. In contrast to deniable encryption, it does not provide stronger security guarantees, but rather simplifies the key distribution. We formalize the standard application of IBE, namely non-interactive secure communication, as constructing an ideal resource that allows parties to be registered for an identity, and to securely sent messages to other parties only known by their identity. Quite surprisingly, we show that it is impossible to construct this resource in the standard model. We show, however, how to adjust any IBE scheme that satisfies the standard security definition to achieve this goal in the random oracle model. We also show that the impossibility result can be avoided in the standard model by considering a weaker ideal resource.

Functional encryption is a very general concept, which encompasses public-key encryption and identity-based encryption as special cases. It allows the generation of restricted secret keys that enable to learn only a specific function of the encrypted data. We formalize the security of functional encryption as constructing an ideal resource that corresponds to a repository with fine-grained access control, and compare this to existing security notions. Again, we show that constructing the most desirable ideal resource is impossible without random oracles, possible in the random oracle model, and that constructing weaker ideal resources is possible in the standard model.

Finally, we consider *access control encryption (ACE)*. While the encryption schemes discussed above allow to control which users can read the encrypted data, ACE additionally allows to restrict write access. As we argue, however, existing security notions are insufficient to provide meaningful security guarantees in realistic settings. We therefore propose new, substantially stronger security definitions and an ACE scheme that provably satisfies them under standard assumptions.

Zusammenfassung

Verschlüsselung wird traditionell verwendet, um vertrauliche Kommunikation zwischen zwei Parteien zu ermöglichen. Verschiedene Arten von Verschlüsselungsverfahren wurden in den letzten Jahren vorgeschlagen, darunter Public-Key Encryption, Identity-Based Encryption, Deniable Encryption und Functional Encryption. Diese Varianten bieten unterschiedliche Funktionen und Sicherheitsgarantien. Ihre Sicherheit ist üblicherweise durch ein Spiel zwischen einem Angreifer und einem Herausforderer definiert. Selbst für einfache Public-Key Encryption wurden verschiedene Sicherheitsdefinitionen vorgeschlagen und es ist nichttrivial, die „richtige“ Definition zu identifizieren. Sicherheitsdefinitionen für komplexere Primitive wie Functional Encryption sind deutlich involvierter und es ist noch viel schwieriger zu evaluieren, ob eine gegebene Definition adäquat ist.

Das Ziel dieser Arbeit ist es, solche Definitionen für verschiedene Arten von Verschlüsselungsverfahren besser zu verstehen, indem sie im *Constructive Cryptography Framework* analysiert werden. In diesem Framework kann man eine kryptographische Primitive so verstehen, dass sie eine sogenannte *ideale Ressource* von einer sogenannten *realen Ressource* konstruiert, für einen wohldefinierten Konstruktionsbegriff. Die reale Ressource formalisiert, was den beteiligten Parteien zur Verfügung steht, zum Beispiel ein gemeinsamer geheimer Schlüssel und ein authentischer Kommunikationskanal, und die ideale Ressource formalisiert, was durch Anwendung des Verschlüsselungsverfahrens erreicht werden soll, zum Beispiel ein sicherer Kanal, der die gesendeten Nachrichten vor Angreifern geheim hält. Dieses Paradigma macht die Voraussetzungen und die erreichten Garantien explizit und hilft bei der Entscheidung, ob ein gegebenes Verfahren für eine bestimmte Anwendung geeignet ist.

Das erste und einfachste Verschlüsselungsverfahren, das wir in dieser

Arbeit betrachten, ist das One-Time-Pad. Wir zeigen, dass es eine Garantie liefert, die ein Ziel von *Deniable Encryption* ist: Es bleibt auch dann sicher, wenn der Empfänger den Schlüssel nach Empfang der Nachricht dem Angreifer offenlegt. Wir modellieren dies in Constructive Cryptography, indem wir dem Empfänger ermöglichen, nach Erhalt der Nachricht unehrlich zu werden.

Als Nächstes betrachten wir *Identity-Based Encryption (IBE).* Im Gegensatz zu Deniable Encryption liefert es keine stärkeren Sicherheitsgarantien, sondern vereinfacht die Schlüsselverteilung. Wir formalisieren die Standardanwendung von IBE, nichtinteraktive, sichere Kommunikation, als Konstruktion einer idealen Ressource, die es Parteien erlaubt, für eine Identität registriert zu werden, und anderen Parteien, die sie nur anhand ihrer Identität kennen, sicher Nachrichten zu senden. Erstaunlicherweise können wir zeigen, dass es unmöglich ist, diese Ressource im Standardmodell zu konstruieren. Wir zeigen jedoch auch, wie man ein beliebiges IBE-Verfahren, das die Standard-Sicherheitsdefinition erfüllt, verwenden kann, um dieses Ziel im Random-Oracle-Modell zu erreichen. Wir zeigen weiter, dass man das Unmöglichkeitsresultat im Standardmodell vermeiden kann, indem man schwächere ideale Ressourcen betrachtet.

Functional Encryption ist ein sehr allgemeines Konzept, das Public-Key Encryption und Identity-Based Encryption als Spezialfälle beinhaltet. Es erlaubt, beschränkte geheime Schlüssel zu erstellen, mit denen man nur eine spezifische Funktion der verschlüsselten Daten erfahren kann. Wir formalisieren die Sicherheit von Functional Encryption als Konstruktion einer idealen Ressource, die einem Speicher mit feingranularer Zugriffskontrolle entspricht, und vergleichen dies mit existierenden Sicherheitsdefinitionen. Auch in diesem Fall zeigen wir, dass die wünschenswerteste ideale Ressource nicht ohne Random Oracles konstruiert werden kann, und dass die Konstruktion von schwächeren idealen Ressourcen im Standardmodell möglich ist.

Schliesslich betrachten wir *Access Control Encryption (ACE).* Während oben genannte Verschlüsselungsverfahren die Kontrolle darüber erlauben, wer verschlüsselte Daten lesen kann, erlaubt ACE zusätzlich den Schreibzugriff zu kontrollieren. Wie wir darlegen, sind existierende Sicherheitsdefinitionen jedoch unzureichend, um die Sicherheit in realistischen Szenarien zu garantieren. Wir schlagen deshalb neue, signifikant stärkere Sicherheitsdefinitionen zusammen mit einem ACE-Verfahren vor, das diese Definitionen unter Standardannahmen beweisbar erfüllt.

Contents

Chapter 1

Introduction

1.1 Motivation

Encryption has been used since ancient times to exchange messages confidentially. In the simplest setting of symmetric encryption, two parties share a secret key, and the sender uses an encryption algorithm that on input the secret key and a message, produces a ciphertext. This ciphertext is then sent to the other party, who uses a decryption algorithm that on input the key and the ciphertext, recovers the message. Many encryption schemes that have been used also have been broken. Instead of relying on encryption schemes that have not been broken yet, it is therefore desirable to have a mathematical proof that a given scheme *cannot* be broken. Prior to being able to perform such a proof, one needs to *define* what it means for a scheme to be secure, and without a meaningful definition, all proofs are meaningless.

A minimal requirement for an encryption scheme is that it should be infeasible for an attacker to recover the encrypted message from a ciphertext (without the secret key). While schemes not satisfying this notion are clearly too weak for most applications, even schemes that provide this guarantee can be very weak. For example, it is not excluded that an attacker can easily recover large parts of the message. Hence, a better requirement seems to be that it should be hard to obtain *any information* about the message from a ciphertext. Turning this intuition into a precise mathematical definition, however, has been a nontrivial task.

Shannon has formalized *perfect secrecy* as the message and the ciphertext being statistically independent [Sha49]. This means that without the key, a ciphertext provides no information about the message. Unfortunately, Shannon has also shown in the same paper that perfect secrecy can only be achieved if the key is at least as long as the message, which limits the applicability of such schemes. Goldwasser and Micali introduced the notion of *semantic security*, which can be seen as an analog to Shannon's perfect secrecy (for public-key encryption), where security is only guaranteed against computationally bounded adversaries [GM84]. They have also shown that semantic security is implied by what is known today as *ciphertext indistinguishability under chosen-plaintext attacks (IND-CPA security)*. Later, two variants of the even stronger notion of *ciphertext indistinguishability under chosen-ciphertext attacks (IND-CCA security)* were introduced [NY90; RS92]. Several additional notions have been proposed and relations among them have been proven [DDN00; BDPR98; CKN03].

The long history and large number of existing security definitions show that there is no universally "right" notion and that it is generally difficult to find a meaningful definition. Many of the aforementioned definitions are also quite technical and it is often hard to tell which definitions are sufficient to guarantee the desired security in a given application. For more advanced types of encryption, as the ones considered in this thesis, the situation only gets worse.

Security definitions and their semantics. Before explaining our approach to analyzing security definitions, we point out that these definitions can serve two entirely different purposes, which are often not clearly distinguished.

The first purpose of a security definition is to serve as a (technical) reference point, on the one hand for devising schemes provably satisfying the definition based on a weak assumption (e.g., a CCA-secure PKE scheme based on the DDH assumption [CS98]), and on the other hand for building more sophisticated primitives from any scheme satisfying the definition (e.g., constructing a CCA-secure PKE scheme from a CPA-secure IBE scheme and a one-time signature scheme [BCHK07]). Results about security definitions often take the form of a comparison, for example an equivalence or a separation statement, meaning that one definition is strictly stronger than another one.

The second purpose of a (technical) security definition is to assure the security of a certain type of application when a scheme satisfying the (technical) security definition is used. While definitions are usually devised with much intuition for what is needed in a certain application (and indeed the definition is often motivated by an application story), it is important to point out that a conventional technical security definition for a cryptographic primitive can generally not directly imply the security of an associated application, for two independent reasons. First, the particular *use* of the primitive within a protocol would have to be precisely specified. For example how is the message to be encrypted formed, to whom is it sent, over what kind of channel, and are certain fields like an IP address included in the MAC? Second, the application and its security requirements would also have to be formalized precisely. For example, game-based security definitions for key agreement do not explicitly guarantee that one can safely use the key in a given context, nor what the requirements are for the channels over which the protocol is executed (e.g., that they must be authenticated).

1.2 Constructive Cryptography

The general question of bridging the gap between a technical (e.g., game-based) security definition and the security of an intended application making use of the primitive is a foundational one. A goal of the constructive cryptography (CC) framework [Mau12; MR11] is to do exactly this: provide constructive semantics for technical security definitions and, based on the semantics, to compare definitions and identify the adequate one(s).

The approach taken is well-established in cryptography and very natural, and it can perhaps be seen as the only viable approach to tightly link security definitions and applications. Namely, one formalizes the application as an ideal-world system, called a resource in CC, which captures both what one wants and what one does not want to happen. For example, secure communication can be modeled as a secure channel where the adversary learns at most the length of the message. Other frameworks that capture security properties by defining an ideal functionality include (variants of) Universal Composability (UC) [Can01; HS15; KT13] and Reactive Simulatability (RSIM) [PW01; BPW07]. These frameworks have a different focus and are designed bottom-up from a specific machine model, while the constructive cryptography framework follows a top-down approach, leading to simpler descriptions and avoiding technicalities.

The use of a cryptographic scheme can be understood as constructing a certain (ideal) resource from certain assumed (real) resources. For example, symmetric encryption can then be understood as constructing a secure (length-leaking) channel from an authenticated channel and a secret key, and a key-agreement protocol can be understood as constructing a secret key (distributed to two parties A and B, where the adversary learns nothing) from a bidirectional authenticated channel. This constructive approach provides composition of constructions, i.e., the constructed resource (e.g., a secret key) can be used in any other construction as an assumed resource, and we have modularity since the overall security follows automatically from the individual security proofs and the composition theorem of constructive cryptography.

1.3 Studied Encryption Types and Results

1.3.1 One-Time Pad

We first analyze the one-time pad in Chapter 3. In contrast to the other types of encryption we consider in this thesis, the one-time pad has not been designed to have enhanced features except for its well known information-theoretic security. We focus on another, completely different aspect, which surfaces only when the receiver (not only the eavesdropper) is considered potentially dishonest, as can be the case in a larger protocol context in which encryption is used as a sub-protocol.

For example, such a dishonest receiver (who is, say, coerced by the eavesdropper) can in normal encryption *verifiably* leak the message to the eavesdropper by revealing the secret key. While this leakage feature can provably not be avoided completely, it is more limited if the one-time pad is used. This is due to the fact that the one-time pad, as *deniable encryption*, has the property that the receiver can after obtaining the ciphertext, present a key that decrypts this ciphertext to an arbitrary message of the receiver's choice. We use the constructive cryptography framework to make these statements precise.

The results in that chapter have been published in [MM13].

1.3.2 Identity-Based Encryption

In Chapter 4, we look at identity-based encryption (IBE). An IBE scheme includes an algorithm to generate a master public key and a master secret key, and an algorithm to generate user secret keys for a given identity using the master secret key. To encrypt a message, only the master public key and the identity of the recipient is needed, and the resulting ciphertext can be decrypted with the user secret key for that identity.

We formalize the standard application of identity-based encryption as constructing an ideal resource which we call delivery controlled channel (DCC). This resource allows users to be registered (by a central authority) for an identity and to send messages securely to other users only known by their identity.

Quite surprisingly, we show that existing security definitions for IBE are not sufficient to construct DCC. In fact, it is impossible to do so in the standard model. We show, however, how to adjust any IBE scheme that satisfies the standard security definition IND-ID-CPA to achieve this goal in the random oracle model.

We also show that the impossibility result can be avoided in the standard model by considering a weaker ideal resource that requires all users to be registered in an initial phase before any messages are sent. To achieve this, a weaker security notion, which we introduce and call IND-ID1-CPA, is actually sufficient. This justifies our new security definition and might open the door for more efficient schemes. We further investigate which ideal resources can be constructed with schemes satisfying the standard notion and variants of selective security.

The results in that chapter have been published in [HMM15].

1.3.3 Functional Encryption

In Chapter 5, we consider Functional encryption (FE), which is a powerful generalization of various types of encryption. Like IBE, it allows to generate a master public key and a master secret key. The master secret key for FE schemes, however, can be used to generate decryption keys for a given function. When data x is encrypted using the master public key and the resulting ciphertext is decrypted using a decryption key for the function f, one learns $f(x)$.

We investigate how functional encryption can be used by a trusted authority to enforce access-control policies to data stored in an untrusted repository. Intuitively, if (functionally) encrypted data items are put in a publicly-readable repository, the effect of the encryption should be that users have access to exactly (and only) those functions of the data items for which they have previously received the corresponding decryption key. That is, in an ideal-world view, the key authority can flexibly manage read access of users to the repository. This appears to be exactly what functional encryption is supposed to achieve, and most natural applications of thereof can be understood as specific uses of such a repository with access control.

We formalize the described ideal-world interpretation as a resource and propose a new conventional security definition, called *composable functional encryption security* (CFE security), which we prove to be equivalent to the construction of this resource. This definition (and hence the described application) is shown to be unachievable in the standard model but achievable in the random oracle model. Moreover, we show that somewhat weaker definitions, which are achievable in the standard model, can be obtained by certain operational restrictions of the ideal-world repository, making explicit how schemes satisfying such a definition can (and cannot) meaningfully be used.

The results in that chapter have been published in [MM15].

1.3.4 Access Control Encryption

Finally, we consider access control encryption (ACE) in Chapter 6. This recently introduced type of encryption enables to control the 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 not only against dishonest users, but also against the sanitizer.

We show that in order to be resilient against realistic attacks, existing security definitions 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, not even by a dishonest sender.

At the end of the chapter, we sketch the natural ideal resource and the construction thereof one would expect ACE to achieve. We then explain why ACE actually only provides very weak guarantees in constructive cryptography and conclude by hinting at ways to resolve this.

Except for the discussion of the modeling in constructive cryptography, the results in that chapter have been published in [BMM17].

1.4 Related Work

Several other types of encryption have been considered in the constructive cryptography framework. These include symmetric encryption [MRT12], (anonymous) public-key encryption [CMT13; KMO+13], and (robust) authenticated encryption [BMM+15a; BMM+15b].

What encryption achieves in an ideal-world sense has also been investigated in the UC framework for public-key encryption [Can01; CKN03] and for symmetric encryption [KT09], and in the RSIM framework for public-key encryption [PW01].

Since those papers consider different types of encryption schemes and have different goals, the results are significantly different from the ones obtained in this thesis. Work related to the types of encryption we consider is referenced in the corresponding chapters.

Chapter 2

Preliminaries

2.1 General Notation

The algorithms and systems in this thesis are described by pseudocode using the following conventions: 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. If $\mathcal{A}$ is an algorithm, $y \leftarrow \mathcal{A}(x)$ denotes executing $\mathcal{A}$ on input x and assigning the returned value to y. For a probabilistic algorithm $\mathcal{A}$ and a (sufficiently long) bit string $r \in \{0,1\}^*$, $\mathcal{A}(x;r)$ denotes the execution of $\mathcal{A}$ on input x with randomness r, and $\mathcal{A}(x)$ denotes the execution with uniformly chosen randomness. 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 $\mathcal{O}$.

We denote the set of natural numbers by $\mathbb{N} = \{0, 1, 2, \ldots\}$, and the set of integers by $\mathbb{Z}$. For $n \in \mathbb{N}$, we use the convention

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

By $\mathbb{Z}_n$ we denote the ring of integers modulo n, and by $\mathbb{Z}_n^*$ its multiplicative group of units. We denote the length of a bit string s by $|s|$ and for $s_1, \ldots, s_n$, $|(s_1, \ldots, s_n)|$ denotes the bit length of (some fixed) unique encoding of $(s_1, \ldots, s_n)$.

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 the event A is denoted by $\neg A$.

2.2 Security Definitions and Advantages

The security of a cryptographic scheme is usually defined via a random experiment (or game) involving an adversary algorithm $\mathcal{A}$. For a given scheme $\mathcal{E}$ and adversary $\mathcal{A}$, one defines 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 $\mathcal{E}$. Such a scheme is considered *secure* if $\mathsf{Adv}^{\mathsf{Sig\text{-}EUF\text{-}CMA}}_{\mathcal{E},\mathcal{A}}$ is *negligible* (in κ) for all *efficient* $\mathcal{A}$. An algorithm $\mathcal{A}$ is *efficient* if it runs in *probabilistic polynomial time (PPT)*, i.e., $\mathcal{A}$ has access to random bits and there is a polynomial p such that $\mathcal{A}(x)$ terminates after at most $p(|x|)$ steps (in some computational model, e.g., Turing machines) for all inputs x. A function f is *negligible* if for every polynomial p, there exists $n_0 \in \mathbb{N}$ such that $f(n) < 1/p(n)$ for all $n \geq n_0$. While these asymptotic definitions yield concise statements, we will mostly derive precise bounds on the advantages, following a concrete security approach.

2.3 Cryptographic Primitives and Games

2.3.1 Decisional Diffie-Hellman Assumption

Definition 2.3.1. Let $G = \langle g \rangle$ be a cyclic group of prime-order q and let g be a generator. Let $\mathcal{A}$ be a probabilistic algorithm that on input q, g, and three elements $X, Y, T \in G$ returns a bit d. Let $\mathsf{DDH}^{\mathrm{real}}_{g,\mathcal{A}}$ 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 $\mathsf{DDH}^{\mathrm{rand}}_{g,\mathcal{A}}$ 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

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

The *decisional Diffie-Hellman (DDH) assumption* for the group G states that $\mathsf{Adv}^{\mathsf{DDH}}_{g,\mathcal{A}}$ is negligible for all efficient $\mathcal{A}$.

2.3.2 Pseudorandom Functions

Definition 2.3.2. 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

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

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. The function family F is called *pseudorandom* if $\mathsf{Adv}^{\mathsf{PRF}}_{F,\mathcal{A}}$ is negligible for all efficient $\mathcal{A}$.

2.3.3 Public-Key Encryption

Definition 2.3.3. 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 $m \in \mathcal{M} \cup \{\bot\}$.

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

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

with probability 1.

Definition 2.3.4. 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 algorithms. Consider the experiment $\mathsf{Exp}^{\mathsf{PKE\text{-}IND\text{-}CPA}}_{\mathcal{E},\mathcal{A}}$ in Figure 2.1. We define the *ciphertext indistinguishability under chosen-plaintext attacks advantage* of $\mathcal{A}$ as

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

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

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

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

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

Figure 2.1: Experiments for the security definitions of public-key encryption and digital signature schemes.

where the probability is over the randomness in $\mathsf{Exp}_{\mathcal{E},\mathcal{A}}^{\mathsf{PKE\text{-}IND\text{-}CPA}}$. The scheme $\mathcal{E}$ has *indistinguishable ciphertexts under chosen-plaintext attacks (is IND-CPA secure)* if $\mathsf{Adv}_{\mathcal{E},\mathcal{A}}^{\mathsf{PKE\text{-}IND\text{-}CPA}}$ is negligible for all efficient $\mathcal{A}$.

2.3.4 Digital Signature Schemes

Definition 2.3.5. 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 2.3.6. Let $\mathcal{E} = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Ver})$ be a signature scheme and let $\mathcal{A}$ be a probabilistic algorithm. Consider the experiment $\mathsf{Exp}_{\mathcal{E},\mathcal{A}}^{\mathsf{Sig\text{-}EUF\text{-}CMA}}$

in Figure 2.1 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

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

where the probability is over the randomness in $\mathsf{Exp}^{\mathsf{Sig\text{-}EUF\text{-}CMA}}_{\mathcal{E},\mathcal{A}}$. The scheme $\mathcal{E}$ is *existentially unforgeable under adaptive chosen-message attacks (EUF-CMA secure)* if $\mathsf{Adv}^{\mathsf{Sig\text{-}EUF\text{-}CMA}}_{\mathcal{E},\mathcal{A}}$ is negligible for all efficient $\mathcal{A}$.

2.3.5 Non-Interactive Zero-Knowledge Proofs

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

Definition 2.3.7. 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 (or for R) 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

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

with probability 1.

Definition 2.3.8 (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

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

The scheme $\mathcal{E}$ is *computationally sound* if $\mathsf{Adv}^{\mathsf{NIZK\text{-}snd}}_{\mathcal{E},\mathcal{A}}$ is negligible for all efficient $\mathcal{A}$ and *perfectly sound* if $\mathsf{Adv}^{\mathsf{NIZK\text{-}snd}}_{\mathcal{E},\mathcal{A}} = 0$ for all $\mathcal{A}$.

Definition 2.3.9 (Computational zero-knowledge)**.** Let $\mathcal{E} = (\mathsf{Gen}, \mathsf{Prove}, \mathsf{Ver})$ be a non-interactive proof system for a relation R and let $S = (S_1, S_2)$ be a pair of PPT algorithms, called *simulator*. Further 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

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

We call $(\mathsf{Gen}, \mathsf{Prove}, \mathsf{Ver}, S_1, S_2)$ a *non-interactive zero-knowledge (NIZK) proof system* for R if $\mathsf{Adv}^{\mathsf{NIZK\text{-}ZK}}_{\mathcal{E},S,\mathcal{A}}$ is negligible for all efficient $\mathcal{A}$; it is called *single-theorem NIZK proof system* if $\mathsf{Adv}^{\mathsf{NIZK\text{-}ZK}}_{\mathcal{E},S,\mathcal{A}}$ is negligible for all efficient $\mathcal{A}$ that make at most one query to their oracle.

Definition 2.3.10 (Knowledge extraction)**.** Let $\mathcal{E} = (\mathsf{Gen}, \mathsf{Prove}, \mathsf{Ver})$ be a non-interactive proof system for a relation R and let $E = (E_1, E_2)$ be a pair of PPT algorithms, called *knowledge extractor*. We define the *knowledge extraction advantages* of a probabilistic algorithm $\mathcal{A}$ as

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

We call $(\mathsf{Gen}, \mathsf{Prove}, \mathsf{Ver}, E_1, E_2)$ a *non-interactive proof of knowledge system* for R if $\mathsf{Adv}^{\mathsf{NIZK\text{-}ext_1}}_{\mathcal{E},E,\mathcal{A}}$ and $\mathsf{Adv}^{\mathsf{NIZK\text{-}ext_2}}_{\mathcal{E},E,\mathcal{A}}$ are negligible for all efficient $\mathcal{A}$.

Definition 2.3.11 (Simulation soundness)**.** Let $\mathcal{E} = (\mathsf{Gen}, \mathsf{Prove}, \mathsf{Ver})$ be a non-interactive proof system for a language L, let $S = (S_1, S_2)$ be a pair of PPT algorithms, and let $\mathcal{A}$ be a probabilistic algorithm. Consider the experiment $\mathsf{Exp}^{\mathsf{NIZK\text{-}sim\text{-}snd}}_{\mathcal{E},S,\mathcal{A}}$ 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 the 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

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

We say $(\mathsf{Gen}, \mathsf{Prove}, \mathsf{Ver}, S_1, S_2)$ is *simulation sound* if $\mathsf{Adv}^{\mathsf{NIZK\text{-}sim\text{-}snd}}_{\mathcal{E},S,\mathcal{A}}$ is negligible for all efficient $\mathcal{A}$, and it is *one-time simulation sound* if $\mathsf{Adv}^{\mathsf{NIZK\text{-}sim\text{-}snd}}_{\mathcal{E},S,\mathcal{A}}$ is negligible for all efficient $\mathcal{A}$ that make at most one query to the oracle S_2.

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.

2.4 Constructive Cryptography

The results in this thesis are formulated using a simulation-based notion of security. There are many protocol frameworks based on such a simulation-based security notion (e.g., [GMW87; Bea92; MR92; Can01; PW01; BPW07; MR11; Mau12]). However, in this work, we use the constructive cryptography (CC) framework [Mau12; MR11].

Briefly, CC makes statements about *constructions* of *resources* from other resources. A resource is a system with interfaces via which the resource interacts with its environment and which can be thought of as being assigned to parties. *Converters* are systems that can be attached to an interface of a resource to change the inputs and outputs at that interface, which yields another resource. The protocols of honest parties and simulators correspond to converters. Dishonest behavior at an interface is captured by *not* applying the protocol (instead of modeling an explicit adversary). An ideal resource is *constructed* from a real resource by a protocol, if the real resource with the protocol converters attached at the honest interfaces is indistinguishable from the ideal resource with the simulators attached at the dishonest interfaces.

We introduce the relevant concepts in more detail, following [MR11], in the following subsections. For readers more familiar with the Universal Composability (UC) framework [Can01], we also include explanations of how the presented concepts relate to similar concepts in UC.

2.4.1 Resources, Converters, and Distinguishers

We consider different types of *systems*, which are objects with *interfaces* via which they interact with their environment. Interfaces are denoted by uppercase letters. One can compose two systems by connecting one interface of each system. The composed object is again a system.

Two types of systems we consider here are *resources* and *converters*. Resources are denoted by (partially) capitalized sans serif fonts, e.g., R or $\mathsf{AutC}^{A,B}$, and have a finite set $\mathcal{I}$ of interfaces. Resources with interface set $\mathcal{I}$ are called *$\mathcal{I}$-resources*. Converters have one *inside* and one *outside interface* and are denoted by lowercase Greek letters or sans serif fonts. In figures, resources are drawn with rectangular boxes and converters with boxes with rounded corners. The inside interface of a converter α can be connected to interface $I \in \mathcal{I}$ of a resource R. The outside interface of α then serves as the new interface I of the composed resource, which is denoted by $\alpha^I\mathsf{R}$. We also write $\alpha_I\mathsf{R}$ instead of $\alpha_I^I\mathsf{R}$ for a converter α_I. For a tuple of converters $\alpha = (\alpha_{I_1}, \dots, \alpha_{I_n})$ with $I_1, \dots, I_n \in \mathcal{I}$ and a set $\mathcal{P} \subseteq \{I_1, \dots, I_n\}$ of interfaces, $\alpha_{\mathcal{P}}\mathsf{R}$ denotes the $\mathcal{I}$-resource that results from connecting α_I to interface I of R for every $I \in \mathcal{P}$. Moreover, $\alpha_{\overline{\mathcal{P}}}\mathsf{R}$ denotes the $\mathcal{I}$-resource one gets when α_I is connected to interface I of R for every $I \in \{I_1, \dots, I_n\} \setminus \mathcal{P}$. For $\mathcal{I}$-resources $\mathsf{R}_1, \dots, \mathsf{R}_m$, the *parallel composition* $[\mathsf{R}_1, \dots, \mathsf{R}_m]$ is defined as the $\mathcal{I}$-resource where each interface $I \in \mathcal{I}$ allows to access the corresponding interfaces of all sub-systems R_i as sub-interfaces. Similarly, for converters $\alpha_1, \dots, \alpha_m$, we define the *parallel composition* $[\alpha_1, \dots, \alpha_m]$ via $[\alpha_1, \dots, \alpha_m]^I[\mathsf{R}_1, \dots, \mathsf{R}_m] := [\alpha_1^I\mathsf{R}_1, \dots, \alpha_m^I\mathsf{R}_m]$.

A *distinguisher* $\mathcal{D}$ for resources with n interfaces is a system with $n+1$ interfaces, where n of them connect to the interfaces of a resource and a bit is output at the remaining one. We write $\Pr[\mathcal{D}\mathsf{R} = 1]$ to denote the probability that $\mathcal{D}$ outputs the bit 1 when connected to resource R. The goal of a distinguisher is to distinguish two resources by outputting a different bit when connected to a different resource. Its success is measured by the distinguishing advantage.

Definition 2.4.1. The *distinguishing advantage* of a distinguisher $\mathcal{D}$ for resources R and S is defined as

$$\Delta^{\mathcal{D}}(\mathsf{R}, \mathsf{S}) := \Pr[\mathcal{D}\mathsf{R} = 1] - \Pr[\mathcal{D}\mathsf{S} = 1].$$

If $\Delta^{\mathcal{D}}(\mathsf{R}, \mathsf{S}) = 0$ for all distinguishers $\mathcal{D}$, we say R and S are *equivalent*, denoted as $\mathsf{R} \equiv \mathsf{S}$. If the distinguishing advantage is negligible for all efficient distinguishers, we say R and S are *computationally indistinguishable*, denoted as $\mathsf{R} \approx \mathsf{S}$.

Remark. One can also consider *statistical indistinguishability* by allowing a negligible distinguishing advantage for all (including inefficient) distinguishers, but we do not need that concept in this thesis.

We introduce two special converters $\mathbf{1}$ and $\perp$. The converter $\mathbf{1}$ forwards all inputs at one of its interfaces to the other one. We thus have for all $\mathcal{I}$-resources R and all $I \in \mathcal{I}$

$$\mathbf{1}^I \mathsf{R} \equiv \mathsf{R}.$$

One can equivalently understand connecting $\mathbf{1}$ to interface I of a resource as not connecting any converter to that interface. Moreover, the converter $\perp$ blocks all inputs at the connected interface. That is, interface I of $\perp^I \mathsf{R}$ does not accept any inputs and there are no outputs at this interface.

Defining systems. We define systems by specifying their input/output behavior either using pseudocode or with a textual description. Such a specification describes how the system reacts to inputs at its interfaces. We use the convention that inputs not matching any of the described cases are ignored. Systems can have an optional initialization phase, which is executed before any inputs are processed.

Efficiency of systems. For defining computational security, one needs a notion of an *efficient* system. The class of these systems can be defined by fixing a model of computation and considering all systems satisfying certain conditions on the number of computational steps and the lengths of the inputs and outputs. The important property needed from such a notion is that connecting efficient systems again yields an efficient system [MR11]. Precisely defining this in a meaningful way is rather involved and the details are not important to understand the results in this thesis.

Relation to UC concepts. In UC, systems as above can correspond to protocols, ideal functionalities, or simulators that interact with the protocol environment. More specifically, resources correspond to ideal functionalities, while converters can correspond to real or hybrid protocols, or to simulators. Namely, a UC protocol can be viewed as a way to convert calls to that protocol to calls to an underlying communication infrastructure (or hybrid functionality). Conversely, a UC simulator can be viewed as a way to convert the network interface of one protocol into that of another one. (In CC, there is no a-priori distinction between I/O and network interfaces; hence, both UC protocols and UC simulators correspond to converters.) Distinguishers as above correspond to the UC protocol environments. In contrast to CC, the systems in UC are fixed to a very specific type of Turing machines. To be compatible with that, the class of efficient systems in CC can be instantiated by such Turing machines.

2.4.2 Filtered Resources

In some situations, specific interactions with a resource might not be guaranteed but only potentially available. To model such situations, we extend the concept of a resource. Let R be an $\mathcal{I}$-resource and let $\phi = (\phi_I)_{I \in \mathcal{I}}$ be a tuple of converters. We define the *filtered resource* R_ϕ as a resource with the same set of interfaces $\mathcal{I}$. For a party connected to interface I of R_ϕ, interactions through the converter ϕ_I are guaranteed to be available, while interactions with R directly are only potentially available to dishonest parties. The converter ϕ_I can be seen as a filter shielding specific functionality of interface I. Dishonest parties can potentially remove the filter to get access to all features of the resource R. Formally, R_ϕ is defined as the set of all resources that allows all interactions allowed by $\phi_{\mathcal{I}}\mathsf{R}$ but not more than allowed by R; see [MR11] for more details.

2.4.3 Basic Resources

An important cryptographic resource is a shared secret key between A and B, which outputs the same random value at the interfaces A and B and nothing at the interface of an eavesdropper.

Definition 2.4.2. A *shared secret key*, denoted as $\mathsf{sKey}^{A,B}$, is a resource with three interfaces A, B, and E. It outputs a uniformly random value at the interfaces A and B and does not output anything at interface E. All inputs are ignored.

Another important example of resources are communication channels, which allow the sender A to send messages from some message space $\mathcal{M} \subseteq \{0,1\}^*$ to the receiver B. We define two such channels, which differ in the capabilities of the adversary E.

Definition 2.4.3. An *authenticated channel from A to B for $n \in \mathbb{N}$ messages and message space $\mathcal{M}$*, denoted as $\mathsf{AutC}^{n,A,B}$, and a *secure channel from A to B for $n \in \mathbb{N}$ messages and message space $\mathcal{M}$*, denoted as $\mathsf{SecC}^{n,A,B}$, are resources with three interfaces A, B, and E.[1] On input a message $m \in \mathcal{M}$ at interface A, they both output the same message m at interface B if less than n messages have been sent before. Additionally, $\mathsf{AutC}^{n,A,B}$ outputs m at interface E and $\mathsf{SecC}^{n,A,B}$ outputs the length $|m|$ of the message at interface E. Other inputs as well as attempts to send more than n messages are ignored. Authenticated and secure channels that allow arbitrarily many messages to be sent are denoted by $\mathsf{AutC}^{A,B}$ and $\mathsf{SecC}^{A,B}$, respectively.

Remark. Alternatively, one could define authenticated and secure channels such that E also has the ability to delete messages. Most results in this thesis can be adapted to such a setting, but our assumption that sent messages are always delivered allows to simplify the presentation.

For authenticated channels, we do not want to guarantee that an adversary learns the message, it is rather not excluded. Similarly, secure channels should not guarantee that the length of the message leaks. To model this, we introduce filters that block all outputs at interface E. We then have that a secure channel is also authenticated, i.e., the set of (filtered) secure channels is a subset of the set of (filtered) authenticated channels.

Definition 2.4.4. Let $\phi^{\mathsf{AutC}} = \phi^{\mathsf{SecC}} := (\mathbf{1}, \mathbf{1}, \bot)$. We will consider the filtered resources $\mathsf{AutC}^{A,B}_{\phi^{\mathsf{AutC}}}$ and $\mathsf{SecC}^{A,B}_{\phi^{\mathsf{SecC}}}$.

Note that

$$\begin{aligned}\phi^{\mathsf{AutC}}_{\{A,B,E\}}\mathsf{AutC}^{A,B} &= \mathbf{1}^A\mathbf{1}^B\bot^E\mathsf{AutC}^{A,B} \\ &\equiv \mathbf{1}^A\mathbf{1}^B\bot^E\mathsf{SecC}^{A,B} = \phi^{\mathsf{SecC}}_{\{A,B,E\}}\mathsf{SecC}^{A,B}\end{aligned}$$

accepts messages at interface A and outputs them at interface B where interface E is inactive.

We finally introduce a more advanced communication resource that has many interfaces and allows a sender to send messages to all other

[1]Since the message space is typically either irrelevant or clear from the context, we do not include it in the notation.

interfaces. It is authenticated in the sense that the messages cannot be modified and everyone receives the same message.

Definition 2.4.5. The *broadcast* resource $\mathsf{BCast}^{A,\mathcal{B}}$ for a set $\mathcal{B}$ has interface set $\{A\} \cup \mathcal{B}$. On input a message $m \in \mathcal{M}$ at interface A, the same message is output at all interfaces $B \in \mathcal{B}$. Other inputs are ignored.

Relation to UC concepts. The presented resources directly correspond to UC ideal functionalities for authenticated, secure, or broadcast channels. The different interfaces of the presented resources correspond to what different parties in UC could send or receive. (Here we note a common design difference in UC and CC: in UC, typically one would assume parties as fixed entities, and model communication and interfaces around them. In CC, one would typically start with the interfaces that reflect the semantic types of in- and outputs of a resource, and only later think of connecting entities like parties.)

2.4.4 Construction of Resources

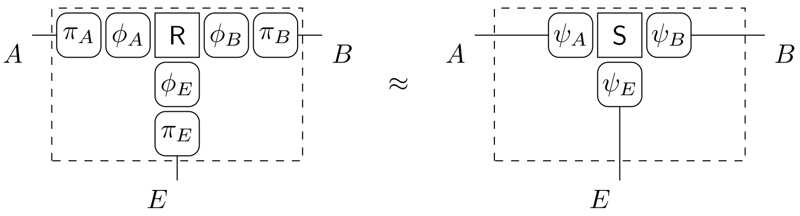
A *protocol* is a tuple of converters with the purpose of constructing a so-called ideal resource from an available real resource. Depending on which parties are considered potentially dishonest, we get a different notion of construction.

As an example from [CMT13], consider the setting for public-key encryption with honest A and B where we want to construct a secure channel $\mathsf{SecC}^{A,B}_{\phi^{\mathsf{SecC}}}$ from authenticated channels $\mathsf{AutC}^{1,B,A}_{\phi^{\mathsf{AutC}}}$ and $\mathsf{AutC}^{A,B}_{\phi^{\mathsf{AutC}}}$ in presence of a dishonest eavesdropper E. Here, the real resource is $\mathsf{R} := \left[\mathsf{AutC}^{1,B,A}_{\phi^{\mathsf{AutC}}}, \mathsf{AutC}^{A,B}_{\phi^{\mathsf{AutC}}}\right]$ and the ideal resource is $\mathsf{S} := \mathsf{SecC}^{A,B}_{\phi^{\mathsf{SecC}}}$. In this setting, a protocol $\pi = (\pi_A, \pi_B, \pi_E)$ constructs S from R with potentially dishonest E if there exists a converter σ_E (called *simulator*) such that

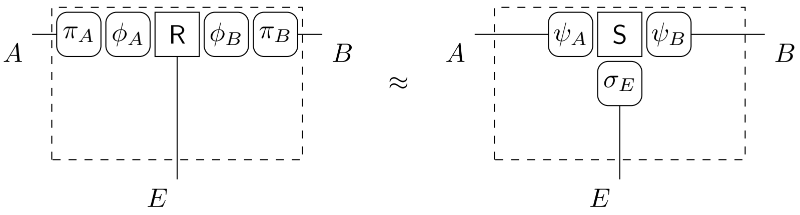
$$\pi_A \pi_B \pi_E \left[\phi_A^{\mathsf{AutC}} \phi_B^{\mathsf{AutC}} \phi_E^{\mathsf{AutC}} \mathsf{AutC}^{1,B,A}, \phi_A^{\mathsf{AutC}} \phi_B^{\mathsf{AutC}} \phi_E^{\mathsf{AutC}} \mathsf{AutC}^{A,B}\right] \\ \approx \phi_A^{\mathsf{SecC}} \phi_B^{\mathsf{SecC}} \phi_E^{\mathsf{SecC}} \mathsf{SecC}^{A,B}$$

and

$$\pi_A \pi_B \left[\phi_A^{\mathsf{AutC}} \phi_B^{\mathsf{AutC}} \mathsf{AutC}^{1,B,A}, \phi_A^{\mathsf{AutC}} \phi_B^{\mathsf{AutC}} \mathsf{AutC}^{A,B}\right] \\ \approx \phi_A^{\mathsf{AutC}} \phi_B^{\mathsf{AutC}} \sigma_E \mathsf{SecC}^{A,B},$$



(a) Correctness condition.



(b) Security condition.

Figure 2.2: Depiction of the two conditions for the construction of a filtered $\{A, B, E\}$-resource S_ψ from the filtered $\{A, B, E\}$-resource R_ϕ with potentially dishonest E by the protocol π.

where σ_E provides a sub-interface to the distinguisher for each channel that constitutes the real resource. See Figure 2.2 for a graphical representation of these two conditions. The first condition ensures that the protocol implements the required functionality and the second condition ensures that whatever Eve can do when connected to the real resource without necessarily following the protocol, she could do as well when connected to the ideal resource by using the simulator σ_E. Since Eve is here only a hypothetical entity, we typically have $\pi_E = \perp$.

In this thesis, we consider the more general setting that includes several potentially dishonest parties that (in contrast to Eve in the above example) also get certain guarantees if they are honest while unable to do more than specified by the ideal resource even if they are dishonest. We define a secure construction as follows.

Definition 2.4.6. Let R_ϕ and S_ψ be filtered $\mathcal{I}$-resources and let $\pi = (\pi_I)_{I \in \mathcal{I}}$ be a protocol. Further let $\mathcal{U} \subseteq \mathcal{I}$ be the set of interfaces with potentially dishonest behavior. We say π *constructs* S_ψ from R_ϕ *with potentially dishonest* $\mathcal{U}$, denoted by

$$\mathsf{R}_\phi \xmapsto[\mathcal{U}]{\pi} \mathsf{S}_\psi,$$

if there exist converters $\sigma = (\sigma_U)_{U \in \mathcal{U}}$ such that

$$\forall \mathcal{P} \subseteq \mathcal{U} : \pi_{\overline{\mathcal{P}}} \phi_{\overline{\mathcal{P}}} \mathsf{R} \approx \sigma_{\mathcal{P}} \psi_{\overline{\mathcal{P}}} \mathsf{S}.$$

For singleton sets $\mathcal{U} = \{E\}$, we here write E instead of $\{E\}$ to simplify the notation. The converters σ_U are called *simulators.*[2]

For $\mathcal{U} = \mathcal{I}$, this definition corresponds to the abstraction notion from [MR11], which considers all parties as potentially dishonest. To apply the above definition to an unfiltered resource R, one can formally introduce trivial filters $\phi_I := \mathbf{1}$ for $I \in \mathcal{I}$ and consider the filtered resource R_ϕ which is identical to R. In such cases, we will omit the filters.

Composition of constructions. The notion of construction is composable in the following sense: We have for all filtered resources R_ϕ, $\mathsf{R}'_{\phi'}$, S_ψ, $\mathsf{S}'_{\psi'}$, and T_τ, and for all protocols $\pi = (\pi_A, \pi_B, \pi_C)$ and $\pi' = (\pi'_A, \pi'_B, \pi'_C)$,

$$\mathsf{R}_\phi \xmapsto[\mathcal{U}]{\pi} \mathsf{S}_\psi \wedge \mathsf{S}_\psi \xmapsto[\mathcal{U}]{\pi'} \mathsf{T}_\tau \quad \Longrightarrow \quad \mathsf{R}_\phi \xmapsto[\mathcal{U}]{\pi' \circ \pi} \mathsf{T}_\tau,$$

$$\mathsf{R}_\phi \xmapsto[\mathcal{U}]{\pi} \mathsf{S}_\psi \wedge \mathsf{R}'_{\phi'} \xmapsto[\mathcal{U}]{\pi'} \mathsf{S}'_{\psi'} \quad \Longrightarrow \quad [\mathsf{R}_\phi, \mathsf{R}'_{\phi'}] \xmapsto[\mathcal{U}]{[\pi, \pi']} [\mathsf{S}_\psi, \mathsf{S}'_{\psi'}],$$

where $\pi' \circ \pi := (\pi'_A \circ \pi_A, \pi'_B \circ \pi_B, \pi'_C \circ \pi_C)$ and $[\pi, \pi'] := \big([\pi_A, \pi'_A], [\pi_B, \pi'_B], [\pi_C, \pi'_C]\big)$. The first property guarantees that the composition of construction steps yields a secure overall construction and the second property ensures that a construction remains secure in any context, i.e., regardless of what happens in parallel. See [Mau12; MR11] for a proof of these statements and further discussions.[3]

[2]Note that we require the existence of one simulator for all efficient distinguishers. This simulator does therefore not depend on the distinguisher and hence is *black-box.*

[3]In fact, [Mau12] only proves the special case for $\{A, B, E\}$-resources with potentially dishonest E, and [MR11] only considers the case $\mathcal{U} = \mathcal{I}$. The more general statement we need, however, follows analogously since the implications can be shown for each $\mathcal{P} \subseteq \mathcal{U}$ separately.

Relation to UC concepts. The “constructs” notion presented above directly corresponds to the UC notion of secure realization. (The combination of π and R corresponds to the real protocol in UC, while S matches the UC ideal protocol.) The “constructs” notion does not consider an explicit adversary on the real protocol. (Instead, in UC terms, a dummy adversary is considered without loss of generality.) There is a difference, however, in the modeling of corruptions. Generally, in UC, adaptive corruptions are considered. In the CC modeling above, only static corruptions of parties are considered. Moreover, instead of modeling corruptions through special “corrupt” messages sent from the adversary or environment, in CC corruptions are modeled simply be letting the distinguisher connect to the interfaces of corrupted parties.

Finally, a subtle difference between CC and UC security is that CC security requires “local” simulators for each interface, whereas in UC, one simulator is required that handles all parties (resp. interfaces) at once. While this makes CC security a stricter notion than UC security, this difference is not important for our results; in particular, our impossibility results have nothing to do with the fact that CC security requires local simulation.

Chapter 3

Deniability of the One-Time Pad

3.1 Introduction

3.1.1 Motivation

The purpose of symmetric encryption in constructive cryptography can be seen as constructing a secure channel from a sender Alice to a receiver Bob from a shared secret key and an authenticated channel. Given a key as long as the message, Alice can encrypt her message by bitwise XORing the key to the message, which yields the corresponding ciphertext. She then sends this ciphertext over the channel to Bob, who can recover the message by bitwise XORing the key to the ciphertext. If the key is uniformly random and used only once, this cryptosystem is called *one-time pad*. It was shown in [Sha49] that an eavesdropper Eve does not learn anything about the message given only the ciphertext. This means constructively that the one-time pad can be used to construct a secure channel from a shared secret key and an authenticated channel if Eve is the only dishonest party, regardless of her computational power.

This analysis assumes that Alice and Bob are always honest, which is a standard assumption when analyzing symmetric encryption schemes. However, when channels are used in a more complex system with several parties, this assumption does not always hold. We analyze the one-time pad when the receiver Bob is potentially dishonest. This allows us to understand situations in which Bob is coerced to give the secret key to another party or in which Bob wants to betray Alice by convincing a third party that she has sent a specific message. If Bob sends Eve the key and Alice sends a message to Bob, Eve can learn the message. Of course, Bob can just send Eve the message, but if she does not trust him, there is no reason for her to believe that he sent her the correct message. However, receiving a key that later decrypts the ciphertext to a meaningful message is more convincing because it might be much harder or even impossible for Bob to find such a key. Hence, the resource generally constructed by encryption schemes cannot exclude that Bob convincingly leaks the message to Eve.

It is known that the one-time pad shares a feature with so-called *deniable encryption* schemes introduced in [CDNO97], which allows one to find a key for each pair of message and ciphertext such that the ciphertext decrypts to the given message with this key. Hence, once Bob knows the message, he can create a fake key that yields an arbitrary message of his choice. Intuitively, this means that receiving a key from a dishonest Bob after a message was sent is meaningless. We show that this intuition can be formalized in the constructive cryptography framework as follows: The one-time pad can be used to construct a resource which potentially allows Bob at the beginning to decide whether he wants to leak the message to Eve or not. However, he has to make this decision before Alice sends the message, i.e., his choice to leak the message or not cannot depend on the message. In contrast, when using ordinary encryption, one cannot exclude that Bob is still able to verifiably leak the message after it was sent. Hence, in addition to perfect secrecy, the one-time pad provides stronger guarantees than other encryption schemes.

3.1.2 Contributions

The main contribution of this chapter is the description of an ideal resource that can be constructed using the one-time pad in a setting with a potentially dishonest receiver. Furthermore, we show that it is impossible to construct a fully secure channel from a shared secret key and an authenticated channel if the receiver is dishonest.

These have been the first results of this form involving more than one potentially dishonest party, which is of independent interest as a new type of example in the constructive cryptography framework.

3.1.3 Related Work

The security of the one-time pad has been analyzed in constructive cryptography by Maurer [Mau12], and in the reactive simulatability framework by Raub, Steinwandt, and Müller-Quade [RSM05]. These works, however, only consider the standard setting with honest sender and receiver.

Deniability and incoercibility have been considered in (variants of) the UC framework [UM10; CV12; AOZZ15]. In comparison to these works, our approach is much simpler since we neither need new definitions nor to modify the constructive cryptography framework.

3.2 Encryption with a Dishonest Receiver

3.2.1 General Limitations

In this section, we examine encryption schemes in general. All results hold with respect to information-theoretic security as well as computational security. We investigate which resources one can construct from a shared secret key and an authenticated channel using encryption in the setting in which not only Eve but also Bob could be dishonest. If this is used as part of a larger protocol which allows Bob to exchange messages with Eve, a dishonest Bob could send her the key, resulting in Eve learning the message. Since the security guarantees are preserved under composition in our framework, the ideal resource constructed by an encryption scheme has to reflect this. That is, the ideal resource potentially gives Bob the option to leak the message to Eve. However, this option is not guaranteed. One reason for this is that Bob cannot leak the message in a more restricted setting where he does not have a communication channel to Eve. Therefore, we have a filtered resource where Bob's leakage button is shielded by a filter.

Something else has to be taken into account. Consider for example the one-time pad. The ciphertext there is a uniformly random bit string. Hence, when Alice sends an encrypted message, Bob and Eve get common randomness. For other types of encryption, they could potentially extract the randomness of the key from the ciphertext and thereby also get common randomness. Even if this is not considered to be an issue, it has to be reflected in the ideal resource. Altogether, the resource a typical encryption scheme constructs from a shared secret key and an authenticated channel is a secure channel which potentially gives Bob the option to leak the message to Eve and which potentially gives common randomness to Bob and Eve.

As we have just argued, ordinary encryption does not construct a secure channel without additional features from an authenticated channel and a shared secret key if Bob and Eve are potentially dishonest. We now show that this construction is generally impossible, not by any type of encryption scheme nor a completely different protocol (a variant of this was already stated in the appendix of [MR11] without proof). We assume here and for the rest of this chapter that the message space of all channels is $\mathcal{M} := \{0,1\}^\ell$ for some fixed $\ell \in \mathbb{N}$, and we only consider channels for sending a single message.

Theorem 3.2.1. *There exists no protocol $\pi = (\pi_A, \pi_B, \pi_E)$ that constructs a secure channel $\mathsf{SecC}^{1,A,B}$ from a shared secret key $\mathsf{sKey}^{A,B}$ and an authenticated channel $\mathsf{AutC}^{1,A,B}$ with potentially dishonest B and E.*

Proof. Assume such a protocol π exists. Then, there exist simulators σ_B and σ_E such that the following conditions hold:

$$\pi_A\pi_B\pi_E\big[\mathsf{sKey}^{A,B}, \mathsf{AutC}^{1,A,B}\big] \equiv \mathsf{SecC}^{1,A,B}, \tag{3.1}$$

$$\pi_A\pi_E\big[\mathsf{sKey}^{A,B}, \mathsf{AutC}^{1,A,B}\big] \equiv \sigma_B\mathsf{SecC}^{1,A,B}, \tag{3.2}$$

$$\pi_A\pi_B\big[\mathsf{sKey}^{A,B}, \mathsf{AutC}^{1,A,B}\big] \equiv \sigma_E\mathsf{SecC}^{1,A,B}, \tag{3.3}$$

$$\pi_A\big[\mathsf{sKey}^{A,B}, \mathsf{AutC}^{1,A,B}\big] \equiv \sigma_B\sigma_E\mathsf{SecC}^{1,A,B}. \tag{3.4}$$

At the beginning, the resource $\pi_A\big[\mathsf{sKey}^{A,B}, \mathsf{AutC}^{1,A,B}\big]$ outputs a key at interface B. Hence, (3.4) implies that $\sigma_B\sigma_E\mathsf{SecC}^{1,A,B}$ also outputs a key at interface B. Furthermore, (3.4) implies that on input a message $m \in \mathcal{M} = \{0,1\}^\ell$ at interface A of $\sigma_B\sigma_E\mathsf{SecC}^{1,A,B}$ afterwards, the outputs at interfaces B and E agree. Since σ_E does not get any information about the message and the key had been output before the message was input, no output of σ_B depends on the message. However, we can conclude from (3.1) and (3.2) that $(\pi_B\sigma_B)^B\mathsf{SecC}^{1,A,B} \equiv \mathsf{SecC}^{1,A,B}$, i.e., applying protocol π_B to the output of σ_B gives the correct message. This is only possible with probability $2^{-\ell}$, yielding a contradiction. $\square$

3.2.2 The One-Time Pad with a Dishonest Receiver

As discussed in the preceding section, ordinary encryption at best constructs a secure channel that potentially allows the receiver Bob to leak the message to an eavesdropper Eve and potentially gives common randomness to Bob and Eve if they are both dishonest. In this section, we show that, even though it is impossible to construct a secure channel without additional features from a shared secret key and an authenticated channel, the one-time pad can be used to construct a stronger resource than the one constructed by ordinary encryption, namely a resource that does not allow Bob to leak the message anymore once he has received it.

Intuitively, this is because the one-time pad shares the feature with so-called *deniable encryption* [CDNO97] that for each pair of message and ciphertext one can find a key such that the ciphertext decrypts to the given message with this key. In case of the one-time pad, such a key can be obtained by computing the bitwise XOR of the message and the ciphertext. Therefore, Bob can generate a fake key to yield any message of his choice if he already knows the message. This makes receiving a key from Bob in that case useless because there is no way to verify whether it is the correct key. This translates to the ideal resource by not allowing Bob to leak the message after receiving it. Note that this does not work before Bob knows the message. Then, he cannot find a key that will decrypt the ciphertext to a message of his choice.

To capture the fact that Bob can leak the message before he receives it but not afterwards, we split the receiver into two phases, B_1 that is active at the beginning and B_2 that is active after receiving a message. This is necessary to model for example that a receiver is following the protocol at first but changes his strategy depending on the received message. We first describe the real resource used in our construction, which consists of a key, an authenticated channel, and a memory, and is denoted by KAcM.

Definition 3.2.2. The resource KAcM has the four interfaces A, B_1, B_2, and E and consists of a shared secret key sKey^{A,B_1} between A and B_1, an authenticated channel AutC^{1,A,B_2} from A to B_2, and an ℓ-bit memory Mem^{B_1,B_2} writable by B_1 and readable by B_2. At interface B_1, one can

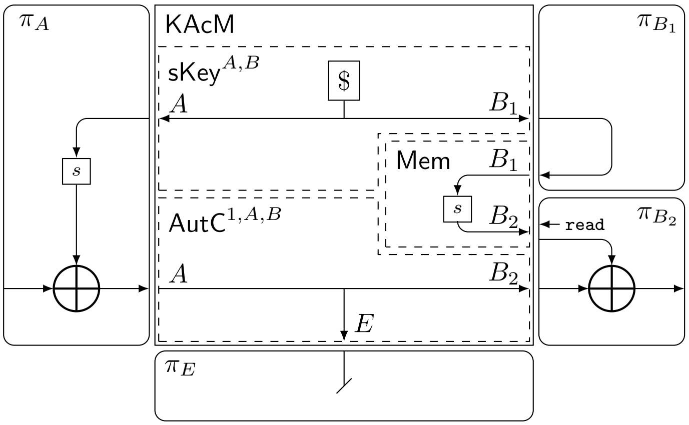


Figure 3.1: The resource KAcM, which consists of sKey^{A,B_1}, AutC^{1,A,B_2}, and Mem^{B_1,B_2} composed in parallel, with the protocol converters attached.

input $x \in \{0,1\}^\ell$ to store x in the memory. On input `read` at interface B_2, the beforehand stored value x is returned to B_2. To model that B_1 and B_2 exist in different phases of the protocol, all inputs at interface B_1 after something is input at interface A and all inputs at interface B_2 before are ignored.

Now we describe the protocol $\pi = (\pi_A, \pi_{B_1}, \pi_{B_2}, \pi_E)$. Let π_A internally store the key it receives at its inside interface, and on input a message at the outside interface, compute the bitwise XOR of this message and the key and input the result into the authenticated channel to B_2. When π_{B_1} receives the key, it stores it in the memory. When π_{B_2} receives a ciphertext from A, it reads the key from the memory, computes the bitwise XOR of the key and the ciphertext and outputs the result at its outside interface. Eve's (hypothetical) protocol $\pi_E := \perp$ ignores all inputs. See Figure 3.1 for an overview of the protocol and the involved resources.

We next describe the ideal resource which is constructed from KAcM by that protocol. The guaranteed functionality allows A to send a message $m \in \mathcal{M}$ to B_2. Moreover, it potentially allows E to learn the length of the message and B_1 to flip a switch such that E afterwards potentially receives the message instead of only its length. Also, B_2 can potentially see whether the switch was flipped. For the same reason we explained in the case of general encryption, B_1, B_2, and E could potentially get common randomness. If B_1 and B_2 are both dishonest, they cannot be prevented from using the memory in KAcM to exchange information. Hence, the ideal resource potentially also allows B_1 to store an ℓ-bit string which can later be read by B_2. We call this resource *secure channel with limited leakability* and formally define it as follows.

Definition 3.2.3. The resource SecCLL has the four interfaces A, B_1, B_2, and E and works as described and illustrated in Figure 3.2. Since some of the functionality of SecCLL is not guaranteed by the protocol but only potentially available, we introduce the following filters: $\phi_{B_1} := \perp$ and $\phi_E := \perp$ ignore all inputs, ϕ_{B_2} converts inputs of the form (r, b, m) at its inside interface to m and ignores other inputs, and $\phi_A := \mathbf{1}$ forwards all inputs. Let $\phi := (\phi_A, \phi_{B_1}, \phi_{B_2}, \phi_E)$.

We finally prove that this protocol actually achieves the construction.

Theorem 3.2.4. *The protocol π defined above constructs SecCLL_ϕ from KAcM with potentially dishonest B_1, B_2, and E, i.e.,*

$$\mathsf{KAcM} \underset{\{B_1, B_2, E\}}{\overset{\pi}{\Longmapsto}} \mathsf{SecCLL}_\phi.$$

Proof. Let $\sigma := (\sigma_{B_1}, \sigma_{B_2}, \sigma_E)$ for the simulators σ_{B_1}, σ_{B_2}, and σ_E defined in Figure 3.3. We have to show that

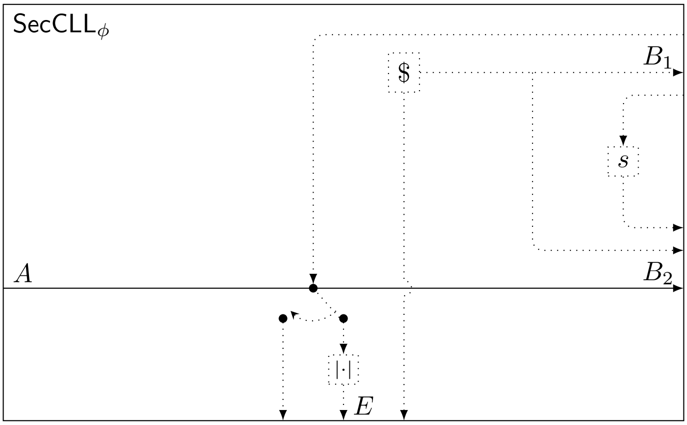
$$\forall \mathcal{P} \subseteq \{B_1, B_2, E\}\ \pi_{\overline{\mathcal{P}}}\mathsf{KAcM} \equiv \sigma_{\mathcal{P}}\phi_{\overline{\mathcal{P}}}\mathsf{SecCLL}.$$

We first verify the conditions with $B_1 \in \mathcal{P}$, i.e., with σ_{B_1} present on the right hand side. In this case, σ_{B_1} outputs a uniformly random ℓ-bit string r at the beginning, as the resource on the left hand side does at interface B_1. Since σ_{B_1} outputs `leak` at its inside interface, the local variable b in SecCLL is `true` when a message m is input at interface A. Hence, σ_{B_2} and σ_E (if present) both output $m \oplus r$. As in the resource on the left hand side, the bitwise XOR of the outputs at interfaces B_1 and B_2 as well as the outputs at interfaces B_1 and E yields the input message $m = r \oplus (m \oplus r)$. On input `read` at interface B_2, σ_{B_2} returns the value stored before a message was input at interface A. Therefore, the resources on the left and those on the right hand side are indistinguishable in these four cases.

Resource SecCLL

```
Initialization
  $b \leftarrow$ false
  $t \leftarrow 1$
  $r \twoheadleftarrow \{0,1\}^\ell$
  output $r$ at interface $B_1$

Interface $A$
Input: $m \in M$
  if $t = 1$ then
      $t \leftarrow 2$
      output $(r, b, m)$ at interface $B_2$
      if $b$ then
          output $(r, m)$ at interface $E$
      else
          output $(r, |m|)$ at interface $E$

Interface $B_1$
Input: leak
  if $t = 1$ then
      $b \leftarrow$ true

Input: (store, $x$), $x \in \{0,1\}^\ell$
  if $t = 1$ then
      $s \leftarrow x$

Interface $B_2$
Input: read
  if $t = 2$ then
      output $s$ at interface $B_2$
```

(a) Definition of the resource SecCLL.



(b) Illustration of the resource SecCLL$_\phi$. Interactions that are not guaranteed but only potentially available to dishonest parties are drawn with dotted lines.

Figure 3.2: The resource SecCLL$_\phi$.

Converter σ_{B_1}

```
Inside interface
Input: $r$
  output leak at inside interface
  output $r$ at outside interface

Outside interface
Input: $x \in \{0,1\}^\ell$
  output (store, $x$) at inside interface
```

Converter σ_E

```
Inside interface
Input: $(r, m)$
  output $m \oplus r$ at outside interface

Input: $(r, |m|)$
  output $r$ at outside interface
```

Converter σ_{B_2}

```
Initialization
  $s \leftarrow$ null

Inside interface
Input: $(r, b, m)$
  if $b$ then
    $s \leftarrow$ returned value from read
                 at inside interface
    output $m \oplus r$ at outside interface
  else
    $s \leftarrow m \oplus r$
    output $r$ at outside interface

Outside interface
Input: read
  if $s \neq$ null then
    output $s$ at outside interface
```

Figure 3.3: Simulators for the construction of SecCLL_ϕ from KAcM.

Now consider the cases with $B_1 \notin \mathcal{P}$. There, b is `false` when a message m is input at interface A, so σ_E will output a uniformly random ℓ-bit string r in case $E \in \mathcal{P}$. If $B_2 \notin \mathcal{P}$, this is indistinguishable from the output at interface E of the resource on the left hand side. Otherwise, σ_{B_2} outputs r as well and sets its internal variable s to $m \oplus r$. Hence, inputting `read` and computing the bitwise XOR of the returned value s and the previous output at interface B_2 results in the message $m = (m \oplus r) \oplus r$, as in the resource on the left hand side. Therefore, the resources are indistinguishable in all cases. □

Chapter 4

Identity-Based Encryption

4.1 Introduction

4.1.1 Motivation

Identity-based encryption (IBE) is a generalization of public-key encryption where messages can be encrypted using a master public key and the *identity* of a user, which can be an arbitrary bit string, such as the user's e-mail address. Ciphertexts can be decrypted with a user secret key for the corresponding identity, where user secret keys are derived from a master secret key, which is generated together with the master public key.

The apparent standard application of IBE is non-interactive secure communication. More specifically, we assume a setting with many parties, and the goal is to enable each party to send any other party (known only by his/her identity) messages in a secure way. This secure communication should be non-interactive (or "one-shot") in the sense that the sending party should not be required to, e.g., look up a public key of the receiving party, or to communicate in any other way (beyond of course sending one message to the receiver). In fact, our requirements and expectations can be described as follows. We define a resource that provides the following basic services (via appropriate calls to the resource):

Registration. Each party is able to *register* his/her identity *id*. (Intuitively, an identity could be an email address or telephone number, that—presumably uniquely—identifies the registering party.)

Communication. Each party is able to *send* a message m to another party with identity id.

While an IBE scheme can be used in an obvious way to syntactically realize this functionality, the application is only secure if the IBE scheme satisfies a suitable security definition.

4.1.2 Identity-Based Encryption and its Security

The concept of identity-based encryption has been conceived as early as 1984 [Sha85]. A first candidate of an IBE scheme was presented in 1991 in [MY91], although without a detailed security model. In the 2000s, however, both a detailed security model [BF01] and a number of concrete IBE schemes (with security proofs under various assumptions) emerged, e.g., [BF01; Coc01; Wat05; GPV08].

Both standard IBE security notions (IND-ID-CPA and IND-ID-CCA) are formalized as a security game. In this game, a hypothetical adversary $\mathcal{A}$ chooses an identity id^*, and messages m_0^* and m_1^*, and tries to distinguish an encryption of m_0^* from an encryption of m_1^* (both prepared for receiver identity id^*). Besides, $\mathcal{A}$ may (adaptively) ask for arbitrary user secret keys for identities $id \neq id^*$. (In case of IND-ID-CCA security, $\mathcal{A}$ additionally gets access to a decryption oracle for arbitrary identities.) If no efficient $\mathcal{A}$ can successfully distinguish these ciphertexts, we consider the system secure.

At this point, we note that these game-based notions of security do allow for a form of adaptivity (in the sense that $\mathcal{A}$ may adaptively ask for user secret keys), but do not directly consider a concrete communication scenario.

4.1.3 Contributions

In this chapter, we investigate the goal of non-interactive communication, and in particular the use of IBE schemes to achieve that goal. Perhaps surprisingly, it turns out that the standard notions of IBE security do *not* imply non-interactive communication in the standard model. However, we prove that standard IBE security notions do imply non-interactive communication in the random oracle model and also weaker forms of non-interactive communication in the standard model. (Loosely speaking, standard IBE security notions achieve non-interactive communication in a setting in which registrations always occur *before* any attempt is made to send messages to the respective receiving party.) Furthermore, we introduce a new security notion that is weaker than the standard notion, but still implies a very natural weaker notion of non-interactive communication in the standard model.

A more technical view. A little more technically, we model non-interactive communication as a "delivery controlled channels" resource DCC.[1] This resource has a number of interfaces, called A, $B_1, \ldots, B_n$, and C, to the involved users. Intuitively, interface C is used to register parties, A is used to send messages[2], and the interfaces B_i are used to receive messages by different parties.

More specifically, our resource admits the following types of queries:

- *Registration* queries (made at interface C) register an interface B_i for receiving messages sent to an identity id. (Depending on the envisioned physical registration process, the *fact* that B_i was registered under identity id may become public. We model this by leaking the pair (id, i) at all interfaces B_j.)

- *Send* queries (at interface A) send a message m to a given identity id. (The message will then be delivered to all interfaces which have been registered for this identity. Besides, any interface B_i which is *later* registered for that identity id will also receive m upon registration.)

- When thinking of an IBE scheme as realizing DCC, we cannot prevent dishonest parties from sharing their keys in the real world. As a result, also the messages sent to that party are shared with every party that got the key. Our ideal system DCC has to make this explicit, so we admit *share* queries (at any interface B_i) that

[1]The name "delivery controlled channels" indicates that a user can specify (or, control) to which recipient the message should be delivered.

[2]In this chapter, we focus on passive attacks (i.e., on eavesdropping adversaries). In particular, we will not consider adversarially sent messages. Thus, for simplicity, we will assume that all incoming requests to *send* a message arrive at a single interface A.

cause all messages sent to this interface to be *potentially*[3] published at all other interfaces B_j that have also made a *share* query.

Furthermore, all parties (i.e., all interfaces B_i) at the beginning (potentially) receive an honestly generated random string (that corresponds to the randomness in the master public key of an IBE scheme that can potentially be extracted). We deem an IBE scheme secure if it implements this resource (when used in the straightforward way) in the sense of constructive cryptography. (In particular, this means that the view of any given party using the real IBE scheme can be simulated efficiently with access to the ideal non-interactive communication resource only.) We note that we do not model secret keys or ciphertexts in our ideal resource.

We remark that a possible ideal functionality in the UC setting would not use interfaces, but instead restrict the registration, send, and share queries to different parties. That is, only a designated "master party" could *register* other parties for receiving messages under certain identities. Every party P could *send* messages, and also issue a *share* query (with the same consequences as in our CC-based formulation).

Why current game-based definitions do not realize DCC. Our first observation is that existing game-based definitions of IBE security (such as IND-ID-CPA or IND-ID-CCA) do not appear to realize the above resource. To explain the reason, suppose that one party P performs its own registration (under an arbitrary identity and at an arbitrary interface B_i) *after* messages are sent to P. (Naturally, P will not be able to receive these messages before obtaining his/her own user secret key during registration.) Now we claim that P's view in that scenario cannot be simulated efficiently. Concretely, observe that P's view with a real IBE scheme essentially consists of two elements: first, a ciphertext c of a yet-unknown message m sent by another party; and second, a user secret key *usk* that allows to decrypt c to m. In order to simulate P's view, a simulator must thus first produce a ciphertext c at a point at which P is not registered as a receiving party. Since at that point, m is not yet known to P, c must in fact be simulated without knowledge of m. Later

[3]Sharing is not guaranteed because our real system does not include channels between the B_i (since they are not needed). When composed with other systems, it might however be the case that such channels become available, so sharing cannot be excluded in a composable framework.

on, however, the simulator must also produce a user secret key usk that opens c as an encryption of m.

Put differently, the simulation thus faces a commitment problem: first, it has to commit to a ciphertext c, and later explain this ciphertext as an encryption of an arbitrary message m. For technically very similar reasons, public-key encryption cannot be simulated in the face of *adaptive* corruptions [Nie02]. (However, we stress that in our case, no adaptive corruptions occur; see also the remark below.) As a consequence, we can show that non-interactive communication (as formalized by our resource DCC) cannot be achieved in the standard model.

Weaker notions of non-interactive communication. Our negative result for the above resource DCC raises the question what we can do to achieve *some* form of non-interactive communication and also what existing, game-based IBE security notions actually achieve.

Recall that the commitment problem that arises with DCC occurs only when identities are registered *after* messages have been sent to this identity. A natural way to avoid this scenario is to assume first a registration phase (in which no message transmissions are allowed), and second a transmission phase (in which no registrations are allowed). This separation into two phases can be modeled as a resource $\mathsf{st2DCC}$ that only allows message transmissions (and from then on ignores registration attempts) after a specific input at the "registration" interface C.[4] We can show that $\mathsf{st2DCC}$ *can* be achieved by IND-ID-CPA secure IBE schemes. In that sense, the commitment problem of DCC is the *only* reason why we cannot achieve that resource. Interestingly, achieving $\mathsf{st2DCC}$ actually corresponds to a game-based notion of IBE security that we introduce and call IND-ID1-CPA security and that is weaker than IND-ID-CPA security.

We also show that IND-ID-CPA security exactly corresponds to a resource stDCC which only allows registrations of identities to which no message has been sent so far. (In that sense, stDCC implements a "local" version of the two-phase separation of $\mathsf{st2DCC}$. Again, we stress

[4]While this separation is easily modeled as a resource, we stress that it is the responsibility of the (designer of the) implementation to physically enforce this separation. For instance, in face of a passive adversary, such a separation into phases could be enforced simply by telling honest parties not to send any messages until the second phase.

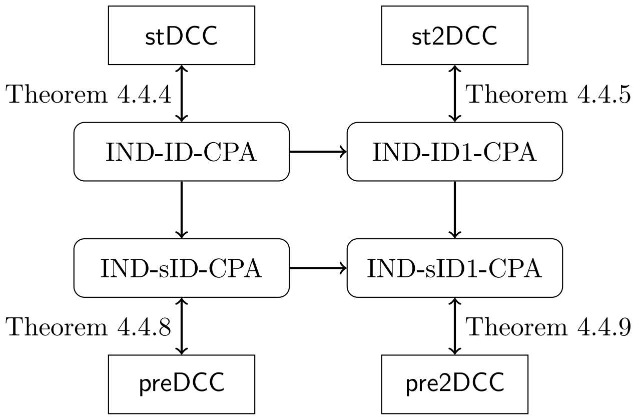


Figure 4.1: Implications among security definitions and the constructed resources. Security definitions are drawn in boxes with rounded corners and resources are shown in rectangular boxes. The figure says for example that by Theorem 4.4.4, an IBE scheme can be used to construct the resource stDCC if and only if it is IND-ID-CPA secure, while IND-ID-CPA security implies IND-sID-CPA security and IND-ID1-CPA security.

that it is the responsibility of the implementation to enforce such a local separation.)

Finally, we provide relaxed resources preDCC and pre2DCC that are "selective" versions of stDCC and st2DCC, respectively. (Here, "selective" means that the set of identities id that can be registered has to be specified initially, over interface A.) We proceed to show that resource preDCC is achieved precisely by selective IND-ID-CPA secure IBE schemes. Similarly, the resource pre2DCC is equivalent to a selective version of the game-based notion associated with the resource st2DCC. The relations among security definitions and the achieved constructions are summarized in Figure 4.1.

Relevance of the impossibility result. While it perhaps appears natural to process all registrations before messages for the corresponding identities are sent, this restriction substantially weakens the usefulness of IBE. For example, if IBE is used in a large context to encrypt emails where the encryption service is independent of the email providers, it seems desirable to be able to send encrypted emails to anyone with a valid email address, without knowing whether they have already registered for the encryption service. In fact, if one has to "ask" whether a user has already received his key before being able to send him a message, one gives up non-interactivity and does not gain much compared to standard public-key encryption.

Moreover, an interesting application, which was suggested in [BF01], is impossible: Assume the key authority every day publishes a key for the identity that corresponds to the current date. One should now be able to send a message "to the future" by encrypting it for the identity corresponding to, e.g., the following day. We are here precisely in the situation where a ciphertext is received before the corresponding key, so standard IBE does not guarantee the security of this application (our construction with random oracles, however, does provide this guarantee).[5]

On dishonest senders. The results in this chapter only consider passive attacks, i.e., we assume only honest parties send messages. This makes our impossibility result only stronger, and all positive results can in principle be lifted to a setting with potentially dishonest senders by replacing the CPA-definitions with their (R)CCA-counterparts. However, this leads to some subtleties in the modeling. For example, one needs to simulate a dishonest sender sending some nonsensical bit string (which does not constitute a valid ciphertext) to a dishonest receiver. Furthermore, the two phases in the results with a separate registration and transmission phase become intermixed, because only honest parties are prevented from sending during the registration phase. To avoid such technicalities and simplify the presentation, we formulate all results only for honest senders.

[5]One can give a less technical argument why standard definitions are insufficient for this application than the inability to simulate: It is not excluded by IND-ID-CPA or IND-ID-CCA that first providing a ciphertext and later the user secret key for the corresponding identity yields a binding commitment (maybe only for some specific subset of the message space). In this case, a dishonest recipient Bob of a ciphertext for the following day can use this ciphertext to commit himself (to some third party) to the encrypted value, and open the commitment on the next day. Note that Bob committed himself to a value *he did not know*, possibly misleading the third party into believing he knew it, which is not possible when an ideal "sending-to-the-future" functionality is used.

4.1.4 Related Work

On the difference to the ideal IBE functionality of Nishimaki, Manabe, and Okamoto. We note that an ideal functionality for identity-based encryption has already been presented by Nishimaki et al. [NMO06] in the UC framework. However, unlike our resources (when interpreted as UC functionalities as sketched above), their functionality was constructed directly along the IBE *algorithms*, and not to model the *goal* of non-interactive communication. Besides, their functionality does not guarantee secrecy for ciphertexts generated before the respective receiver has been initialized. (This relaxed guarantee can be understood as corresponding to our relaxed resource stDCC that disallows registrations after communication attempts to the given identity, where instead of disallowing such registrations, the functionality by Nishimaki et al. allows them, but then gives up the security guarantees.)

As a consequence, [NMO06] could indeed show that the standard game-based definition of security for IBE schemes is equivalent to realizing their ideal functionality. Specifically, their IBE abstraction thus compares differently from ours to game-based IBE security notions.

Relation to adaptive corruptions in the public-key setting. As noted, *technically*, the commitment problem we encounter is very similar to the commitment problem faced in adaptively secure public-key encryption [Nie02]. There, a simulation would have to first produce a ciphertext (without knowing the supposed plaintext). Later, upon an adaptive corruption of the respective receiver, the simulation would have to provide a secret key that opens that ciphertext suitably.

However, in our case, the actual *setting* in which the problem occurs is not directly related to corruptions. Namely, in our setting, a similar commitment problem occurs because messages may be sent to an identity prior to an "activation" of the corresponding communication channel. (In fact, since the mapping of receiving parties to identities may not be clear beforehand, prior to such an activation it is not even clear where to route the corresponding sent messages.) Hence, we can argue that the commitment problem we face is inherent to the IBE setting, independently of adaptive corruptions (all results in this thesis are actually formulated for static corruptions).

4.2 Delivery Controlled Channels

4.2.1 Overview and General Definition

A broadcast channel allows a sender A to send messages to recipients $B_1, \ldots, B_n$. One can understand the application of an IBE scheme to add some form of delivery control to such a channel. More specifically, the enhanced channel allows A to send a message for some identity id in an identity space $\mathcal{ID}$ such that only the B_i that are registered for this identity receive the message, even if several other B_i are dishonest. We assume this registration is managed by a central authority C. We formalize this by a *delivery controlled channel* DCC. This resource also allows the registration of identities after messages have been sent for this identity. In this case, the corresponding user after registration learns all such messages.

Because the public key and each ciphertext contain randomness, during initialization and for each sent message, all parties (potentially) receive common randomness. Moreover, when someone gets registered for an identity, this identity together with a corresponding user secret key is sent to this party over a secure channel. By definition, a secure channel can leak the length of the transmitted messages. Since the length of user secret keys can depend on the identity for which the key has been generated and also on the used randomness, dishonest users potentially learn which identity has just been registered for whom and potentially even which randomness was used to generate the corresponding secret key. Furthermore, dishonest recipients can share their secret keys with others in the real world, which has the effect in the ideal world that the other recipients also learn the messages sent for an identity that has been registered for the user who shared his keys. We model this by a special symbol `share` that B_i can input. A message sent for identity id is then received by B_i if id has been registered for B_i or if there is a B_j such that B_i and B_j have input `share` and id has been registered for B_j.

Definition 4.2.1. Let $n, \rho \in \mathbb{N}$, $\mathcal{M} := \{0,1\}^*$, and let $\mathcal{ID}$ be a nonempty set. The resource $\mathsf{DCC}^{n,\mathcal{ID},\rho}$ has the interfaces A, C, and B_i for $i \in \{1, \ldots, n\}$. The resource internally manages the set $S \subseteq \{B_1, \ldots, B_n\}$ of interface names that want to share their identities and for each $i \in \{1, \ldots, n\}$, the set $I_i \subseteq \mathcal{ID}$ of identities registered for interface B_i. Initially, these sets are empty. The resource works as described in Figure 4.2.

Resource $\mathsf{DCC}^{n,\mathcal{ID},\rho}$

```
Initialization
  $S \leftarrow \emptyset$
  $j \leftarrow 1$
  $r \twoheadleftarrow \{0,1\}^\rho$
  for all $i \in \{1,\dots,n\}$ do
      $I_i \leftarrow \emptyset$
      output $r$ at interface $B_i$

Interface $A$
Input: $(id_j, m_j) \in \mathcal{ID} \times \mathcal{M}$
  $r_j \twoheadleftarrow \{0,1\}^\rho$
  for all $i \in \{1,\dots,n\}$ do
      if $id_j \in I_i \vee \big(B_i \in S$
                  $\wedge\ id_j \in \bigcup_{B_k \in S} I_k\big)$ then
          output $(id_j, m_j, r_j)$ at int. $B_i$
      else
          output $(id_j, |m_j|, r_j)$ at int. $B_i$
  $j \leftarrow j + 1$

Interface $B_i$
Input: share
  $S \leftarrow S \cup \{B_i\}$

Interface $C$
Input: $(id, i) \in \mathcal{ID} \times \{1,\dots,n\}$
  $I_i \leftarrow I_i \cup \{id\}$
  $r \twoheadleftarrow \{0,1\}^\rho$
  for all $k \in \{1,\dots,n\}$ do
      output $(id, i, r)$ at interface $B_k$
      if $k = i \vee \{B_i, B_k\} \subseteq S$ then
          for all $l \in \{1,\dots,j-1\}$ do
              if $id_l = id$ then
                  output $m_l$ at int. $B_k$
```

Figure 4.2: Definition of the resource $\mathsf{DCC}^{n,\mathcal{ID},\rho}$.

The randomness that the B_i get corresponds to randomness one can potentially extract from the public key, the ciphertexts, and the length of the user secret keys of an IBE scheme. Honest users are not guaranteed to receive this randomness, we rather cannot exclude that dishonest parties do so. Similarly, we cannot exclude that dishonest parties share their identities, learn the identity for which a message is designated and the length of the message without being registered for that identity, or learn who gets registered for which identity. To model that these interactions are not guaranteed, we introduce the following filters: For $i \in \{1,\dots,n\}$, let $\phi^{\mathsf{DCC}}_{B_i}$ be the converter that on input $(id, m, r) \in \mathcal{ID} \times \mathcal{M} \times \{0,1\}^\rho$ at its inside interface, outputs (id, m) at its outside interface, on input $m \in \mathcal{M}$ at its inside interface, outputs m at its outside interface, and on input $(id, k, r) \in \mathcal{ID} \times \{1,\dots,n\} \times \{0,1\}^\rho$ with $k = i$ at its inside interface, outputs id at its outside interface. All other inputs at any of its interfaces are ignored and thereby blocked. Further let $\phi^{\mathsf{DCC}}_A := \phi^{\mathsf{DCC}}_C := \mathbf{1}$ be the converter that forwards all inputs at one of its interfaces to the other one and let $\phi^{\mathsf{DCC}} := (\phi^{\mathsf{DCC}}_A, \phi^{\mathsf{DCC}}_C, \phi^{\mathsf{DCC}}_{B_1}, \dots, \phi^{\mathsf{DCC}}_{B_n})$. We will consider the filtered resource $\mathsf{DCC}^{n,\mathcal{ID},\rho}_{\phi^{\mathsf{DCC}}}$.

Remark. The resource defined above assumes that a central authority C registers all identities and allows one party to have more than one identity and one identity to be registered for several users. That resource can now be used in larger context where this registration process is regulated. For example, one can have a protocol programmed on top of DCC that requires B_i to send his identity together with a copy of his passport to C. Moreover, C could ensure that each identity is registered for at most one user. In such an application, the resource DCC could directly be used without considering how it was constructed. Due to composition of the constructive cryptography framework, we can thus focus on the construction of DCC and decouple confidentiality from the actual registration process.

4.2.2 Static Identity Management

We now define a more restricted resource that only allows the registration of an identity as long as no message has been sent for this identity.

Definition 4.2.2. Let $n, \rho \in \mathbb{N}$, $\mathcal{M} := \{0,1\}^*$, and let $\mathcal{ID}$ be a nonempty set. The resource $\mathsf{stDCC}^{n,\mathcal{ID},\rho}$ is identical to $\mathsf{DCC}^{n,\mathcal{ID},\rho}$ except that inputs $(id, i) \in \mathcal{ID} \times \{1,\dots,n\}$ at interface C are ignored if $id \in \bigcup_{k=1}^{j-1}\{id_k\}$. We will use the same filters as above and consider the resource $\mathsf{stDCC}^{n,\mathcal{ID},\rho}_{\phi^{\mathsf{DCC}}}$.

The above resource prevents identities for which messages have been sent to be registered, but other identities can still be registered. The following resource restricts the registration process further and operates in two phases: Initially, only registrations are allowed and no messages can be sent. At any point, C can end the registration phase and enable A to send messages.

Definition 4.2.3. Let $n, \rho \in \mathbb{N}$, $\mathcal{M} := \{0,1\}^*$, and let $\mathcal{ID}$ be a nonempty set. The resource $\mathsf{st2DCC}^{n,\mathcal{ID},\rho}$ behaves as $\mathsf{DCC}^{n,\mathcal{ID},\rho}$ except that it initially ignores all inputs at interface A. On input the special symbol `endRegistration` at interface C, it outputs `registrationEnded` at interfaces $B_1, \dots, B_n$, and from then on ignores all inputs at interface C and allows inputs at interface A. We will consider the filtered resource $\mathsf{st2DCC}^{n,\mathcal{ID},\rho}_{\phi^{\mathsf{DCC}}}$.

Note that ϕ^{DCC} blocks the output `registrationEnded` for honest users, i.e., it is not necessarily guaranteed that everyone learns that the registration has ended. It is not excluded by our protocol since C there informs A that messages may now be sent, and this communication could be observed by dishonest users. If it is desirable in an application that everyone learns that the registration has ended, one can still use $\mathsf{st2DCC}^{n,\mathcal{ID},\rho}$ by letting C explicitly send that information to all B_i via an additional channel. This would happen outside of the resource $\mathsf{st2DCC}^{n,\mathcal{ID},\rho}$ as a separate construction.

Further note that in stDCC, A can prevent the registration of an identity by sending a message for it. On the other hand, $\mathsf{st2DCC}$ gives C full control over the registration process while being less dynamic. Depending on the application, one or the other resource might be preferable.

4.2.3 Predetermined Identities

We finally introduce two resources that additionally require all identities that are used be determined at the beginning. This allows us to capture the guarantees provided by selectively secure IBE schemes (see Definition 4.3.3).

Definition 4.2.4. Let $n, \rho \in \mathbb{N}$, $\mathcal{M} := \{0,1\}^*$, and let $\mathcal{ID}$ be a nonempty set. The resources $\mathsf{preDCC}^{n,\mathcal{ID},\rho}$ and $\mathsf{pre2DCC}^{n,\mathcal{ID},\rho}$ have the interfaces A, C, and B_i for $i \in \{1,\ldots,n\}$. Before the resources output anything or accept any input, they wait for the input of a finite set $\mathcal{J} \subseteq \mathcal{ID}$ (encoded as a list of its elements) at interface A. On this input, they output `ok` at interfaces $B_1,\ldots,B_n$. Afterwards, $\mathsf{preDCC}^{n,\mathcal{ID},\rho}$ behaves identically to $\mathsf{stDCC}^{n,\mathcal{ID},\rho}$ and $\mathsf{pre2DCC}^{n,\mathcal{ID},\rho}$ behaves identically to $\mathsf{st2DCC}^{n,\mathcal{ID},\rho}$ with the exception that they only accept inputs $(id_j, m_j) \in \mathcal{J} \times \mathcal{M}$ at interface A (there is no restriction on inputs at interface C). We will again consider the filtered resources $\mathsf{preDCC}^{n,\mathcal{ID},\rho}_{\phi^{\mathsf{DCC}}}$ and $\mathsf{pre2DCC}^{n,\mathcal{ID},\rho}_{\phi^{\mathsf{DCC}}}$.[6]

4.3 IBE Schemes and Protocols

4.3.1 Standard Definitions for IBE

We first give the formal definition of an IBE scheme.

[6]Again, the filter ϕ^{DCC} blocks the outputs `ok` and `registrationEnded` at interfaces B_i.

Definition 4.3.1. An *identity-based encryption (IBE) scheme* $\mathcal{E}$ with message space $\mathcal{M}$ and identity space $\mathcal{ID}$ consists of the following four PPT algorithms:

Key generation: On input a security parameter 1^κ, the algorithm Gen outputs a *master public key* mpk and a *master secret key* msk.

Extraction: On input a master secret key msk and an identity $id \in \mathcal{ID}$, the algorithm Ext outputs a *user secret key* usk_{id}.

Encryption: On input a master public key mpk, an identity $id \in \mathcal{ID}$, and a message $m \in \mathcal{M}$, the algorithm Enc outputs a *ciphertext* c.

Decryption: On input a user secret key usk_{id}, an identity $id \in \mathcal{ID}$, and a ciphertext c, the algorithm Dec outputs a message $m \in \mathcal{M} \cup \{\perp\}$.

For correctness, we require that for all $(mpk, msk) \leftarrow \mathsf{Gen}(1^\kappa)$, all $id \in \mathcal{ID}$, all $m \in \mathcal{M}$, all $c \leftarrow \mathsf{Enc}(mpk, id, m)$, and all $usk_{id} \leftarrow \mathsf{Ext}(msk, id)$, we always have $\mathsf{Dec}(usk_{id}, id, c) = m$.

We now provide the standard security definition for IBE schemes against passive attacks.

Definition 4.3.2 (IND-ID-CPA security)**.** For an IBE scheme $\mathcal{E} = (\mathsf{Gen}, \mathsf{Ext}, \mathsf{Enc}, \mathsf{Dec})$ and an algorithm $\mathcal{A}$, consider the experiment $\mathsf{Exp}^{\mathsf{IND\text{-}ID\text{-}CPA}}_{\mathcal{E},\mathcal{A}}$ in Figure 4.3. In this experiment, $\mathcal{A}$ is not allowed to output an identity id that it has queried to its Ext oracle, or to later query id to Ext. Furthermore, $\mathcal{A}$ must output m_0, m_1 of equal length. Let

$$\mathsf{Adv}^{\mathsf{IND\text{-}ID\text{-}CPA}}_{\mathcal{E},\mathcal{A}} := 2 \cdot \Pr\left[\mathsf{Exp}^{\mathsf{IND\text{-}ID\text{-}CPA}}_{\mathcal{E},\mathcal{A}} = 1\right] - 1.$$

We say that $\mathcal{E}$ has *indistinguishable ciphertexts under chosen-plaintext attacks (is IND-ID-CPA secure)* if $\mathsf{Adv}^{\mathsf{IND\text{-}ID\text{-}CPA}}_{\mathcal{E},\mathcal{A}}$ is negligible for all PPT $\mathcal{A}$.

We further consider a weaker security notion introduced in [CHK03] where the adversary has to specify the identity he wants to attack at the beginning of the experiment.

Definition 4.3.3 (IND-sID-CPA security)**.** Consider the experiment $\mathsf{Exp}^{\mathsf{IND\text{-}sID\text{-}CPA}}_{\mathcal{E},\mathcal{A}}$ in Figure 4.3 for an IBE scheme $\mathcal{E} = (\mathsf{Gen}, \mathsf{Ext}, \mathsf{Enc}, \mathsf{Dec})$

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

```
Input: $1^\kappa, \kappa \in \mathbb{N}$
  $(mpk, msk) \leftarrow \mathsf{Gen}(1^\kappa)$
  $(st, id, m_0, m_1) \leftarrow \mathcal{A}^{\mathsf{Ext}(msk,\cdot)}(mpk)$
  $b \twoheadleftarrow \{0,1\}$
  $c^* \leftarrow \mathsf{Enc}(mpk, id, m_b)$
  $b' \leftarrow \mathcal{A}^{\mathsf{Ext}(msk,\cdot)}(st, c^*)$
  return 1 if $b' = b$, else return 0
```

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

```
Input: $1^\kappa, \kappa \in \mathbb{N}$
  $(st, id) \leftarrow \mathcal{A}(1^\kappa)$
  $(mpk, msk) \leftarrow \mathsf{Gen}(1^\kappa)$
  $(st', m_0, m_1) \leftarrow \mathcal{A}^{\mathsf{Ext}(msk,\cdot)}(st, mpk)$
  $b \twoheadleftarrow \{0,1\}$
  $c^* \leftarrow \mathsf{Enc}(mpk, id, m_b)$
  $b' \leftarrow \mathcal{A}^{\mathsf{Ext}(msk,\cdot)}(st', c^*)$
  return 1 if $b' = b$, else return 0
```

Figure 4.3: The IND-(s)ID-CPA experiment with scheme $\mathcal{E}$ and adversary $\mathcal{A}$.

and an algorithm $\mathcal{A}$. In this experiment, $\mathcal{A}$ is not allowed to query id to Ext and has to output m_0, m_1 of equal length. Let

$$\mathsf{Adv}_{\mathcal{E},\mathcal{A}}^{\mathsf{IND\text{-}sID\text{-}CPA}} := 2 \cdot \Pr\left[\mathsf{Exp}_{\mathcal{E},\mathcal{A}}^{\mathsf{IND\text{-}sID\text{-}CPA}} = 1\right] - 1.$$

We say that $\mathcal{E}$ has *indistinguishable ciphertexts under selective identity, chosen-plaintext attacks (is IND-sID-CPA secure)* if $\mathsf{Adv}_{\mathcal{E},\mathcal{A}}^{\mathsf{IND\text{-}sID\text{-}CPA}}$ is negligible for all PPT $\mathcal{A}$.

4.3.2 Non-Adaptive Security

We next introduce two novel security notions for IBE schemes that loosely correspond to variants of the standard definitions under "lunchtime attacks" [NY90]. While CCA1 in contrast to CCA allows the adversary only to ask decryption queries in an initial phase, our definitions restrict the adversary to ask Ext queries only in an initial phase.

Definition 4.3.4 (IND-(s)ID1-CPA security)**.** Consider the two experiments $\mathsf{Exp}_{\mathcal{E},\mathcal{A}}^{\mathsf{IND\text{-}ID1\text{-}CPA}}$ and $\mathsf{Exp}_{\mathcal{E},\mathcal{A}}^{\mathsf{IND\text{-}sID1\text{-}CPA}}$ for an IBE scheme $\mathcal{E} = (\mathsf{Gen}, \mathsf{Ext}, \mathsf{Enc}, \mathsf{Dec})$ and an algorithm $\mathcal{A}$ in Figure 4.4. In these experiments, $\mathcal{A}$ is only considered valid if all queries to its Ext oracle are different from id and if $|m_0| = |m_1|$. Let

$$\mathsf{Adv}_{\mathcal{E},\mathcal{A}}^{\mathsf{IND\text{-}ID1\text{-}CPA}} := 2 \cdot \Pr\left[\mathsf{Exp}_{\mathcal{E},\mathcal{A}}^{\mathsf{IND\text{-}ID1\text{-}CPA}} = 1\right] - 1,$$
$$\mathsf{Adv}_{\mathcal{E},\mathcal{A}}^{\mathsf{IND\text{-}sID1\text{-}CPA}} := 2 \cdot \Pr\left[\mathsf{Exp}_{\mathcal{E},\mathcal{A}}^{\mathsf{IND\text{-}sID1\text{-}CPA}} = 1\right] - 1.$$

Experiment $\mathsf{Exp}^{\mathsf{IND\text{-}ID1\text{-}CPA}}_{\mathcal{E},\mathcal{A}}$

```
Input: $1^\kappa, \kappa \in \mathbb{N}$
    $(mpk, msk) \leftarrow \mathsf{Gen}(1^\kappa)$
    $st \leftarrow \mathcal{A}^{\mathsf{Ext}(msk,\cdot)}(1^\kappa)$
    $(st', id, m_0, m_1) \leftarrow \mathcal{A}(st, mpk)$
    $b \twoheadleftarrow \{0,1\}$
    $c^* \leftarrow \mathsf{Enc}(mpk, id, m_b)$
    $b' \leftarrow \mathcal{A}(st', c^*)$
    return 1 if $b' = b$, else return 0
```

Experiment $\mathsf{Exp}^{\mathsf{IND\text{-}sID1\text{-}CPA}}_{\mathcal{E},\mathcal{A}}$

```
Input: $1^\kappa, \kappa \in \mathbb{N}$
    $(st, id) \leftarrow \mathcal{A}(1^\kappa)$
    $(mpk, msk) \leftarrow \mathsf{Gen}(1^\kappa)$
    $st' \leftarrow \mathcal{A}^{\mathsf{Ext}(msk,\cdot)}(st)$
    $(st'', m_0, m_1) \leftarrow \mathcal{A}(st', mpk)$
    $b \twoheadleftarrow \{0,1\}$
    $c^* \leftarrow \mathsf{Enc}(mpk, id, m_b)$
    $b' \leftarrow \mathcal{A}(st'', c^*)$
    return 1 if $b' = b$, else return 0
```

Figure 4.4: The IND-(s)ID1-CPA experiment with scheme $\mathcal{E}$ and adversary $\mathcal{A}$.

We say that $\mathcal{E}$ has *indistinguishable ciphertexts under non-adaptive chosen-plaintext attacks (is IND-ID1-CPA secure)* if $\mathsf{Adv}^{\mathsf{IND\text{-}ID1\text{-}CPA}}_{\mathcal{E},\mathcal{A}}$ is negligible for all valid PPT $\mathcal{A}$, and $\mathcal{E}$ has *indistinguishable ciphertexts under selective identity, non-adaptive chosen-plaintext attacks (is IND-sID1-CPA secure)* if $\mathsf{Adv}^{\mathsf{IND\text{-}sID1\text{-}CPA}}_{\mathcal{E},\mathcal{A}}$ is negligible for all valid PPT $\mathcal{A}$.

4.3.3 Using IBE Schemes in Constructions

In this section, we define the real resources we assume to be available and describe the protocol converters that are designed to construct the resources defined in Section 4.2. These protocol converters internally use an IBE scheme. Whether the constructions are achieved according to Definition 2.4.6 depends on the security properties of the IBE scheme, which we analyze in Section 4.4.

Delivery Controlled Channels. To construct a delivery controlled channel from a broadcast channel[7], we use an IBE scheme in a straightforward way: The party at interface C generates all keys, sends the public key authentically to A and the user secret keys securely to the corresponding B_i. To send a message, A broadcasts an encryption thereof and the B_i

[7]Note that we consider the sender to be honest. Hence, assuming a broadcast channel to be available is not a strong assumption.

with matching identity decrypt it. Hence, we need in addition to the broadcast channel an authenticated channel from C to A to transmit the public key and secure channels from C to each B_i. We abbreviate the network consisting of these channels as

$$\mathsf{NW} := \left[\mathsf{BCast}^{A,\{B_1,\dots,B_n\}}, \mathsf{AutC}^{1,C,A}, \mathsf{SecC}^{C,B_1}, \dots, \mathsf{SecC}^{C,B_n}\right].$$

The real resource in our construction corresponds to the filtered resource $\mathsf{NW}_{\phi^{\mathsf{NW}}}$ where $\phi^{\mathsf{NW}} := (\phi^{\mathsf{NW}}_A, \phi^{\mathsf{NW}}_C, \phi^{\mathsf{NW}}_{B_1}, \dots, \phi^{\mathsf{NW}}_{B_n})$ with $\phi^{\mathsf{NW}}_I := [\mathbf{1}, \phi^{\mathsf{AutC}}_I, \phi^{\mathsf{SecC}}_I, \dots, \phi^{\mathsf{SecC}}_I]$ for $I \in \{A, C, B_1, \dots, B_n\}$.[8]

For an IBE scheme $\mathcal{E}$, we define protocol converters enc, dec, and reg as follows and let $\mathsf{ibe} := (\mathsf{enc}, \mathsf{reg}, \mathsf{dec}, \dots, \mathsf{dec})$: The converter enc first expects to receive a master public key mpk at its inside interface and stores it internally. On input a message and identity $(id, m) \in \mathcal{ID} \times \mathcal{M}$ at its outside interface, it computes $c \leftarrow \mathsf{Enc}(mpk, id, m)$ and outputs (id, c) at its inside sub-interface to $\mathsf{BCast}^{A,\{B_1,\dots,B_n\}}$. The converter dec on input an identity and a corresponding user secret key (id, usk_{id}) at its inside interface, stores this tuple internally and outputs id at its outside interface. For all pairs (id_j, c_j) with $id_j = id$ stored internally, dec computes $m_j \leftarrow \mathsf{Dec}(usk_{id}, id, c_j)$ and outputs m_j at its outside interface. On input an identity and a ciphertext (id, c) at its inside interface, it stores (id, c) internally and if it has stored a user secret key for the identity id, computes $m \leftarrow \mathsf{Dec}(usk_{id}, id, c)$ and outputs (id, m) at its outside interface. The converter reg initially computes $(mpk, msk) \leftarrow \mathsf{Gen}(1^\kappa)$, stores msk internally, and outputs mpk at its inside sub-interface to $\mathsf{AutC}^{1,C,A}_{\phi^{\mathsf{AutC}}}$. On input (id, i) at its outside interface, it computes $usk_{id} \leftarrow \mathsf{Ext}(msk, id)$ and outputs (id, usk_{id}) at its inside sub-interface to $\mathsf{SecC}^{C,B_i}_{\phi^{\mathsf{SecC}}}$.

Static identity management. To construct stDCC, the protocol at interface C has to reject registration requests for identities for which

[8]In this context, the channel SecC^{C,B_i} is a resource with $n+2$ interfaces where interface C corresponds to interface A of the resource in Definition 2.4.3, interface B_i corresponds to interface B, and interfaces B_j for $j \neq i$ correspond to copies of interface E. Similarly, ϕ^{SecC}_C corresponds to ϕ^{SecC}_A in Definition 2.4.4, $\phi^{\mathsf{SecC}}_{B_i}$ corresponds to ϕ^{SecC}_B, and $\phi^{\mathsf{SecC}}_{B_j}$ to ϕ^{SecC}_E for $j \neq i$. For simplicity, we do not introduce a different notation for the different filters.

messages have already been sent. To be able to do so, it needs to know for which identities this is the case. We thus assume there is an additional authenticated channel from A to C that is used to inform C about the usage of identities. The real resource is then $\mathsf{NW}^{+}_{\phi^{\mathsf{NW}+}}$ for

$$\mathsf{NW}^{+} := \Big[\mathsf{BCast}^{A,\{B_1,\dots,B_n\}}, \mathsf{AutC}^{A,C}, \mathsf{AutC}^{1,C,A}, \\ \mathsf{SecC}^{C,B_1}, \dots, \mathsf{SecC}^{C,B_n}\Big]$$

and $\phi^{\mathsf{NW}^+} := (\phi_A^{\mathsf{NW}^+}, \phi_C^{\mathsf{NW}^+}, \phi_{B_1}^{\mathsf{NW}^+}, \dots, \phi_{B_n}^{\mathsf{NW}^+})$ where $\phi_I^{\mathsf{NW}} := [\mathbf{1}, \phi_I^{\mathsf{AutC}}, \phi_I^{\mathsf{AutC}}, \phi_I^{\mathsf{SecC}}, \dots, \phi_I^{\mathsf{SecC}}]$ for $I \in \{A, C, B_1, \dots, B_n\}$.

We define the protocol $\mathsf{ibe}^{\mathsf{s}} := (\mathsf{enc}^{\mathsf{s}}, \mathsf{reg}^{\mathsf{s}}, \mathsf{dec}^{\mathsf{s}}, \dots, \mathsf{dec}^{\mathsf{s}})$ by describing the differences from ibe as follows: On input $(id, m) \in \mathcal{ID} \times \mathcal{M}$ at its outside interface, $\mathsf{enc}^{\mathsf{s}}$ additionally outputs id at its inside interface to $\mathsf{AutC}^{A,C}_{\phi^{\mathsf{AutC}}}$. The converter $\mathsf{reg}^{\mathsf{s}}$ on input id at its inside interface, stores this identity internally. It subsequently ignores inputs (id, i) at its outside interface if it has stored id.

Note that it is crucial for this construction that $\mathsf{AutC}^{A,C}$ cannot be interrupted or delayed. Otherwise an attacker could prevent C from learning that some identity has already been used to send messages and this identity could still be registered. In practice, one could realize such a channel by letting C acknowledge the receipt while A sends the message only after receiving this acknowledgment. This would, however, contradict the goal of non-interactivity.

If such a reliable channel is not available, we can still construct $\mathsf{st2DCC}$ from NW using the protocol $\mathsf{ibe}^{\mathsf{2s}} := (\mathsf{enc}^{\mathsf{2s}}, \mathsf{reg}^{\mathsf{2s}}, \mathsf{dec}^{\mathsf{2s}}, \dots, \mathsf{dec}^{\mathsf{2s}})$ defined as follows: It works as ibe, except that $\mathsf{reg}^{\mathsf{2s}}$ initially does not send mpk to A. On input `endRegistration` at its outside interface, $\mathsf{reg}^{\mathsf{2s}}$ sends mpk to A and ignores further inputs. The converter $\mathsf{enc}^{\mathsf{2s}}$ ignores all inputs until it receives mpk at its inside interface and from then on handles all inputs as enc.

Remark. Note that sending mpk is here used to signal A that it can now start sending messages. Since we assume that the sender is always honest, we do not need to require, e.g., that mpk cannot be computed from user secret keys; as long as mpk has not been sent, A will not send any messages.

Predetermined identities. To construct $\mathsf{preDCC}_{\phi^{\mathsf{DCC}}}$ from $\mathsf{NW}^{+}_{\phi^{\mathsf{NW+}}}$, we define the protocol $\mathsf{ibe}^{\mathsf{p}} = (\mathsf{enc}^{\mathsf{p}}, \mathsf{reg}^{\mathsf{p}}, \mathsf{dec}^{\mathsf{p}}, \ldots, \mathsf{dec}^{\mathsf{p}})$ that uses a selectively secure IBE scheme. The protocol is almost identical to $\mathsf{ibe}^{\mathsf{s}}$ with the difference that $\mathsf{enc}^{\mathsf{p}}$ initially expects a finite set $\mathcal{J} \subseteq \mathcal{ID}$ (encoded as a list of its elements) as input at its outside interface. On this input, it stores $\mathcal{J}$ internally, sends $\mathtt{ok}$ to C via $\mathsf{AutC}^{A,C}_{\phi^{\mathsf{AutC}}}$, and subsequently ignores all inputs (id, m) for $id \notin \mathcal{J}$. The converter $\mathsf{reg}^{\mathsf{p}}$ initially waits and ignores all inputs at its outside interface until it receives the input $\mathtt{ok}$ at its inside interface. It then sends mpk to A and from then on behaves identically to $\mathsf{reg}^{2\mathsf{s}}$.

Similarly, we define a protocol $\mathsf{ibe}^{2\mathsf{p}} = (\mathsf{enc}^{2\mathsf{p}}, \mathsf{reg}^{2\mathsf{p}}, \mathsf{dec}^{2\mathsf{p}}, \ldots, \mathsf{dec}^{2\mathsf{p}})$ to construct $\mathsf{pre2DCC}_{\phi^{\mathsf{DCC}}}$ from $\mathsf{NW}^{+}_{\phi^{\mathsf{NW+}}}$. It works as ibe except that $\mathsf{enc}^{2\mathsf{p}}$ initially expects a finite set $\mathcal{J} \subseteq \mathcal{ID}$ (encoded as a list of its elements) as input at its outside interface. On this input, it stores $\mathcal{J}$ internally, sends $\mathtt{ok}$ to C via $\mathsf{AutC}^{A,C}_{\phi^{\mathsf{AutC}}}$, and ignores all further inputs until it receives mpk over $\mathsf{AutC}^{C,A}_{\phi^{\mathsf{AutC}}}$. From then on, it handles all inputs as enc, but ignores inputs (id, m) for $id \notin \mathcal{J}$. The converter $\mathsf{reg}^{2\mathsf{p}}$ initially waits and ignores all inputs at its outside interface until it receives the input $\mathtt{ok}$ at its inside interface. It then accepts registration requests at its outside interface as reg. On input $\mathtt{endRegistration}$ at its outside interface, $\mathsf{reg}^{2\mathsf{p}}$ sends mpk to A and ignores further inputs.

Remark. While both $\mathsf{ibe}^{\mathsf{p}}$ and $\mathsf{ibe}^{2\mathsf{p}}$ need $\mathsf{AutC}^{A,C}_{\phi^{\mathsf{AutC}}}$, $\mathsf{ibe}^{2\mathsf{p}}$ uses this channel only once in the beginning to let A send $\mathtt{ok}$ to C. The availability of such a channel only at the beginning might be easier to guarantee in practice.

4.4 Constructions with IBE

4.4.1 Impossibility Result

We now show that there is no IBE scheme that can be used to construct $\mathsf{DCC}_{\phi^{\mathsf{DCC}}}$ from $\mathsf{NW}_{\phi^{\mathsf{NW}}}$.

Theorem 4.4.1. *Let $n > 0$, $\mathcal{ID}$ a nonempty set, and let $\rho \in \mathbb{N}$. Then there is no IBE scheme such that we have for the corresponding protocol* ibe

$$\mathsf{NW}_{\phi^{\mathsf{NW}}} \quad \underset{\{B_1, \ldots, B_n\}}{\overset{\mathsf{ibe}}{\Longmapsto}} \quad \mathsf{DCC}^{n,\mathcal{ID},\rho}_{\phi^{\mathsf{DCC}}}.$$

Proof. This proof closely resembles Nielsen's impossibility proof of non-committing public-key encryption [Nie02]. Assume the protocol $\mathsf{ibe} = (\mathsf{enc}, \mathsf{reg}, \mathsf{dec}, \ldots, \mathsf{dec})$ achieves the construction and let $\mathcal{P} := \{B_1\}$. Then there exists a converter σ_{B_1} such that $\mathsf{ibe}_{\overline{\mathcal{P}}} \phi^{\mathsf{NW}}_{\overline{\mathcal{P}}} \mathsf{NW} \approx \sigma_{\mathcal{P}} \phi^{\mathsf{DCC}}_{\overline{\mathcal{P}}} \mathsf{DCC}^{n,\mathcal{ID},\rho}$. Let $id \in \mathcal{ID}$, let ν be an upper bound on the length of the output of $\mathsf{Ext}(\cdot, id)$, and consider the following distinguisher: The distinguisher $\mathcal{D}$ chooses $m \in \{0,1\}^{\nu+1}$ uniformly at random and inputs (id, m) at interface A. Let (id, c) be the resulting output at interface B_1 (if there is no such output, $\mathcal{D}$ returns 0). Then, $\mathcal{D}$ inputs $(id, 1)$ at interface C. Let (id, usk) be the resulting output at interface B_1 and return 0 if there is no such output or if $|usk| > \nu$. Finally, $\mathcal{D}$ inputs first (id, c) and then (id, usk) at the inside interface of dec and returns 1 if dec outputs id and m at its outside interface, and 0 otherwise.

Correctness of the IBE scheme implies that $\mathcal{D}$ always outputs 1 if connected to the real resource. In the ideal world, c is generated independently of m only given $|m|$ because σ_{B_1} does not learn m until $(id, 1)$ is input at interface C. Moreover, there are at most 2^ν possible values for usk such that $|usk| \leq \nu$. Hence, there are at most 2^ν values of m such that there exists a usk that decrypts c to m with probability more than $\frac{1}{2}$. Since m was chosen uniformly from $\{0,1\}^{\nu+1}$, $\mathcal{D}$ outputs 1 with probability at most $\frac{1}{2} + \frac{1}{2} \cdot \frac{1}{2} = \frac{3}{4}$ when connected to the ideal resource. Thus, the distinguishing advantage is at least $\frac{1}{4}$, which is a contradiction. □

4.4.2 Construction Equivalent to IND-ID-CPA

While no IBE scheme constructs $\mathsf{DCC}_{\phi^{\mathsf{DCC}}}$ from $\mathsf{NW}_{\phi^{\mathsf{NW}}}$, we show that IND-ID-CPA security is sufficient to construct $\mathsf{stDCC}_{\phi^{\mathsf{DCC}}}$ from $\mathsf{NW}^+_{\phi^{\mathsf{NW}^+}}$.

Lemma 4.4.2. *Let ρ be an upper bound on the randomness used in one invocation of* Gen*,* Ext*, and* Enc*. Then, there exist efficient converters $\sigma_{B_1}, \ldots, \sigma_{B_n}$ such that for all $\mathcal{P} \subseteq \{B_1, \ldots, B_n\}$ and for all efficient distinguishers $\mathcal{D}$ that input at most q messages at interface A, there exists an efficient algorithm $\mathcal{A}$ such that*

$$\Delta^{\mathcal{D}}\left(\mathsf{ibe}^{\mathsf{s}}_{\overline{\mathcal{P}}} \phi^{\mathsf{NW}^+}_{\overline{\mathcal{P}}} \mathsf{NW}^+, \sigma_{\mathcal{P}} \phi^{\mathsf{DCC}}_{\overline{\mathcal{P}}} \mathsf{stDCC}^{n,\mathcal{ID},\rho}\right) = q \cdot \mathsf{Adv}^{\mathsf{IND\text{-}ID\text{-}CPA}}_{\mathcal{E},\mathcal{A}}.$$

Proof. Consider the simulator σ_{B_i} for $i \in \{1, \ldots, n\}$ in Figure 4.5, let $\mathcal{P} \subseteq \{B_1, \ldots, B_n\}$, and let $\mathcal{D}$ be an efficient distinguisher for the resources

```
Converter $\sigma_{B_i}$

Inside interface
Input: $r \in \{0,1\}^\rho$
    $(mpk, msk) \leftarrow \mathsf{Gen}(1^\kappa; r)$
    output share at inside interface
    output $mpk$ at outside sub-interface simulating $\mathsf{AutC}^{1,C,A}$

Input: $(id, m, r) \in \mathcal{ID} \times \mathcal{M} \times \{0,1\}^\rho$
    $c \leftarrow \mathsf{Enc}(mpk, id, m; r)$
    output $id$ at outside sub-interface simulating $\mathsf{AutC}^{A,C}$
    output $(id, c)$ at outside sub-interface simulating $\mathsf{BCast}^{A,\{B_1,\ldots,B_n\}}$

Input: $(id, |m|, r) \in \mathcal{ID} \times \mathbb{N} \times \{0,1\}^\rho$
    $c \leftarrow \mathsf{Enc}(mpk, id, 0^{|m|}; r)$
    output $id$ at outside sub-interface simulating $\mathsf{AutC}^{A,C}$
    output $(id, c)$ at outside sub-interface simulating $\mathsf{BCast}^{A,\{B_1,\ldots,B_n\}}$

Input: $(id, k, r) \in \mathcal{ID} \times \{1,\ldots,n\} \times \{0,1\}^\rho$
    $usk \leftarrow \mathsf{Ext}(msk, id; r)$
    if $k = i$ then
        output $(id, usk)$ at outside sub-interface simulating $\mathsf{SecC}^{C,B_i}$
    else
        output $|(id, usk)|$ at outside sub-interface simulating $\mathsf{SecC}^{C,B_k}$
```

Figure 4.5: The simulator σ_{B_i} for the proof of Lemma 4.4.2. It ignores all inputs at its outside interface and handles inputs at its inside interface as described (where other inputs at its inside interface are also ignored).

$\mathsf{ibe}^{\mathsf{s}}_{\overline{\mathcal{P}}}\phi^{\mathsf{NW}^+}_{\overline{\mathcal{P}}}\mathsf{NW}^+$ and $\sigma_{\mathcal{P}}\phi^{\mathsf{DCC}}_{\overline{\mathcal{P}}}\mathsf{stDCC}^{n,\mathcal{ID},\rho}$ that inputs at most q messages at interface A. We assume without loss of generality that $\mathcal{D}$ does not make any inputs that are ignored by both resources. We can further assume that the distinguisher $\mathcal{D}$ does not input (id, i) at interface C for i with $B_i \notin \mathcal{P}$, because correctness of the IBE scheme implies that such inputs cannot improve the distinguishing advantage. Moreover, we can assume that $\mathcal{D}$ does not input $(id, m) \in \mathcal{ID} \times \mathcal{M}$ at interface A if (id, i) was input before at interface C for some i, because such inputs to any of the two resources result in the output of an encryption of m for id at the interfaces $B_i \in \mathcal{P}$ and this result can be simulated by the distinguisher on its own.

We let $\mathcal{A}$ run $\mathcal{D}$ by emulating the resource $\mathcal{D}$ is supposed to be connected to as follows: When algorithm $\mathcal{A}$ is invoked with a master public key mpk, it sets $j \leftarrow 0$, draws $j' \in \{1,\ldots,q\}$ uniformly at random and outputs mpk at the sub-interfaces of B_i corresponding to $\mathsf{AutC}^{1,C,A}$ for all $B_i \in \mathcal{P}$. When $\mathcal{D}$ inputs $(id, i) \in \mathcal{ID} \times \{1, \ldots, n\}$ at interface C, $\mathcal{A}$ makes the oracle-query id to receive usk_{id}. It then outputs (id, usk_{id}) at the sub-interface of B_i corresponding to SecC^{C,B_i} and $|(id, usk_{id})|$ at the sub-interfaces of $B_k \in \mathcal{P}$ corresponding to SecC^{C,B_i} for $k \neq i$. When $\mathcal{D}$ inputs $(id, m) \in \mathcal{ID} \times \mathcal{M}$ at interface A, $\mathcal{A}$ increments j by 1. If $j < j'$, $\mathcal{A}$ computes $c \leftarrow \mathsf{Enc}(mpk, id, m)$ and outputs (id, c) at the sub-interfaces of B_i corresponding to $\mathsf{BCast}^{A,\{B_1,\ldots,B_n\}}$ and id at the sub-interfaces corresponding to $\mathsf{AutC}^{A,C}$ for all $B_i \in \mathcal{P}$. If $j = j'$, $\mathcal{A}$ stores mpk, id, and the state of $\mathcal{D}$ in st, sets $m_0 \leftarrow m$, $m_1 \leftarrow 0^{|m|}$, and returns (st, id, m_0, m_1).

The algorithm $\mathcal{A}$ is then invoked with input (st, c^*). It extracts mpk, id, and the state of $\mathcal{D}$ from st and continues the execution of $\mathcal{D}$ by outputting (id, c^*) at the sub-interfaces of B_i corresponding to $\mathsf{BCast}^{A,\{B_1,\ldots,B_n\}}$ and id at the sub-interfaces corresponding to $\mathsf{AutC}^{A,C}$ for all $B_i \in \mathcal{P}$. When $\mathcal{D}$ inputs $(id, m) \in \mathcal{ID} \times \mathcal{M}$ at interface A, $\mathcal{A}$ computes $c \leftarrow \mathsf{Enc}(mpk, id, 0^{|m|})$ and outputs (id, c) at the sub-interfaces of B_i corresponding to $\mathsf{BCast}^{A,\{B_1,\ldots,B_n\}}$ and id at the sub-interfaces corresponding to $\mathsf{AutC}^{A,C}$ for all $B_i \in \mathcal{P}$. Inputs at interface C are handled as above. Finally $\mathcal{A}$ returns the same bit as $\mathcal{D}$. Note that $\mathcal{A}$ is a valid adversary according to Definition 4.3.2 since $|m_0| = |m_1|$ and it never queries the returned identity to its oracle. The latter is because we assumed that $\mathcal{D}$ does not input (id, m) at interface A if it input (id, i) before at interface C. Moreover, inputting (id, i) at interface C afterwards would be ignored by $\mathsf{ibe}^{\mathsf{s}}_{\mathcal{P}} \phi^{\mathsf{NW}^+}_{\mathcal{P}} \mathsf{NW}^+$ and $\sigma_{\mathcal{P}} \phi^{\mathsf{DCC}}_{\mathcal{P}} \mathsf{stDCC}^{n,\mathcal{ID},\rho}$ and we assumed that $\mathcal{D}$ does not make any inputs that are ignored by both resources.

The relation between the distinguishing advantage of $\mathcal{D}$ and the advantage of $\mathcal{A}$ can be proven by a hybrid argument. To this end, for $i \in \{0, \ldots, q\}$, let H_i be the resource that corresponds to $\mathsf{ibe}^{\mathsf{s}}_{\mathcal{P}} \phi^{\mathsf{NW}^+}_{\mathcal{P}} \mathsf{NW}^+$ for the first i inputs at interface A and afterwards on input $(id, m) \in \mathcal{ID} \times \mathcal{M}$ at interface A outputs $\big(id, \mathsf{Enc}\big(mpk, id, 0^{|m|}\big)\big)$ at the sub-interfaces of B_i corresponding to $\mathsf{BCast}^{A,\{B_1,\ldots,B_n\}}$ and id at the sub-interfaces corresponding to $\mathsf{AutC}^{A,C}$ for all $B_i \in \mathcal{P}$, where mpk corresponds to the initial output of the resource at these interfaces B_i. Note that

$$\Delta^{\mathcal{D}}\big(\mathsf{H}_0, \sigma_{\mathcal{P}} \phi^{\mathsf{DCC}}_{\mathcal{P}} \mathsf{stDCC}^{n,\mathcal{ID},\rho}\big) = \Delta^{\mathcal{D}}\big(\mathsf{H}_q, \mathsf{ibe}^{\mathsf{s}}_{\mathcal{P}} \phi^{\mathsf{NW}^+}_{\mathcal{P}} \mathsf{NW}^+\big) = 0.$$

We further have

$$\Pr\big[\mathsf{Exp}^{\mathsf{IND\text{-}ID\text{-}CPA}}_{\mathcal{E},\mathcal{A}} = 0 \mid b = 0\big] = \frac{1}{q}\sum_{i=1}^{q} \Pr\big[\mathcal{D}\mathsf{H}_i = 1\big]$$

and

$$\Pr\big[\mathsf{Exp}^{\mathsf{IND\text{-}ID\text{-}CPA}}_{\mathcal{E},\mathcal{A}} = 1 \mid b = 1\big] = \frac{1}{q}\sum_{i=1}^{q} \Pr\big[\mathcal{D}\mathsf{H}_{i-1} = 1\big].$$

This yields

$$\begin{aligned}
&\Delta^{\mathcal{D}}\Big(\mathsf{ibe}^{\mathsf{s}}_{\overline{\mathcal{P}}}\phi^{\mathsf{NW}^+}_{\overline{\mathcal{P}}}\mathsf{NW}^+, \sigma_{\mathcal{P}}\phi^{\mathsf{DCC}}_{\overline{\mathcal{P}}}\mathsf{stDCC}^{n,\mathcal{ID},\rho}\Big)\\
&= \Delta^{\mathcal{D}}(\mathsf{H}_0, \mathsf{H}_q)\\
&= \Pr[\mathcal{D}\mathsf{H}_0 = 1] - \Pr[\mathcal{D}\mathsf{H}_q = 1]\\
&= \sum_{i=1}^{q}\Pr[\mathcal{D}\mathsf{H}_{i-1} = 1] - \sum_{i=1}^{q}\Pr[\mathcal{D}\mathsf{H}_i = 1]\\
&= q\cdot\Pr\big[\mathsf{Exp}^{\mathsf{IND\text{-}ID\text{-}CPA}}_{\mathcal{E},\mathcal{A}} = 1 \mid b = 1\big] - q\cdot\Pr\big[\mathsf{Exp}^{\mathsf{IND\text{-}ID\text{-}CPA}}_{\mathcal{E},\mathcal{A}} = 0 \mid b = 0\big]\\
&= q\cdot\Big(\Pr\big[\mathsf{Exp}^{\mathsf{IND\text{-}ID\text{-}CPA}}_{\mathcal{E},\mathcal{A}} = 1 \mid b = 1\big] + \Pr\big[\mathsf{Exp}^{\mathsf{IND\text{-}ID\text{-}CPA}}_{\mathcal{E},\mathcal{A}} = 1 \mid b = 0\big] - 1\Big)\\
&= q\cdot\mathsf{Adv}^{\mathsf{IND\text{-}ID\text{-}CPA}}_{\mathcal{E},\mathcal{A}},
\end{aligned}$$

which concludes the proof. □

We now prove conversely that IND-ID-CPA security is also necessary for the construction:

Lemma 4.4.3. *Let $\rho \in \mathbb{N}$ and $\mathcal{P} \subseteq \{B_1, \ldots, B_n\}, \mathcal{P} \neq \emptyset$. Then, for all valid IND-ID-CPA adversaries $\mathcal{A}$ and for all efficient converters σ_{B_i} for $B_i \in \mathcal{P}$, there exists an efficient distinguisher $\mathcal{D}$ such that*

$$\mathsf{Adv}^{\mathsf{IND\text{-}ID\text{-}CPA}}_{\mathcal{E},\mathcal{A}} = 2\cdot\Delta^{\mathcal{D}}\Big(\mathsf{ibe}^{\mathsf{s}}_{\overline{\mathcal{P}}}\phi^{\mathsf{NW}^+}_{\overline{\mathcal{P}}}\mathsf{NW}^+, \sigma_{\mathcal{P}}\phi^{\mathsf{DCC}}_{\overline{\mathcal{P}}}\mathsf{stDCC}^{n,\mathcal{ID},\rho}\Big).$$

Proof. Let $\mathcal{A}$ be a valid IND-ID-CPA adversary and let σ_{B_i} be efficient converters for $B_i \in \mathcal{P}$. Further let $B_i \in \mathcal{P}$. We now define two distinguishers, $\mathcal{D}_0$ and $\mathcal{D}_1$. Let *mpk* be the initial output at interface B_i of the resource connected to the distinguisher (if nothing is output, let *mpk* be some default value[9]). Both distinguishers then invoke $\mathcal{A}(mpk)$. The oracle query id' of $\mathcal{A}$ is answered as follows by both distinguishers: They input (id', i) at interface C and let the answer to the query be $usk_{id'}$ where $(id', usk_{id'})$ is the resulting output of the resource at interface B_i (and let $usk_{id'}$ be some default value if there is no such output). If $\mathcal{A}$ returns (st, id, m_0, m_1), $\mathcal{D}_0$ and $\mathcal{D}_1$ input (id, m_0) and (id, m_1) at interface A, respectively. Now let (id, c^*) be the resulting output at the sub-interface of B_i corresponding to $\mathsf{BCast}^{A,\{B_1,\ldots,B_n\}}$ (and let c^* be some default value if there is no such output). Both distinguishers then invoke $\mathcal{A}$ on input (st, c^*). Oracle queries are answered as above. Note that id will not be queried since $\mathcal{A}$ is a valid IND-ID-CPA adversary and therefore inputs at interface C will be handled as before. Finally, $\mathcal{D}_0$ and $\mathcal{D}_1$ output the bit returned by $\mathcal{A}$.

Note that for all $\beta \in \{0,1\}$

$$\Pr\bigl[\mathcal{D}_\beta\bigl(\mathsf{ibe}^{\mathsf{s}}_{\overline{\mathcal{P}}}\phi^{\mathsf{NW}^+}_{\overline{\mathcal{P}}}\mathsf{NW}^+\bigr) = 1\bigr] = \Pr\bigl[\mathsf{Exp}^{\mathsf{IND\text{-}ID\text{-}CPA}}_{\mathcal{E},\mathcal{A}} = \beta \bigm| b = \beta\bigr]$$

because the outputs of the real system are precisely generated as the corresponding values in the IND-ID-CPA experiment. Hence,

$$\begin{aligned}
&\mathsf{Adv}^{\mathsf{IND\text{-}ID\text{-}CPA}}_{\mathcal{E},\mathcal{A}}\\
&= 2 \cdot \Pr\bigl[\mathsf{Exp}^{\mathsf{IND\text{-}ID\text{-}CPA}}_{\mathcal{E},\mathcal{A}} = 1\bigr] - 1\\
&= \Pr\bigl[\mathsf{Exp}^{\mathsf{IND\text{-}ID\text{-}CPA}}_{\mathcal{E},\mathcal{A}} = 1 \bigm| b = 0\bigr] + \Pr\bigl[\mathsf{Exp}^{\mathsf{IND\text{-}ID\text{-}CPA}}_{\mathcal{E},\mathcal{A}} = 1 \bigm| b = 1\bigr] - 1\\
&= \Pr\bigl[\mathsf{Exp}^{\mathsf{IND\text{-}ID\text{-}CPA}}_{\mathcal{E},\mathcal{A}} = 1 \bigm| b = 1\bigr] - \Pr\bigl[\mathsf{Exp}^{\mathsf{IND\text{-}ID\text{-}CPA}}_{\mathcal{E},\mathcal{A}} = 0 \bigm| b = 0\bigr]\\
&= \Pr\bigl[\mathcal{D}_1\bigl(\mathsf{ibe}^{\mathsf{s}}_{\overline{\mathcal{P}}}\phi^{\mathsf{NW}^+}_{\overline{\mathcal{P}}}\mathsf{NW}^+\bigr) = 1\bigr] - \Pr\bigl[\mathcal{D}_0\bigl(\mathsf{ibe}^{\mathsf{s}}_{\overline{\mathcal{P}}}\phi^{\mathsf{NW}^+}_{\overline{\mathcal{P}}}\mathsf{NW}^+\bigr) = 1\bigr].
\end{aligned}$$

Further note that we have

$$\Pr\bigl[\mathcal{D}_0\bigl(\sigma_{\mathcal{P}}\phi^{\mathsf{DCC}}_{\overline{\mathcal{P}}}\mathsf{stDCC}^{n,\mathcal{ID},\rho}\bigr) = 1\bigr] = \Pr\bigl[\mathcal{D}_1\bigl(\sigma_{\mathcal{P}}\phi^{\mathsf{DCC}}_{\overline{\mathcal{P}}}\mathsf{stDCC}^{n,\mathcal{ID},\rho}\bigr) = 1\bigr]$$

since $\mathcal{D}_0$ and $\mathcal{D}_1$ only differ in the message they input and σ_{B_i} only learns the length of that message, which is the same for the two messages (since $\mathcal{A}$ is a valid IND-ID-CPA adversary), so its output does not depend on the choice of the message. Now let $\mathcal{D}$ be the distinguisher that chooses

[9]Note that this is only possible in the ideal system if σ_{B_i} is flawed. Hence, one could distinguish better in this case, but we do not need that for the proof.

$\beta \in \{0,1\}$ uniformly at random, runs $\mathcal{D}_\beta$, and outputs 1 if $\mathcal{D}_\beta$ outputs β, and 0 otherwise. Then, $\Pr\big[\mathcal{D}\big(\sigma_{\mathcal{P}}\phi^{\mathsf{DCC}}_{\overline{\mathcal{P}}}\mathsf{stDCC}^{n,\mathcal{ID},\rho}\big) = 1\big] = \frac{1}{2}$, and thus

$$\begin{aligned}
&\mathsf{Adv}^{\mathsf{IND\text{-}ID\text{-}CPA}}_{\mathcal{E},\mathcal{A}} \\
&= \Pr\big[\mathcal{D}_1\big(\mathsf{ibe}^{\mathsf{s}}_{\overline{\mathcal{P}}}\phi^{\mathsf{NW}^+}_{\overline{\mathcal{P}}}\mathsf{NW}^+\big) = 1\big] - \Pr\big[\mathcal{D}_0\big(\mathsf{ibe}^{\mathsf{s}}_{\overline{\mathcal{P}}}\phi^{\mathsf{NW}^+}_{\overline{\mathcal{P}}}\mathsf{NW}^+\big) = 1\big] \\
&= \Pr\big[\mathcal{D}_0\big(\mathsf{ibe}^{\mathsf{s}}_{\overline{\mathcal{P}}}\phi^{\mathsf{NW}^+}_{\overline{\mathcal{P}}}\mathsf{NW}^+\big) = 0\big] - \Pr\big[\mathcal{D}_0\big(\sigma_{\mathcal{P}}\phi^{\mathsf{DCC}}_{\overline{\mathcal{P}}}\mathsf{stDCC}^{n,\mathcal{ID},\rho}\big) = 0\big] \\
&\quad + \Pr\big[\mathcal{D}_1\big(\mathsf{ibe}^{\mathsf{s}}_{\overline{\mathcal{P}}}\phi^{\mathsf{NW}^+}_{\overline{\mathcal{P}}}\mathsf{NW}^+\big) = 1\big] - \Pr\big[\mathcal{D}_1\big(\sigma_{\mathcal{P}}\phi^{\mathsf{DCC}}_{\overline{\mathcal{P}}}\mathsf{stDCC}^{n,\mathcal{ID},\rho}\big) = 1\big] \\
&= 2 \cdot \Pr\big[\mathcal{D}\big(\mathsf{ibe}^{\mathsf{s}}_{\overline{\mathcal{P}}}\phi^{\mathsf{NW}^+}_{\overline{\mathcal{P}}}\mathsf{NW}^+\big) = 1\big] - 1 \\
&= 2 \cdot \Delta^{\mathcal{D}}\big(\mathsf{ibe}^{\mathsf{s}}_{\overline{\mathcal{P}}}\phi^{\mathsf{NW}^+}_{\overline{\mathcal{P}}}\mathsf{NW}^+, \sigma_{\mathcal{P}}\phi^{\mathsf{DCC}}_{\overline{\mathcal{P}}}\mathsf{stDCC}^{n,\mathcal{ID},\rho}\big).
\end{aligned}$$

This concludes the proof. □

Lemmata 4.4.2 and 4.4.3 together imply the following theorem:

Theorem 4.4.4. *Let ρ be an upper bound on the randomness used in one invocation of* Gen*,* Ext*, and* Enc*. We then have*

$$\mathsf{NW}^+_{\phi^{\mathsf{NW}^+}} \quad \overset{\mathsf{ibe}^{\mathsf{s}}}{\underset{\{B_1,\ldots,B_n\}}{\Longmapsto}} \quad \mathsf{stDCC}^{n,\mathcal{ID},\rho}_{\phi^{\mathsf{DCC}}}$$

$\iff$ *the underlying IBE scheme is IND-ID-CPA secure.*

The following theorem can be proven very similarly by observing that the reductions used to prove Theorem 4.4.4 translate queries to the Ext oracle by the adversary to inputs at interface C by the distinguisher and vice versa and that $\mathsf{NW}_{\phi^{\mathsf{NW}}}$ and $\mathsf{st2DCC}^{n,\mathcal{ID},\rho}_{\phi^{\mathsf{DCC}}}$ restrict such inputs exactly as $\mathcal{A}$ is restricted in $\mathsf{Exp}^{\mathsf{IND\text{-}ID1\text{-}CPA}}_{\mathcal{E},\mathcal{A}}$.

Theorem 4.4.5. *Let ρ be an upper bound on the randomness used in one invocation of* Gen*,* Ext*, and* Enc*. We then have*

$$\mathsf{NW}_{\phi^{\mathsf{NW}}} \quad \overset{\mathsf{ibe}^{2\mathsf{s}}}{\underset{\{B_1,\ldots,B_n\}}{\Longmapsto}} \quad \mathsf{st2DCC}^{n,\mathcal{ID},\rho}_{\phi^{\mathsf{DCC}}}$$

$\iff$ *the underlying IBE scheme is IND-ID1-CPA secure.*

4.4.3 Construction Equivalent to IND-sID-CPA

We now prove that IND-sID-CPA security is sufficient to construct $\mathsf{preDCC}_{\phi^{\mathsf{DCC}}}$ from $\mathsf{NW}^+_{\phi^{\mathsf{NW}+}}$.

Lemma 4.4.6. *Let ρ be an upper bound on the randomness used in one invocation of* Gen*,* Ext*, and* Enc*. Then, there exist efficient converters $\sigma_{B_1}, \ldots \sigma_{B_n}$ such that for all $\mathcal{P} \subseteq \{B_1, \ldots, B_n\}$ and for all efficient distinguishers $\mathcal{D}$ that input a set of identities of size at most μ and at most q messages at interface A, there exists an efficient algorithm $\mathcal{A}$ such that*

$$\Delta^{\mathcal{D}}\left(\mathsf{ibe}^{\mathsf{p}}_{\overline{\mathcal{P}}}\phi^{\mathsf{NW}^+}_{\overline{\mathcal{P}}}\mathsf{NW}^+, \sigma_{\mathcal{P}}\phi^{\mathsf{DCC}}_{\overline{\mathcal{P}}}\mathsf{preDCC}^{n,\mathcal{ID},\rho}\right) \leq \mu q \cdot \mathsf{Adv}^{\mathsf{IND\text{-}sID\text{-}CPA}}_{\mathcal{E},\mathcal{A}}.$$

Proof. Let $\mathcal{P} \subseteq \{B_1, \ldots, B_n\}$ and let σ_{B_i} process all inputs as the simulator in the proof of Lemma 4.4.2 and in addition on input `ok` at its inside interface, output `ok` at its outside interface. We again assume that $\mathcal{D}$ is an efficient distinguisher that does not make inputs that do not increase the distinguishing advantage, i.e., $\mathcal{D}$ does not make any inputs that are ignored by both resources, does not input (id, i) at interface C for i with $B_i \notin \mathcal{P}$, and does not input $(id, m) \in \mathcal{ID} \times \mathcal{M}$ at interface A if (id, i) was input before at interface C for some i. We further assume that $\mathcal{D}$ initially inputs a nonempty set $\mathcal{J} \subseteq \mathcal{ID}$ at interface A because otherwise it cannot input anything at interface A and the distinguishing advantage is 0 in this case. Moreover, we assume that there is always an identity in $\mathcal{J}$ that $\mathcal{D}$ does not input at interface C since by our other assumptions, $\mathcal{D}$ would otherwise again not input any message at interface A and have distinguishing advantage 0.

We let $\mathcal{A}$ emulate an execution of $\mathcal{D}$ as follows: When $\mathcal{D}$ inputs a set of identities $\mathcal{J} \subseteq \mathcal{ID}$ at interface A, $\mathcal{A}$ outputs `ok` at the sub-interface of B_i corresponding to $\mathsf{AutC}^{1,A,C}$ for all $B_i \in \mathcal{P}$, chooses one element in $\mathcal{J}$ uniformly at random, and returns it as the challenge identity id^* together with the state of $\mathcal{D}$ and id^* in st. When algorithm $\mathcal{A}$ is invoked with input (st, mpk), it continues the execution of $\mathcal{D}$, sets $j \leftarrow 0$, draws $j' \in \{1, \ldots, q\}$ uniformly at random, and outputs mpk at the sub-interfaces of B_i corresponding to $\mathsf{AutC}^{1,C,A}$ for all $B_i \in \mathcal{P}$. When $\mathcal{D}$ inputs $(id, i) \in (\mathcal{ID} \setminus \{id^*\}) \times \{1, \ldots, n\}$ at interface C, $\mathcal{A}$ makes the oracle-query id to receive usk_{id}. It then outputs (id, usk_{id}) at the sub-interface of B_i corresponding to SecC^{C,B_i} and $|(id, usk_{id})|$ at the sub-interfaces of

$B_k \in \mathcal{P}$ corresponding to SecC^{C,B_i} for $k \neq i$. If $\mathcal{D}$ inputs (id^*, i) for some i at interface C, $\mathcal{A}$ terminates and returns a uniform bit. When $\mathcal{D}$ inputs $(id, m) \in \mathcal{ID} \times \mathcal{M}$ at interface A, $\mathcal{A}$ increments j by 1. If $j < j'$, $\mathcal{A}$ computes $c \leftarrow \mathsf{Enc}(mpk, id, m)$ and outputs (id, c) at the sub-interfaces of B_i corresponding to $\mathsf{BCast}^{A,\{B_1,\dots,B_n\}}$ and id at the sub-interfaces corresponding to $\mathsf{AutC}^{A,C}$ for all $B_i \in \mathcal{P}$. If $j = j'$ and $id = id^*$, $\mathcal{A}$ stores mpk, id^*, and the state of $\mathcal{D}$ in st', sets $m_0 \leftarrow m$, $m_1 \leftarrow 0^{|m|}$, and returns (st', id, m_0, m_1). If $j = j'$ and $id \neq id^*$, $\mathcal{A}$ terminates and returns a uniform bit. When $\mathcal{A}$ is invoked with input (st', c^*), it continues the execution of $\mathcal{D}$ by outputting (id^*, c^*) at the sub-interfaces of B_i corresponding to $\mathsf{BCast}^{A,\{B_1,\dots,B_n\}}$ and id at the sub-interfaces corresponding to $\mathsf{AutC}^{A,C}$ for all $B_i \in \mathcal{P}$. When $\mathcal{D}$ inputs $(id, m) \in \mathcal{ID} \times \mathcal{M}$ at interface A, $\mathcal{A}$ computes $c \leftarrow \mathsf{Enc}(mpk, id, 0^{|m|})$ and outputs (id, c) at the sub-interfaces of B_i corresponding to $\mathsf{BCast}^{A,\{B_1,\dots,B_n\}}$ and id at the sub-interfaces corresponding to $\mathsf{AutC}^{A,C}$ for all $B_i \in \mathcal{P}$. Inputs at interface C are handled as above. Finally, when $\mathcal{D}$ outputs the bit b', $\mathcal{A}$ returns the bit $1 - b'$.

We now introduce essentially the same hybrids as in the proof of Lemma 4.4.2. More precisely, for $i \in \{0, \dots, q\}$, let H_i be the resource that corresponds to $\mathsf{ibe}^{\mathsf{p}}_{\mathcal{P}} \phi^{\mathsf{NW}^+}_{\mathcal{P}} \mathsf{NW}^+$ for the first i inputs of the form (id, m) at interface A and afterwards on input $(id, m) \in \mathcal{ID} \times \mathcal{M}$ at interface A outputs $\big(id, \mathsf{Enc}\big(mpk, id, 0^{|m|}\big)\big)$ at the sub-interfaces of B_i corresponding to $\mathsf{BCast}^{A,\{B_1,\dots,B_n\}}$ and id at the sub-interfaces corresponding to $\mathsf{AutC}^{A,C}$ for all $B_i \in \mathcal{P}$, where mpk corresponds to the initial output of the resource at these interfaces B_i. We then again have

$$\Delta^{\mathcal{D}}\big(\mathsf{H}_0, \sigma_{\mathcal{P}} \phi^{\mathsf{DCC}}_{\mathcal{P}} \mathsf{preDCC}^{n,\mathcal{ID},\rho}\big) = \Delta^{\mathcal{D}}\big(\mathsf{H}_q, \mathsf{ibe}^{\mathsf{p}}_{\mathcal{P}} \phi^{\mathsf{NW}^+}_{\mathcal{P}} \mathsf{NW}^+\big) = 0.$$

For $i \in \{1, \dots, q\}$, we define a random variable Q_i in the experiment that involves $\mathcal{A}$ internally running $\mathcal{D}$ as described above as follows: If the i-th input at interface A (not counting the input of $\mathcal{J}$) is (id, m), let $Q_i = id$. If $\mathcal{D}$ makes less than i inputs at interface A (because it returns a bit before or because $\mathcal{A}$ terminates the execution before), let Q_i be a uniform identity in $\mathcal{J}$ that has not been input together with a message at interface A (by our assumptions on $\mathcal{D}$, such an identity always exists). Note that $\mathcal{A}$ terminating prematurely is equivalent to the event $Q_{j'} \neq id^*$ because (id^*, m) is by assumption only input at interface A if id^* has not been input at interface C, and after the input (id^*, m), id^* is not input at interface C because this would be ignored by both resources. We thus have $\Pr[Q_{j'} = id^*] = \frac{1}{|\mathcal{J}|}$ since id^* is chosen uniformly and the view of $\mathcal{D}$ is independent of id^* as long as $\mathcal{A}$ does not terminate prematurely.

Note that given $Q_{j'} = id^*$, the view of $\mathcal{D}$ in this experiment is identical to its view in $\mathcal{D}\mathsf{H}_{j'}$ if $b = 0$ and its view in $\mathcal{D}\mathsf{H}_{j'-1}$ if $b = 1$. This yields

$$
\begin{aligned}
&\mathsf{Adv}^{\mathsf{IND\text{-}sID\text{-}CPA}}_{\mathcal{E},\mathcal{A}}\\
&= \Pr\big[\mathsf{Exp}^{\mathsf{IND\text{-}sID\text{-}CPA}}_{\mathcal{E},\mathcal{A}} = 1 \mid b = 1\big] + \Pr\big[\mathsf{Exp}^{\mathsf{IND\text{-}sID\text{-}CPA}}_{\mathcal{E},\mathcal{A}} = 1 \mid b = 0\big] - 1\\
&= \Pr\big[\mathsf{Exp}^{\mathsf{IND\text{-}sID\text{-}CPA}}_{\mathcal{E},\mathcal{A}} = 1 \mid b = 1\big] - \Pr\big[\mathsf{Exp}^{\mathsf{IND\text{-}sID\text{-}CPA}}_{\mathcal{E},\mathcal{A}} = 0 \mid b = 0\big]\\
&= \Pr[Q_{j'} \neq id^*] \cdot \frac{1}{2} + \Pr[Q_{j'} = id^*] \cdot \frac{1}{q}\sum_{i=1}^{q} \Pr[\mathcal{D}\mathsf{H}_{i-1} = 0]\\
&\qquad - \Pr[Q_{j'} \neq id^*] \cdot \frac{1}{2} + \Pr[Q_{j'} = id^*] \cdot \frac{1}{q}\sum_{i=1}^{q} \Pr[\mathcal{D}\mathsf{H}_{i} = 0]\\
&= \frac{\Pr[Q_{j'} = id^*]}{q} \cdot \left(\sum_{i=1}^{q} \Pr[\mathcal{D}\mathsf{H}_{i} = 1] - \sum_{i=1}^{q} \Pr[\mathcal{D}\mathsf{H}_{i-1} = 1]\right)\\
&= \frac{1}{q|\mathcal{J}|} \cdot \big(\Pr[\mathcal{D}\mathsf{H}_{q} = 1] - \Pr[\mathcal{D}\mathsf{H}_{0} = 1]\big)\\
&\geq \frac{1}{q\mu}\Delta^{\mathcal{D}}\Big(\mathsf{ibe}^{\mathsf{P}}_{\overline{\mathcal{P}}}\phi^{\mathsf{NW}^+}_{\overline{\mathcal{P}}}\mathsf{NW}^+, \sigma_{\mathcal{P}}\phi^{\mathsf{DCC}}_{\overline{\mathcal{P}}}\mathsf{preDCC}^{n,\mathcal{ID},\rho}\Big).
\end{aligned}
$$

Rearranging the inequality concludes the proof. □

Remark. The result from [BB04] that any IND-sID-CPA secure IBE scheme is also IND-ID-CPA secure with a loss of the factor $|\mathcal{ID}|$ in security can be seen as a corollary to Lemma 4.4.6: The resource $\mathsf{preDCC}^{n,\mathcal{ID},\rho}$ can be used in the same way as $\mathsf{stDCC}^{n,\mathcal{ID},\rho}$ when the full set $\mathcal{ID}$ is initially input at interface A. This comes at the cost of precisely a factor of $|\mathcal{ID}|$ in the distinguishing advantage. However, our result is more general because it makes explicit that even if $\mathcal{ID}$ is large, one can use a IND-sID-CPA secure IBE scheme in a scenario where messages are only sent for a smaller subset of $\mathcal{ID}$ but all identities in $\mathcal{ID}$ can be registered by users.

The following lemma implies that IND-sID-CPA security is also necessary for the construction. Its proof is omitted since it is exactly analogous to the proof of Lemma 4.4.3.

Lemma 4.4.7. *Let $\rho \in \mathbb{N}$ and $\mathcal{P} \subseteq \{B_1, \dots, B_n\}, \mathcal{P} \neq \emptyset$. Then, for all valid IND-sID-CPA adversaries $\mathcal{A}$ and for all efficient converters σ_{B_i} for $B_i \in \mathcal{P}$, there exists an efficient distinguisher $\mathcal{D}$ such that*

$$\mathsf{Adv}^{\mathsf{IND\text{-}sID\text{-}CPA}}_{\mathcal{E},\mathcal{A}} = 2 \cdot \Delta^{\mathcal{D}}\Big(\mathsf{ibe}^{\mathsf{p}}_{\overline{\mathcal{P}}} \phi^{\mathsf{NW}^+}_{\overline{\mathcal{P}}} \mathsf{NW}^+, \sigma_{\mathcal{P}} \phi^{\mathsf{DCC}}_{\overline{\mathcal{P}}} \mathsf{preDCC}^{n,\mathcal{ID},\rho}\Big).$$

Lemmata 4.4.6 and 4.4.7 together imply the following theorem:

Theorem 4.4.8. *Let ρ be an upper bound on the randomness used in one invocation of* Gen*,* Ext*, and* Enc*. We then have*

$$\mathsf{NW}^+_{\phi^{\mathsf{NW}^+}} \underset{\{B_1,\dots,B_n\}}{\overset{\mathsf{ibe}^{\mathsf{p}}}{\Longmapsto}} \mathsf{preDCC}^{n,\mathcal{ID},\rho}_{\phi^{\mathsf{DCC}}}$$

$\iff$ *the underlying IBE scheme is IND-sID-CPA secure.*

As in Section 4.4.2, we can prove the following theorem very similarly.

Theorem 4.4.9. *Let ρ be an upper bound on the randomness used in one invocation of* Gen*,* Ext *and* Enc*. We then have*

$$\mathsf{NW}^+_{\phi^{\mathsf{NW}^+}} \underset{\{B_1,\dots,B_n\}}{\overset{\mathsf{ibe}^{\mathsf{2p}}}{\Longmapsto}} \mathsf{pre2DCC}^{n,\mathcal{ID},\rho}_{\phi^{\mathsf{DCC}}}$$

$\iff$ *the underlying IBE scheme is IND-sID1-CPA secure.*

4.5 Construction with Random Oracles

4.5.1 Random Oracles

We next show how any IND-ID-CPA secure IBE scheme $\mathcal{E} = (\mathsf{Gen}, \mathsf{Ext}, \mathsf{Enc}, \mathsf{Dec})$ can be used to construct DCC from the resource $\mathsf{NW}^{\mathsf{RO}}$, which corresponds to our network together with a random oracle. A random oracle is a uniform random function $\{0,1\}^* \to \{0,1\}^k$ for some k to which all parties have access. The heuristic to model a hash function as a random oracle was proposed by Bellare and Rogaway [BR93]. Theorem 4.4.1 implies that no hash function can be used to instantiate the random oracle in this construction. However, if a random oracle is actually available, e.g., via a trusted party or secure hardware, the overall construction is sound. For our purpose, it is sufficient to consider random oracles with binary codomain.

Definition 4.5.1. The resource RO has interfaces A, C, and $B_1, \dots, B_n$. On input $x \in \{0,1\}^*$ at interface $I \in \{A, C, B_1, \dots, B_n\}$, if x has not been input before (at any interface), RO chooses $y \in \{0,1\}$ uniformly at random and outputs y at interface I; if x has been input before and the resulting output was y, RO outputs y at interface I.

Programmability. For our construction, we will assume that a random oracle is available as part of the real resource. Our protocol then constructs an ideal resource that does not give the honest parties access to the random oracle. Thus, the simulators in the ideal world can answer queries to the random oracle arbitrarily as long as they are consistent with previous answers and are indistinguishable from uniform bits. This gives the simulators additional power which allows us to overcome the impossibility result from Theorem 4.4.1. Since the simulators can in some sense "reprogram" the random oracle, we are in a scenario that is often referred to as *programmable random oracle model.*

4.5.2 Construction of Delivery Controlled Channels

Our protocol $\mathsf{ibe}^{\mathsf{ro}}$ uses the same idea as Nielsen's scheme [Nie02] and essentially corresponds to the transformation from [BSW11, Section 5.3] applied to an IBE scheme. At a high level, it works as follows: To send a message m for identity id, choose a bit string $r \twoheadleftarrow \{0,1\}^\kappa$ uniformly at random, input $(r, 1), \dots, (r, |m|)$ to the random oracle to obtain a uniform value r' with $|r'| = |m|$. Finally encrypt r with the IBE scheme for identity id and send the resulting ciphertext together with $m \oplus r'$. The security proof exploits that the one-time pad is non-committing and the random oracle is programmable. A detailed description of the protocol and the involved resources follows.

Real resource. The real resource in our construction consists of NW and RO. We thus define

$$\mathsf{NW}^{\mathsf{RO}} := \left[\mathsf{BCast}^{A,\{B_1,\dots,B_n\}}, \mathsf{AutC}^{1,C,A}, \mathsf{SecC}^{C,B_1}, \dots, \mathsf{SecC}^{C,B_n}, \mathsf{RO}\right]$$

and $\phi^{\mathsf{NW}^{\mathsf{RO}}} := (\phi_A^{\mathsf{NW}^{\mathsf{RO}}}, \phi_C^{\mathsf{NW}^{\mathsf{RO}}}, \phi_{B_1}^{\mathsf{NW}^{\mathsf{RO}}}, \dots, \phi_{B_n}^{\mathsf{NW}^{\mathsf{RO}}})$ where for $I \in \{A, C, B_1, \dots, B_n\}$, $\phi_I^{\mathsf{NW}^{\mathsf{RO}}} := [\mathbf{1}, \phi_I^{\mathsf{AutC}}, \phi_I^{\mathsf{SecC}}, \dots, \phi_I^{\mathsf{SecC}}, \mathbf{1}]$.

Protocol. For an IBE scheme $\mathcal{E}$, we define protocol converters $\mathsf{enc}^{\mathsf{ro}}$, $\mathsf{dec}^{\mathsf{ro}}$, and $\mathsf{reg}^{\mathsf{ro}}$ as follows and let $\mathsf{ibe}^{\mathsf{ro}} := (\mathsf{enc}^{\mathsf{ro}}, \mathsf{reg}^{\mathsf{ro}}, \mathsf{dec}^{\mathsf{ro}}, \dots, \mathsf{dec}^{\mathsf{ro}})$: For $r \in \{0,1\}^*$ and $\ell \in \mathbb{N}$, we write $r' \leftarrow H(r, \ell)$ as an abbreviation for: Output $(r, 1), \dots, (r, \ell)$ at the inside sub-interface to RO, let $r'_1, \dots, r'_\ell$ be the answers from the random oracle, and let $r' := r'_1 \dots r'_\ell$.

The converter $\mathsf{enc}^{\mathsf{ro}}$ first expects to receive a master public key mpk at its inside interface and stores it internally. On input a message and identity $(id, m) \in \mathcal{ID} \times \mathcal{M}$ at its outside interface, it chooses $r \twoheadleftarrow \{0,1\}^\kappa$ uniformly at random and computes $c^{\mathsf{IBE}} \leftarrow \mathsf{Enc}(mpk, id, r)$ and $r' \leftarrow H(r, |m|)$. The converter $\mathsf{enc}^{\mathsf{ro}}$ then sets $c^{\mathsf{OTP}} \leftarrow m \oplus r'$ and outputs $(id, c^{\mathsf{IBE}}, c^{\mathsf{OTP}})$ at its inside sub-interface to $\mathsf{BCast}^{A,\{B_1,\dots,B_n\}}$.

The converter $\mathsf{dec}^{\mathsf{ro}}$ on input an identity and a corresponding user secret key (id, usk_{id}) at its inside interface, stores this tuple internally and outputs id at its outside interface. For all pairs $(id_j, c_j^{\mathsf{IBE}}, c_j^{\mathsf{OTP}})$ with $id_j = id$ stored internally, $\mathsf{dec}^{\mathsf{ro}}$ computes $r_j \leftarrow \mathsf{Dec}(usk_{id}, id, c_j^{\mathsf{IBE}})$ and $r' \leftarrow H(r, |c_j^{\mathsf{OTP}}|)$, and outputs $(id, c_j^{\mathsf{OTP}} \oplus r')$ at its outside interface. On input $(id, c^{\mathsf{IBE}}, c^{\mathsf{OTP}})$ at its inside interface, $\mathsf{dec}^{\mathsf{ro}}$ computes $r \leftarrow \mathsf{Dec}(usk_{id}, id, c^{\mathsf{IBE}})$ and $r' \leftarrow H(r, |c^{\mathsf{OTP}}|)$, and outputs $(id, c^{\mathsf{OTP}} \oplus r')$ at its outside interface if it has stored a user secret key for the identity id, and stores $(id, c^{\mathsf{IBE}}, c^{\mathsf{OTP}})$ internally otherwise.

The converter $\mathsf{reg}^{\mathsf{ro}}$ is identical to the converter reg: It initially computes $(mpk, msk) \leftarrow \mathsf{Gen}(1^\kappa)$, stores msk internally, and outputs mpk at its inside sub-interface to $\mathsf{AutC}_{\phi^{\mathsf{AutC}}}^{1,C,A}$. On input (id, i) at its outside interface, the converter $\mathsf{reg}^{\mathsf{ro}}$ computes $usk_{id} \leftarrow \mathsf{Ext}(msk, id)$ and outputs (id, usk_{id}) at its inside sub-interface to $\mathsf{SecC}_{\phi^{\mathsf{SecC}}}^{C,B_i}$.

Ideal resource and construction. As explained in Section 4.5.1, honest parties do not have access to the random oracle in the ideal world. Therefore, we define $\phi^{\mathsf{RO}} := \{\bot, \dots, \bot\}$ to block access to RO in the ideal world. The ideal resource in our construction then corresponds to $\left[\mathsf{DCC}_{\phi^{\mathsf{DCC}}}^{n,\mathcal{ID},\rho+\kappa}, \mathsf{RO}_{\phi^{\mathsf{RO}}}\right]$.

Theorem 4.5.2. *Let ρ be an upper bound on the randomness used in one invocation of* Gen*,* Ext *and* Enc*. If $\mathcal{E}$ is IND-ID-CPA secure, we have*

$$\mathsf{NW}^{\mathsf{RO}}_{\phi^{\mathsf{NW}^{\mathsf{RO}}}} \quad \underset{\{B_1,\dots,B_n\}}{\overset{\mathsf{ibe}^{\mathsf{ro}}}{\Longmapsto}} \quad \left[\mathsf{DCC}^{n,\mathcal{ID},\rho+\kappa}_{\phi^{\mathsf{DCC}}}, \mathsf{RO}_{\phi^{\mathsf{RO}}}\right].$$

Proof sketch. For $i \in \{1,\dots,n\}$ the simulator σ_{B_i} maintains an initially empty list R and remembers all its inputs and outputs. It reacts to inputs as described in Figure 4.6. Let $\mathcal{P} \subseteq \{B_1,\dots,B_n\}$ and let $\mathcal{D}$ be an efficient distinguisher for the resources $\mathsf{ibe}^{\mathsf{ro}}_{\overline{\mathcal{P}}}\phi^{\mathsf{NW}^{\mathsf{RO}}}_{\overline{\mathcal{P}}}\mathsf{NW}^{\mathsf{RO}}$ and $\sigma_{\mathcal{P}}\left[\phi^{\mathsf{DCC}}_{\overline{\mathcal{P}}}\mathsf{DCC}^{n,\mathcal{ID},\rho+\kappa}, \phi^{\mathsf{RO}}_{\overline{\mathcal{P}}}\mathsf{RO}\right]$. Note that since all σ_{B_i} initially input `share` to $\mathsf{DCC}^{n,\mathcal{ID},\rho+\kappa}$, they all receive the same outputs from that resource. Thus, they all maintain the same list R.

Let E be the event that some simulator aborts and let F be the event that there exists some $id \in \mathcal{ID}$ such that $\mathcal{D}$ inputs a random oracle query x before receiving a key for id and some simulator has output $(id, \mathsf{Enc}(mpk, id, x), r''; r)$ for some r and r'' before. Note that as long as neither E nor F occur, the resources $\mathsf{ibe}^{\mathsf{ro}}_{\overline{\mathcal{P}}}\phi^{\mathsf{NW}^{\mathsf{RO}}}_{\overline{\mathcal{P}}}\mathsf{NW}^{\mathsf{RO}}$ and $\sigma_{\mathcal{P}}\left[\phi^{\mathsf{DCC}}_{\overline{\mathcal{P}}}\mathsf{DCC}^{n,\mathcal{ID},\rho+\kappa}, \phi^{\mathsf{RO}}_{\overline{\mathcal{P}}}\mathsf{RO}\right]$ behave identically since all keys are generated equally by both resources and for all outputs $(id, c^{\mathsf{IBE}}, c^{\mathsf{OTP}})$ after input (id, m), c^{IBE} is an encryption of a uniform bit string r' for id and the j-th bit of c^{OTP} is the XOR of the j-th bit of m and the answer of the random when queried on (r', j). Event E occurs only if the resource outputs some $r' \in \{0,1\}^{\kappa}$ that collides with a previously used value, which is the case with negligible probability. Event F also has negligible probability by the IND-ID-CPA security of the IBE scheme, which can be shown by a reduction similar to the one in the proof of Lemma 4.4.2. $\square$

Converter σ_{B_i}

```
Inside interface
Input: $(r|r') \in \{0,1\}^{\rho+\kappa}$
  $(mpk, msk) \leftarrow \mathsf{Gen}(1^\kappa; r)$
  output share at inside sub-interface to $\mathsf{DCC}^{n,\mathcal{ID},\rho+\kappa}$
  output $mpk$ at outside sub-interface simulating $\mathsf{AutC}^{1,C,A}$

Input: $(id, m, r|r') \in \mathcal{ID} \times \mathcal{M} \times \{0,1\}^{\rho+\kappa}$
  $r'' \leftarrow H(r', |m|)$
  if $R$ contains $((r', j), y)$ for some $j \in \{1, \dots, |m|\}$, and $y \neq r''_j$ then
      abort
  else
      add $((r', j), r''_j)$ to $R$ for $j \in \{1, \dots, |m|\}$
      $c^{\mathsf{IBE}} \leftarrow \mathsf{Enc}(mpk, id, r'; r)$
      $c^{\mathsf{OTP}} \leftarrow m \oplus r''$
      output $(id, c^{\mathsf{IBE}}, c^{\mathsf{OTP}})$ at outside sub-interface simulating $\mathsf{BCast}^{A,\{B_1,\dots,B_n\}}$

Input: $(id, |m|, r|r') \in \mathcal{ID} \times \mathbb{N} \times \{0,1\}^{\rho+\kappa}$
  $r'' \leftarrow H(r', |m|)$
  $c^{\mathsf{IBE}} \leftarrow \mathsf{Enc}(mpk, id, r'; r)$
  output $(id, c^{\mathsf{IBE}}, r'')$ at outside sub-interface simulating $\mathsf{BCast}^{A,\{B_1,\dots,B_n\}}$

Input: $((id, k, r|r'), m_1, \dots, m_l) \in \left(\mathcal{ID} \times \{1, \dots, n\} \times \{0,1\}^{\rho+\kappa}\right) \times \mathcal{M}^l$
  for $j \in \{1, \dots, l\}$ do
      $(id, c^{\mathsf{IBE}}, c^{\mathsf{OTP}}) \leftarrow$ output for $j$-th input of form $(id, m, \tilde{r}|\tilde{r}')$ or $(id, |m|, \tilde{r}|\tilde{r}')$
      $r'' \leftarrow m_j \oplus c^{\mathsf{OTP}}$
      if $R$ contains $((\tilde{r}', j'), y)$ for some $j' \in \{1, \dots, |m_j|\}$, and $y \neq r''_{j'}$ then
          abort
      else
          add $((\tilde{r}', j'), r''_{j'})$ to $R$ for $j' \in \{1, \dots, |m|\}$
  $usk \leftarrow \mathsf{Ext}(msk, id; r)$
  if $k = i$ then
      output $(id, usk)$ at outside sub-interface simulating $\mathsf{SecC}^{C,B_i}$
  else
      output $|(id, usk)|$ at outside sub-interface simulating $\mathsf{SecC}^{C,B_k}$

Outside interface
Input: $x \in \{0,1\}^*$
  if $R$ contains $(x, y)$ for some $y \in \{0,1\}$ then
      output $y$ at outside sub-interface simulating $\mathsf{RO}$
  else
      output $x$ at inside sub-interface to $\mathsf{RO}$ and let $y$ be the answer
      output $y$ at outside sub-interface simulating $\mathsf{RO}$
```

Figure 4.6: Description of the simulator σ_{B_i}. Other inputs are ignored.

Chapter 5

Functional Encryption

5.1 Introduction

5.1.1 Motivation

Functional encryption (FE) is a very general concept formally introduced by Boneh, Sahai, and Waters [BSW11]. Many types of encryption such as public-key encryption, identity-based encryption [Sha85; MY91; BF01], and attribute-based encryption [SW05] can be seen as a special case of FE. Briefly, an FE scheme is parametrized by a value space and a set of functions on the value space. A trusted authority holding a master secret key corresponding to the master public key can generate secret keys for all functions in this set. Given the master public key one can encrypt messages, and given a secret key for a particular function f and an encryption of a value x, one can efficiently compute $f(x)$ but does not learn anything more about x. Moreover, even if one pools the secret keys for many functions $f_1, \dots f_k$, one can compute nothing about an encrypted value x except for exactly these function values, i.e., $f_1(x), \dots, f_k(x)$.

Formalizing these intuitive security requirements has caused more trouble than one might expect, and several security definitions for functional encryption exist in the literature. While some of them were shown to be too weak since schemes that should not be considered secure could be proven to satisfy them, others are so strong that even for very simple sets of functions, no scheme exists that satisfies them in the plain model

[BSW11; ONe10; AGVW13; BO13; DIJ+13]. The question of which definition is suitable for being used in a certain application therefore seems particularly relevant for functional encryption.

In an ideal-world view, however, it is rather straightforward to state what one expects from FE: If a publicly readable data repository (e.g., a public web repository) is available, then, by virtue of encrypting the data items before putting them into the repository, one constructs a repository where the data is now secret and where a specific entity has the capability of assigning to entities the right to access certain functions (but not more) of the data items stored in the repository. Obviously, granting the access right to an entity in this ideal world view corresponds in the real world to providing the decryption key of the corresponding function, and accessing the data corresponds to decrypting the corresponding ciphertext.

Compared to FE, standard public-key encryption achieves the same goal of providing access control for a repository, but access would be all-or-nothing: If one knows the secret key, one can decrypt, otherwise one cannot. A more versatile access control mechanism is obtained by identity-based encryption, which allows the trusted party to grant users the right to access data for a specific identity, where the input data contains the data itself and the identity that should be able to access it. More advanced access policies can be implemented using attribute-based encryption. The high flexibility of FE is demonstrated by the following application proposed in [BSW12; GKP+13]: Assume some user receives encrypted emails that are stored on the provider's server and mails with a high probability of being spam should not be downloaded, to minimize traffic. The trusted party, which in this case can coincide with the recipient of the mails, generates a special secret key for the provider that only allows to compute the score function of the spam filter. The provider now cannot read the contents of the mails but is still able to filter out spam and notify the recipient only about the remaining incoming messages. The repository here corresponds to the mail server and inputting data into the repository to sending the recipient encrypted emails.

Problem statement. We address the following questions: Does the above-described intuition of a secure repository really hold? Does one of the existing security definitions imply that such a view is valid, and if so, which one? What, then, do the other definitions achieve?

5.1.2 Contributions

We show that the exact characterization of FE as the construction of a certain access-controlled repository from a public repository and certain channels (to transmit secret keys and public keys) is indeed formally correct. This means, in particular, that one can compose constructions, according to the composition theorem of constructive cryptography. Concretely, if one has designed an application by defining it "on top" of a certain (assumed) repository with access control, then this application remains secure if the repository is implemented using a public repository, where data is encrypted using an FE scheme satisfying the appropriate definition.

However, for this to be true, none of the existing security definitions seem to suffice. Therefore we propose a new conventional security definition for FE, called CFE security, derived from the constructive viewpoint, which corresponds to an adequately modified version of an established game-based definition by Boneh, Sahai, and Waters [BSW11, Definition 4]. This suggests that CFE security is the appropriate definition if strong guarantees are required. We show that, as the definition in [BSW11], CFE security is impossible to achieve in the standard model but achievable in the random oracle model. In doing so, we also exemplify how results in the (programmable) random oracle model can be translated to a construction in CC.

The adequacy of CFE security might seem surprising since it is similar to [BSW11, Definition 4] and an example from [BF13] was supposed to show that this definition is insufficient. We recall that example and explain why the criticism is invalid. This demonstrates that traditional security definitions for FE are not well understood, which leads to misconceptions that do not arise in the constructive approach.

Moreover, we show that a weaker security definition by Gorbunov, Vaikuntanathan, and Wee [GVW12, Definition 1], which *is* achievable in the standard model, is sufficient for constructing a repository that restricts the number and order of interactions, making explicit how such schemes can (and cannot) meaningfully be used.

Finally, we show how adequate security definitions for generalizations of FE (multi-input, randomized functions, malicious ciphertext generation, etc.) can be obtained by straightforward extensions of the repository. For a constructive security definition (requiring that a certain repository be constructed), one can extract the corresponding conventional security definition and compare it to existing generalized definitions. We conjecture that the latter do generally not correspond to a meaningful construction of a repository, but carrying out the complete analysis for all definitions is beyond the scope of this thesis.

Overall, this leads to a unified treatment of many FE variants and makes explicit how they can be composed within a higher-level protocol.

5.1.3 Related Work and Relation to IBE

A paper by Sadikin et al. [SPPM13] provides an ideal functionality for functional encryption in UC, but only for a special class of functional encryption schemes, namely attribute-based encryption. Thus, their results are not suitable to understand definitions for general functional encryption, as we do in this chapter.

Since identity-based encryption, treated in Chapter 4, is known to be a special case of functional encryption [BSW11], one could expect the results in this chapter to be very similar. And indeed, the impossibility results and the constructions in the random oracle model are closely related. There are, however, two important differences: Firstly, the ideal resource considered for IBE in Chapter 4 involves many potential receivers with different identities, and the intended recipients directly receive sent messages. For functional encryption, on the other hand, we model a repository that allows to access certain information about the stored data. While this could also be modeled with many parties, we only consider one interface for inputting data, one interface for granting access rights, and one interface for accessing data. This simplifies the presentation and allows us to focus more on the relevant aspects of functional encryption. Secondly, the security definitions considered in Chapter 4 are specific to IBE, while we look at more involved simulation-based security definitions for general functional encryption in this chapter.

5.2 Definition of Functional Encryption

A functional encryption scheme is a generalized public-key encryption scheme defined for a set F of functions with common domain X. Given the public key, one can encrypt data $x \in X$ and given a secret key for a function $f \in F$, one can efficiently compute $f(x)$ from an encryption of x. The secret keys for all $f \in F$ can be generated using a so-called master secret key which is generated together with the public key. To capture which information ciphertexts leak about the encrypted data, a special leakage function $f_0 \in F$ is considered. An intuitive security requirement guarantees that given a ciphertext for some x and secret keys for $f_1, \dots, f_n$, one should not be able to learn more about x than what can be learned from $f_0(x), \dots, f_n(x)$. We here only define the syntax and correctness condition of a functional encryption scheme and refer to later sections for formal security definitions. The definition here only covers unary and deterministic functions. See Section 5.8 for a discussion of more general notions of functional encryption.

Definition 5.2.1. Let X be a nonempty set and F be a set of functions with domain X such that F contains a distinguished leakage function f_0. A *functional encryption scheme* for F consists of the following efficient probabilistic algorithms:

Setup: The algorithm setup on input a security parameter 1^κ generates a *public key pk* and a *master secret key mk*.

Key generation: Given a master secret key mk and a function $f \in F$, the algorithm keygen generates a *secret key* sk_f for this f, where sk_{f_0} equals the empty string.

Encryption: Given a public key pk and some value $x \in X$, the algorithm enc computes a *ciphertext c*.

Decryption: The algorithm dec on input a secret key sk_f and a ciphertext c, outputs some value x.

For correctness, we require for all $x \in X$, and for all $f \in F$ that $\mathsf{dec}\big(sk_f, \mathsf{enc}(pk, x)\big) = f(x)$ with probability 1, for $(pk, mk) \leftarrow \mathsf{setup}(1^\kappa)$ and $sk_f \leftarrow \mathsf{keygen}(mk, f)$.

Remark. Following [BSW11], we assume everyone can always evaluate the function f_0. This can be seen as a rather artificial requirement; if $f_0(x)$, e.g., reveals the bit length $|x|$ of x, there has to be an efficient algorithm that precisely computes $|x|$ from a ciphertext. A more natural approach would not guarantee all parties to compute f_0, but rather not exclude in the security definition that dishonest parties can do so. To formalize that something is not guaranteed but potentially possible, one can use filtered resources in constructive cryptography. While all results in this chapter extend to such a definition, we stick to the definition above to simplify the presentation.

5.3 Repositories and Access Control

5.3.1 Repository Resources

In this section, we introduce a repository resource that allows users to input and access data and that naturally captures how a repository works. We first define a repository with access control and then specify a public repository without access control as a special case thereof. Users can input data from a *data set* X into the repository. After inputting data, the resource returns a *handle* (e.g., a URL or a memory address) from a set H via which the data can be accessed later. This handle could be chosen by the resource, by the user who inputs data, or by both in an interactive protocol. Since the particular procedure to generate handles is irrelevant for our purposes, we will refer to a method `getHandle` that returns an element of H without describing its implementation. We only assume that the returned handles are distinct, that is, no data is overwritten.

Motivated by the syntax of functional encryption, we consider a set F of *access functions* containing functions with domain X and allow users to retrieve such functions of input data. Which functions a user can access depends on the rights of this user, i.e., for each $f \in F$ users can have the right to obtain $f(x)$ for previously input $x \in X$. Everyone has the right to obtain $f_0(x)$ for a special function f_0 and a trusted authority can grant users additional rights.

We consider a resource with an interface for Alice who can input data, an interface for Bob who can access data, and an interface for the trusted party Charlie who can grant rights to Bob. Alice and Bob are not necessarily single users but correspond to roles users can have. All results in this chapter regard Bob as the only potentially dishonest party. In case he is dishonest, one can think of him as a group of colluding dishonest parties who try to combine their rights to get access to a function of some data none of them alone could access. Hence, one dishonest party is

Resource Rep_F

```
Initialization
  $R \leftarrow \{f_0\}$
  for $h \in H$ do
      $M[h] \leftarrow \bot$

Interface $A$
Input: $x \in X$
  $h \leftarrow$ getHandle
  $M[h] \leftarrow x$
  output $h$ at interface $A$

Interface $B$
Input: $(f, h) \in F \times H$
  if $f \in R \wedge M[h] \neq \bot$ then
      output $f(M[h])$ at interface $B$

Interface $C$
Input: $f \in F$
  $R \leftarrow R \cup \{f\}$
  output $f$ at interface $B$
```

Figure 5.1: Definition of the resource Rep_F. All inputs not matching the given format are ignored.

sufficient to cover collusion resistance. Similarly, the resource can be used in a context with multiple honest parties inputting data.

Definition 5.3.1. Let X be a nonempty set and F a set of functions with domain X and $f_0 \in F$. The resource Rep_F has the interfaces A, B, and C. It internally manages the set R of functions Bob is allowed to access, and a map M assigning to a handle $h \in H$ the value $M[h] \in X \cup \{\bot\}$ where $\bot \notin X$ is a special symbol. Initially, $R = \{f_0\}$ and $M[h] = \bot$ for all $h \in H$. The resource works as described in Figure 5.1.[1]

Remark. Note that Bob needs to know the handles to access data. Hence, when this resource is used in a larger protocol, Alice needs to send Bob the handles over an additional channel. This is in fact very natural: If Alice uploads a document to her web server and wants Bob to download it, she needs to tell him the URL, e.g., via email. See Section 5.9 for an example application of the resource that demonstrates how other protocols can use the constructed repository and how to apply the composition theorem of the constructive cryptography framework.

We now define a public repository without access control, which will serve as an assumed resource in our constructions. It corresponds to a

[1]We define all resources to ignore invalid inputs. Alternatively, the resources could return error messages. While this alternative might be closer to the behavior of real systems, we decided to simply ignore invalid inputs because they are not relevant for our results.

repository as defined above where f_0 is the identity function on X, i.e., everyone is allowed to access the (identity function of) stored data.

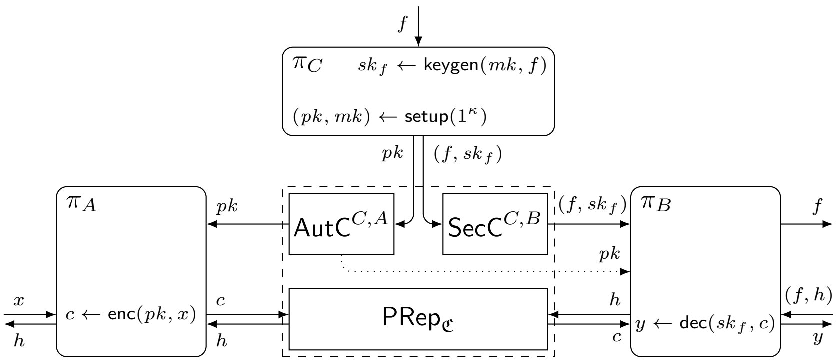
Definition 5.3.2. Let X be a nonempty set, $f_0 \coloneqq \mathrm{id}_X \colon X \to X, x \mapsto x$, and $P \coloneqq \{f_0\}$. We define the *public repository for X* as $\mathsf{PRep}_X \coloneqq \mathsf{Rep}_P$. For inputs at Bob's interface, we will write h instead of (id_X, h) to simplify notation.

5.3.2 Access Control via Functional Encryption

A versatile repository supports a large class of access functions and restricts Bob's initial rights as much as possible. In this section, we describe how to use functional encryption to construct such a repository from a public repository. More precisely, let $\mathcal{E} = (\mathsf{setup}, \mathsf{keygen}, \mathsf{enc}, \mathsf{dec})$ be a functional encryption scheme for a set F of functions with domain X and let $\mathfrak{C}$ be the range of enc. Our goal is to construct Rep_F from $\mathsf{PRep}_{\mathfrak{C}}$. To distribute keys, we additionally need an authenticated channel $\mathsf{AutC}^{C,A}$ from Charlie to Alice and a secure channel $\mathsf{SecC}^{C,B}$ from Charlie to Bob,[2] i.e., the assumed resource in our construction corresponds to $\left[\mathsf{PRep}_{\mathfrak{C}}, \mathsf{AutC}^{C,A}, \mathsf{SecC}^{C,B}\right]$.

To achieve this construction, we define the protocol $\pi = (\pi_A, \pi_B, \pi_C)$ for the functional encryption scheme $\mathcal{E}$ as follows: At the beginning, π_C invokes $(pk, mk) \leftarrow \mathsf{setup}(1^\kappa)$, stores mk and sends pk to Alice over the authenticated channel. This public key is internally stored by π_A. On input $x \in X$ at its outside interface, π_A outputs $c \leftarrow \mathsf{enc}(pk, x)$ at its inside interface to the repository and outputs the returned handle h at its outside interface. On input $f \in F$ at its outside interface, π_C computes $sk_f \leftarrow \mathsf{keygen}(mk, f)$ and sends (f, sk_f) to B over the secure channel. The corresponding secret key is stored by π_B and f is output at its outside interface. On input $(f, h) \in F \times H$ at its outside interface, π_B outputs h at its inside interface to the repository if it has stored a secret key sk_f for this function f or if $f = f_0$. If it receives a ciphertext c from the repository, it outputs $y \leftarrow \mathsf{dec}(sk_f, c)$ at its outside interface. All other inputs are ignored. See Figure 5.2 for an illustration of the protocol.

[2]For both channels, a dishonest Bob assumes the role of an eavesdropper. That is, he can learn the public key, which is sent over the authenticated channel from Charlie to Alice. If the resource is used in a context with many Bobs, it is important that the channel from Charlie to each of them is secure to prevent dishonest users from eavesdropping on the secret keys.



f
π_C $sk_f \leftarrow \mathsf{keygen}(mk, f)$
$(pk, mk) \leftarrow \mathsf{setup}(1^\kappa)$
pk
(f, sk_f)
π_A
pk
$\mathsf{AutC}^{C,A}$
$\mathsf{SecC}^{C,B}$
(f, sk_f)
π_B
f
pk
x
$c \leftarrow \mathsf{enc}(pk, x)$
c
$\mathsf{PRep}_{\mathfrak{C}}$
h
$y \leftarrow \mathsf{dec}(sk_f, c)$
(f, h)
h
h
c
y

Figure 5.2: Overview of the protocol (π_A, π_B, π_C). The dashed rectangle represents the assumed resource, corresponding to the parallel composition of $\mathsf{AutC}^{C,A}$, $\mathsf{SecC}^{C,B}$, and $\mathsf{PRep}_{\mathfrak{C}}$, the dotted line depicts that a dishonest Bob can learn the public key while the protocol does not use it.

The following lemma states that this protocol constructs the desired resource if all parties are honest. It follows directly from the correctness of the functional encryption scheme.

Lemma 5.3.3. *For the protocol $\pi = (\pi_A, \pi_B, \pi_C)$ defined above, we have*

$$\pi_A \pi_B \pi_C \big[\mathsf{PRep}_{\mathfrak{C}}, \mathsf{AutC}^{C,A}, \mathsf{SecC}^{C,B}\big] \;\approx\; \mathsf{Rep}_F.$$

5.4 Security of Functional Encryption

5.4.1 Definition of CFE Security

The protocol described in the previous section constructs the desired resource with a dishonest Bob only if the underlying functional encryption scheme satisfies a suitable security definition. We propose such a definition, based on [BSW11, Definition 4], and refer to it as *composable functional encryption security*[3] (*CFE security* for short). We extend the

[3]This name is justified by Theorem 5.4.4, which together with the composition theorem of the constructive cryptography framework implies that CFE-secure schemes indeed guarantee composability.

Experiment $\mathsf{Exp}^{\mathsf{CFE\text{-}real}}_{\mathcal{E},\mathcal{A}}$

```
Input: $1^\kappa, \kappa \in \mathbb{N}$
  $(pk, mk) \leftarrow \mathsf{setup}(1^\kappa)$
  $(l, \tau) \leftarrow (0, 0)$
  repeat
    $l \leftarrow l + 1$
    $x_l \leftarrow \mathcal{A}_1^{\mathsf{keygen}(mk,\cdot)}(pk)[[\tau]]$
    $c_l \leftarrow \mathsf{enc}(pk, x_l)$
    $t \leftarrow \mathcal{A}_2(c_l)[[\tau]]$
  until $t = \mathtt{true}$
  return $\tau$
```

Experiment $\mathsf{Exp}^{\mathsf{CFE\text{-}ideal}}_{\mathcal{E},S,\mathcal{A}}$

```
Input: $1^\kappa, \kappa \in \mathbb{N}$
  $(pk, s) \leftarrow S_1(1^\kappa)$
  $(l, \tau) \leftarrow (0, 0)$
  repeat
    $l \leftarrow l + 1$
    $x_l \leftarrow \mathcal{A}_1^{\mathcal{O}(\cdot, x_1, \ldots, x_{l-1})[[s]]}(pk)[[\tau]]$
    $(f_1, \ldots, f_q) \leftarrow$ queries by $\mathcal{A}_1$
    $c_l \leftarrow S_3(f_0(x_l), \ldots, f_q(x_l))[[s]]$
    $t \leftarrow \mathcal{A}_2(c_l)[[\tau]]$
  until $t = \mathtt{true}$
  return $\tau$
```

Figure 5.3: Experiments for the CFE security definition, for a scheme $\mathcal{E}$, an adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, and a simulator $S = (S_1, S_2, S_3)$. The oracle $\mathcal{O}$ is defined as $\mathcal{O}(f, x_1, \ldots, x_{l-1})[[s]] :=S_2(f, f(x_1), \ldots, f(x_{l-1}))[[s]]$.

definition from [BSW11] to adaptive adversaries that can choose messages depending on ciphertexts for previous messages. This extension was already mentioned in that paper but not formalized. Our definition additionally restricts oracle access of the involved algorithms. These changes are discussed after the definition. We note that SS1 security defined in [BO13] also corresponds to an adaptive variant of [BSW11, Definition 4] but also differs in other aspects and is not equivalent to the definition we propose here. In particular, SS1 security includes some auxiliary inputs that the authors claim to eliminate a weakness described in [BF13]. As we explain in Section 5.4.3, however, the effect pointed out in [BF13] is in fact not a weakness, so there is no need for a fix.

We follow the notation from [BSW11], i.e., for algorithms A and B, $A^{B(\cdot)}(x)$ denotes that A gets x as input and has oracle access to B, that is, $B(q)$ is answered to A in response to an oracle query q. Moreover, $A(\cdot)[[s]]$ means that A gets s as an additional input and can update its value. More precisely, $A(x)[[s]]$ corresponds to the algorithm that invokes $(y, s) \leftarrow A(x, s)$ and returns y.

Definition 5.4.1. Let $\mathcal{E} = (\mathsf{setup}, \mathsf{keygen}, \mathsf{enc}, \mathsf{dec})$ be a functional encryption scheme for a set F of functions with domain X. We introduce the experiments in Figure 5.3 for an efficient probabilistic oracle algorithm $\mathcal{A}_1$

and efficient probabilistic algorithms $\mathcal{A}_2$, S_1, S_2, and S_3. The advantage of a distinguisher $\mathcal{D}$ in distinguishing the outputs of these experiments is denoted by

$$\mathsf{Adv}^{\mathsf{CFE}}_{\mathcal{E},S,\mathcal{A},\mathcal{D}} := \Delta^{\mathcal{D}}\Big(\mathsf{Exp}^{\mathsf{CFE\text{-}real}}_{\mathcal{E},\mathcal{A}}, \mathsf{Exp}^{\mathsf{CFE\text{-}ideal}}_{\mathcal{E},S,\mathcal{A}}\Big),$$

where $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ and $S = (S_1, S_2, S_3)$. We say the scheme $\mathcal{E}$ is *CFE secure* if there exist efficient S_1, S_2, and S_3 such that the distinguishing advantage $\mathsf{Adv}^{\mathsf{CFE}}_{\mathcal{E},S,\mathcal{A},\mathcal{D}}$ is negligible for all efficient $\mathcal{A}_1$, $\mathcal{A}_2$ and for all efficient distinguishers $\mathcal{D}$.

Note that the value τ in the experiments can be used to share the state between the algorithms $\mathcal{A}_1$ and $\mathcal{A}_2$, and the value s to share the state between S_1, S_2, and S_3. Intuitively, S_1 simulates the generation of the public key, S_2 generates simulated secret keys, and S_3 simulates ciphertexts for values x given only the images of x under the functions the adversary has already requested secret keys for. A scheme is considered secure if these simulated values are indistinguishable from the corresponding values generated in an execution of the actual protocol. The intuition is that in this case, a ciphertext does not leak anything about the encrypted value that an adversary cannot conclude from the function values he is supposed to learn.

As mentioned before, this definition is close to a fully adaptive version of the one given by Boneh et al. [BSW11]. One difference is that the algorithm $\mathcal{A}_2$ is not given oracle access to a key-generation oracle in our definition. This simplifies especially the ideal experiment and is not necessary in our case since $\mathcal{A}_2$ can instead store its query in τ and $\mathcal{A}_1$ can then query its oracle and continue the execution of $\mathcal{A}_2$ in the following iteration. Furthermore, S_3 in [BSW11] has oracle access to $\mathcal{A}_2$ and can therefore run $\mathcal{A}_2$ on several inputs and discard undesired outputs. Our definition is stronger because we do not allow this. It was already noted in [AGVW13] that this oracle access might be problematic. Note that, in contrast to [BSW11, Definition 4], it is sufficient for the experiments to return τ because $\mathcal{A}_2$ can encode all relevant information in it and S_3 cannot tamper with it.

In contrast to many other definitions (e.g., [ONe10; BF13; GVW12; GKP+13]), we do allow S_1 and S_2 to "fake" the public key and the secret keys, respectively. In contrast to what is claimed in [BF13], it turns out that this is not a problem (see Section 5.4.3 for more details). This shows that there are many degrees of freedom in defining security experiments for functional encryption and that the consequences of a particular choice are often unclear. On the other hand, the constructive approach we follow in this thesis makes explicit what a protocol satisfying the definitions achieves, by specifying the resource that is constructed.

5.4.2 Equivalence of CFE Security and Construction

The goal of this section is to prove that the protocol defined in Section 5.3.2 constructs the corresponding repository resource if and only if the underlying functional encryption scheme is CFE secure. This implies that CFE security is precisely the definition needed for our purpose. The following lemma shows that CFE security is sufficient for the construction.

Lemma 5.4.2. *Let S_1, S_2, and S_3 be efficient probabilistic algorithms. Then there exists an efficient converter σ_B such that for all efficient distinguishers $\mathcal{D}$ for $\pi_A \pi_C \big[\mathsf{PRep}_{\mathfrak{C}}, \mathsf{AutC}^{C,A}, \mathsf{SecC}^{C,B}\big]$ and $\sigma_B \mathsf{Rep}_F$, there is an efficient probabilistic oracle algorithm $\mathcal{A}_1$, an efficient probabilistic algorithm $\mathcal{A}_2$, and an efficient distinguisher $\mathcal{D}'$ for the CFE experiment such that*

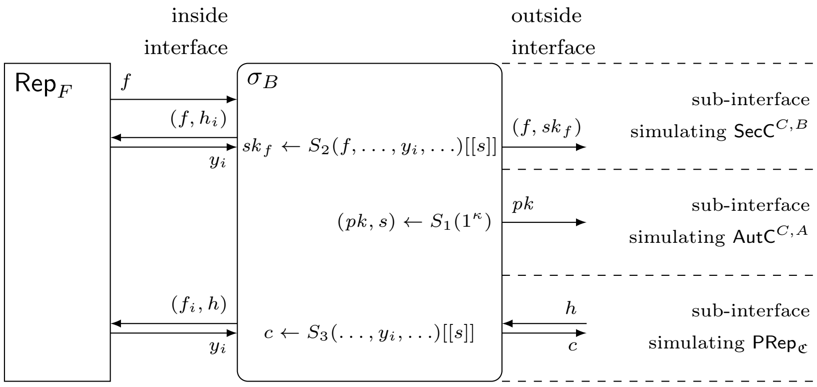
$$\Delta^{\mathcal{D}}\big(\pi_A \pi_C \big[\mathsf{PRep}_{\mathfrak{C}}, \mathsf{AutC}^{C,A}, \mathsf{SecC}^{C,B}\big], \sigma_B \mathsf{Rep}_F\big) = \mathsf{Adv}^{\mathsf{CFE}}_{\mathcal{E},S,\mathcal{A},\mathcal{D}'}.$$

Proof. We define the simulator σ_B in Figure 5.4. Let $\mathcal{D}$ be an efficient distinguisher for the resources $\pi_A \pi_C \big[\mathsf{PRep}_{\mathfrak{C}}, \mathsf{AutC}^{C,A}, \mathsf{SecC}^{C,B}\big]$ and $\sigma_B \mathsf{Rep}_F$. We can assume without loss of generality that $\mathcal{D}$ inputs $h \in H$ at interface B only if this h was output at interface A before, because other h will be ignored by both resources. We further assume that each $h \in H$ is input at most once, since both resources return the same value for each input of the same handle h.

We now describe the algorithms $\mathcal{A}_1$ and $\mathcal{A}_2$. When $\mathcal{A}_1$ is invoked with pk and $\tau = 0$, it starts a new simulation of the distinguisher $\mathcal{D}$, outputting pk at interface B from the authenticated channel. When $\mathcal{D}$ inputs $f \in F$ at interface C, $\mathcal{A}_1$ invokes its oracle with query f and outputs f and the answer to $\mathcal{D}$ at interface B from the secure channel. When $\mathcal{D}$ inputs $x \in X$ at interface A, $\mathcal{A}_1$ invokes `getHandle` and outputs the returned handle h at interface A. It further sets $M[h] \leftarrow x$ for a map M. When

Converter σ_B

```
Initialization
    $(l, q) \leftarrow (0, 0)$
    $(pk, s) \leftarrow S_1(1^\kappa)$
    output $pk$ at outside sub-interface simulating $\mathsf{AutC}^{C,A}$

Inside interface
Input: $f \in F$
    $q \leftarrow q + 1$
    $f_q \leftarrow f$
    for $i = 1, \ldots, l$ do
        output $(f, h_i)$ at inside interface, let $y_i$ be the returned value
    $sk_f \leftarrow S_2(f, y_1, \ldots, y_l)[[s]]$
    output $(f, sk_f)$ at outside sub-interface simulating $\mathsf{SecC}^{C,B}$

Outside interface
Input: $h \in H$
    if $\exists k \in \{1, \ldots, l\}$ $h_k = h$ then
        output $c_k$ at outside sub-interface simulating $\mathsf{PRep}_{\mathfrak{C}}$
    else if output $(f_0, h)$ at inside interface not ignored then        ▷ data stored for $h$
        $l \leftarrow l + 1$
        $h_l \leftarrow h$
        for $i = 0, \ldots, q$ do
            output $(f_i, h)$ at inside interface, let $y_i$ be the returned value
        $c_l \leftarrow S_3(y_0, \ldots, y_q)[[s]]$
        output $c_l$ at outside sub-interface simulating $\mathsf{PRep}_{\mathfrak{C}}$
```



inside interface
outside interface
Rep_F
σ_B
f
(f, h_i)
y_i
$sk_f \leftarrow S_2(f, \ldots, y_i, \ldots)[[s]]$
(f, sk_f)
sub-interface simulating $\mathsf{SecC}^{C,B}$
$(pk, s) \leftarrow S_1(1^\kappa)$
pk
sub-interface simulating $\mathsf{AutC}^{C,A}$
(f_i, h)
y_i
$c \leftarrow S_3(\ldots, y_i, \ldots)[[s]]$
h
c
sub-interface simulating $\mathsf{PRep}_{\mathfrak{C}}$

Figure 5.4: Definition and visualization of the simulator σ_B from the proof of Lemma 5.4.2 attached to Rep_F.

$\mathcal{D}$ inputs $h \in H$ at interface B, $\mathcal{A}_1$ saves M and the state of $\mathcal{D}$ in τ and returns $M[h]$. When $\mathcal{D}$ returns a bit b, $\mathcal{A}_1$ sets $\tau \leftarrow (\texttt{return}, b)$ and returns a random $x \in X$. After $\mathcal{A}_1$ terminated, $\mathcal{A}_2$ is invoked on input a ciphertext c and τ. If $\tau = (\texttt{return}, b)$, for some $b \in \{0,1\}$, $\mathcal{A}_2$ returns `true`. Otherwise, it saves c in τ and returns `false`. Afterwards, $\mathcal{A}_1$ is invoked on input pk and $\tau \neq 0$. In this case, it reads c and the state of $\mathcal{D}$ from τ and continues the simulation by outputting c at interface B from the repository. $\mathcal{A}_1$ then proceeds as above.

The distinguisher $\mathcal{D}'$ on input $\tau = (\texttt{return}, b)$ outputs the bit b. Note that the distribution of the outputs of $\mathcal{D}'$ given outputs of the real experiment equals the distribution of the outputs of $\mathcal{D}$ connected to the resource $\pi_A \pi_C \left[\mathsf{PRep}_{\mathfrak{C}}, \mathsf{AutC}^{C,A}, \mathsf{SecC}^{C,B}\right]$ and the output distribution of $\mathcal{D}'$ given outputs of the ideal experiment equals the distribution of the outputs of $\mathcal{D}$ connected to $\sigma_B \mathsf{Rep}_F$. Hence, the corresponding distinguishing advantages are the same. □

The next lemma shows that CFE security is not only sufficient but also necessary for constructions of the desired repository resource. Since this notion is even stronger than [BSW11, Definition 4], known impossibility results translate to our model.

Lemma 5.4.3. *Let σ_B be an efficient converter. Then there exist efficient probabilistic algorithms S_1, S_2, and S_3 such that for all efficient probabilistic oracle algorithms $\mathcal{A}_1$, all efficient probabilistic algorithms $\mathcal{A}_2$, and all efficient distinguishers $\mathcal{D}$ for the CFE experiment, there exists an efficient distinguisher $\mathcal{D}'$ for $\pi_A \pi_C \left[\mathsf{PRep}_{\mathfrak{C}}, \mathsf{AutC}^{C,A}, \mathsf{SecC}^{C,B}\right]$ and $\sigma_B \mathsf{Rep}_F$ such that*

$$\mathsf{Adv}^{\mathsf{CFE}}_{\mathcal{E},S,\mathcal{A},\mathcal{D}} = \Delta^{\mathcal{D}'}\left(\pi_A \pi_C \left[\mathsf{PRep}_{\mathfrak{C}}, \mathsf{AutC}^{C,A}, \mathsf{SecC}^{C,B}\right], \sigma_B \mathsf{Rep}_F\right).$$

Proof. The algorithms S_1, S_2, and S_3 together simulate an execution of σ_B. S_1 starts the simulation, prepares an initially empty map M (i.e., $M[f][h] = \bot$ for all f and h), and sets $(l, q) \leftarrow (0, 0)$. It returns the first output at the outside interface of σ_B together with an encoding of the state of σ_B, l, q, and M in s. On input $f \in F$, $f(x_1), \ldots, f(x_l)$ and some s, S_2 extracts M, l, q, $h_1, \ldots, h_l$, and the state of σ_B from s, sets $q \leftarrow q + 1$, $f_q \leftarrow f$, and $M[f_q][h_i] \leftarrow f(x_i)$ for $i = 1, \ldots, l$. It then inputs f at the inside interface of σ_B. When σ_B outputs (f, sk_f) at its outside interface, S_2 stores the state of σ_B together with M, q, and

f_q in s and returns sk_f. On input $(f_0(x), \dots, f_q(x))$ and s, S_3 extracts the state of σ_B, M, l, q, and $f_1, \dots, f_q$ from s, sets $l \leftarrow l+1$, invokes $h_l \leftarrow$ `getHandle` (if `getHandle` requires interaction with interface A, S_3 emulates it using an arbitrary fixed strategy), sets $M[f_i][h_l] \leftarrow f_i(x)$ for $i = 0, \dots, q$, and inputs h at the outside sub-interface of σ_B simulating the repository. When σ_B outputs c at its outside interface, S_3 saves the state of σ_B, M, l, and h_l in s and returns c. Outputs of the form (f, h) at the inside interface of σ_B are handled equally by S_1, S_2, and S_3 by inputting $M[f][h]$ at its inside interface if $M[f][h] \neq \bot$. Otherwise, that input is ignored. Note that such an input is ignored if and only if f has not been input at the inside interface of σ_B or h has not been input at its outside interface before. This is consistent with Rep_F if all handles returned at interface A are immediately input at interface B afterwards. The distinguisher defined below always does this.

Now let $\mathcal{A}_1$ be an efficient probabilistic oracle algorithm, $\mathcal{A}_2$ an efficient probabilistic algorithm, and let $\mathcal{D}$ be an efficient distinguisher for the CFE experiment. We define a distinguisher $\mathcal{D}'$ for the resources $\pi_A \pi_C \left[\mathsf{PRep}_{\mathfrak{C}}, \mathsf{AutC}^{C,A}, \mathsf{SecC}^{C,B}\right]$ and $\sigma_B \mathsf{Rep}_F$ as follows. It first runs $\mathcal{A}_1$ on input the initial output pk at interface B and $\tau = 0$. A query $f \in F$ from $\mathcal{A}_1$ to its oracle is answered by inputting f at interface C, receiving (f, sk_f) at interface B, and returning sk_f as the answer. When $\mathcal{A}_1$ returns x, $\mathcal{D}'$ inputs x at interface A and then the returned handle h at interface B to the repository (where $\mathcal{D}'$ uses the same strategy as S_3 for inputs at interface A for `getHandle`, such that handles are equally distributed). When c is output at interface B, the distinguisher $\mathcal{D}'$ invokes $\mathcal{A}_2$ on input c and τ. If it returns $t = \texttt{false}$, $\mathcal{D}'$ repeats this procedure by running $\mathcal{A}_1$ on input pk and τ. Otherwise, $\mathcal{D}'$ invokes $\mathcal{D}$ on input τ. Finally, $\mathcal{D}'$ outputs the output of $\mathcal{D}$. Since the distribution of τ if $\mathcal{D}'$ is connected to $\pi_A \pi_C \left[\mathsf{PRep}_{\mathfrak{C}}, \mathsf{AutC}^{C,A}, \mathsf{SecC}^{C,B}\right]$ is the same as in the real CFE experiment and the same as in the ideal one if $\mathcal{D}'$ is connected to $\sigma_B \mathsf{Rep}_F$, the corresponding distinguishing advantages are equal. □

Combining Lemmata 5.3.3, 5.4.2 and 5.4.3, we get the following theorem:

Theorem 5.4.4. *For the protocol π defined above, we have*

$$\left[\mathsf{PRep}_{\mathfrak{C}}, \mathsf{AutC}^{C,A}, \mathsf{SecC}^{C,B}\right] \xRightarrow[B]{\pi} \mathsf{Rep}_F \quad \Longleftrightarrow \quad \mathcal{E} \textit{ is CFE secure.}$$

5.4.3 Alleged Insufficiency of BSW's Definition

The results from the previous section seem to contradict an example given in [BF13], which was meant to show that [BSW11, Definition 4] is not adequate and which can easily be extended to our definition. We first recall the example for a fixed set of functions from the full version of [BF13] and then explain why it is not a problem in our context. Assume $\mathcal{E} = (\mathsf{setup}, \mathsf{keygen}, \mathsf{enc}, \mathsf{dec})$ is a functional encryption scheme for a set F of functions with domain X with $\mathrm{id}_X \in F$ and $P \subseteq F$ where P is a family of trapdoor one-way permutations on X. We consider a modified scheme $\mathcal{E}' = (\mathsf{setup}', \mathsf{keygen}', \mathsf{enc}, \mathsf{dec}')$ as follows. The algorithm setup' first runs setup and samples a permutation $p^* \in P$ according to the key-generation algorithm of the trapdoor one-way permutation. It then includes the description of p^* in the public key $pk' := (pk, p^*)$ and the master secret key $mk' := (mk, p^*)$ (and discards the trapdoor). The algorithm keygen' on input (mk', f) does the same as keygen if $f \neq p^*$, and returns (p^*, sk_{id}) with $sk_{\mathrm{id}} \leftarrow \mathsf{keygen}(mk, \mathrm{id}_X)$ if $f = p^*$. The algorithm enc is not modified and dec' on input $((p^*, sk_{\mathrm{id}}), c)$ returns $p^*(\mathsf{dec}(sk_{\mathrm{id}}, c))$ and $\mathsf{dec}(sk_f, c)$ on input (sk_f, c). As in [BF13], it can be shown that $\mathcal{E}'$ is CFE secure if $\mathcal{E}$ is. Intuitively, the simulator, which generates the public key, can store the trapdoor and hence compute x from $p^*(x)$, enabling it to simulate.

According to [BF13], this scheme should not be considered secure because one can learn x instead of only $p^*(x)$ given a key for p^*. They conclude that the simulator should therefore not be allowed to generate the public key. We claim that this is actually not a problem: An adversary cannot choose for which $p \in P$ he wants to learn x instead of $p(x)$, but some p^* is chosen at random by the key-generation algorithm of the trapdoor one-way permutation. By simply invoking this algorithm themselves, all users can obtain a trapdoor for such a random permutation p^* and hence compute x from $p^*(x)$ even if the original scheme is used. The only difference is that in the modified scheme, this particular permutation is contained in the public key and the master secret key. Using this permutation in another protocol built on top of the modified scheme would be problematic if the designer of the composed protocol assumes that one cannot compute x if given a key for this p^* (which is the case in the original scheme if the description of a random p^* is included in the public key, but not in the modified scheme). However, if schemes are composed according to the constructive cryptography framework, such confusion cannot arise since the protocol converters use the keys only internally and do not publish them to their outside interfaces. Hence, the constructed resource does not provide a distinguished function corresponding to the permutation in the public key to any party. This means that when the constructed resource is used in another protocol, that protocol cannot explicitly use this particular p^*. Because $p^* \in P$ is chosen randomly and the set P needs to be large, the probability that it is still used in another context is negligible. Therefore the modified scheme is as secure as the original one in all applications if the protocols are composed properly.

5.5 Special Cases and Impossibility Results

5.5.1 Public-Key Encryption and its Impossibility

As described in [BSW11], many types of encryption can be seen as special cases of functional encryption. We restate how standard public-key encryption is captured as a special case and explain why this immediately leads to strong impossibility results.

Consider public-key encryption with plaintext space M. We can set $X \coloneqq M$ and $F_{\mathsf{PKE}} \coloneqq \{f_0, \mathrm{id}_X\}$ with $f_0 \colon X \to \mathbb{N}, x \mapsto |x|$ revealing the bit length of x. This provides the same functionality as public-key encryption [BSW11]: The holder of the secret key (corresponding to the key for id_X) can decrypt a ciphertext to the encrypted message while without the secret key, one can only learn the length of the message.

Depending on how the encryption scheme is intended to be used, different security properties are required. Typically, one assumes that there is a legitimate receiver Bob knowing the secret key from the beginning and an eavesdropper Eve who never learns the secret key. The repository resource $\mathsf{Rep}_{F_{\mathsf{PKE}}}$, however, allows to adaptively grant Bob the right to learn the input data. In case of a dishonest Bob, this enables the distinguisher to adaptively retrieve the secret key after receiving ciphertexts. Hence, we are in a situation of adaptive adversaries [CFGN96]. Therefore, the result by Nielsen [Nie02] stating that the length of secret keys in any adaptively secure scheme must be at least the total number of bits to be encrypted, on which the impossibility results in [BSW11; BO13] are based, can be applied directly here. Since there is no restriction on the number of messages Alice can input in the repository, the distinguisher can input messages whose total length exceed an upper bound on the length of secret keys before granting access rights to Bob. We therefore get the following theorem as a direct consequence of Theorem 5.4.4 and the result from [Nie02]. Note that the same reasoning led to our impossibility result for identity-based encryption in Section 4.4.1. This shows another advantage of our constructive approach, namely that defining security of a protocol by what it achieves simplifies reusing results without reproving them for new definitions as in [BSW11] and [BO13].

Theorem 5.5.1. *There is no CFE-secure functional encryption scheme for F_{PKE}.*

As we have seen, while public-key encryption can be considered to be a special case of functional encryption, typical security definitions for public-key encryption do not correspond to the given security definition for functional encryption. This is not a weakness of our security definition but comes from the fact that public-key encryption is usually used in a specific setting and thus a less restrictive definition is sufficient. See [CMT13] for a treatment of public-key encryption in the constructive cryptography framework. Similarly, it is still meaningful to consider security definitions for other special cases of functional encryption such as identity-based and attribute-based encryption that take into account how these schemes are intended to be used.

5.5.2 Circumventing Impossibility Results

As seen in the previous section, realizing CFE-secure functional encryption schemes, even for very simple sets of functions, is impossible without further assumptions. In general, there are two ways to circumvent such impossibility results. One can either start with a stronger assumed resource or construct a weaker resource. The authors in [BSW11] use a random oracle to realize secure functional encryption schemes for a large class of functions. This falls in the first category and can be understood in our model as adding a random oracle to the assumed resource. In Section 5.6, we recast the scheme from [BSW11] in our framework and show that it can be used to construct Rep_F from $[\mathsf{PRep}_{\mathfrak{C}}, \mathsf{AutC}^{C,A}, \mathsf{SecC}^{C,B}, \mathsf{RO}]$. Bellare and O'Neill [BO13] generalize the result from [BSW11] and obtain secure functional encryption schemes without random oracles by allowing the keys to be longer than all encrypted messages together. This can also be interpreted as restricting the amount of data the repository can store. Hence, this is an example of weakening the constructed resource.

Many papers consider different security definitions [ONe10; BO13; GVW12; AGVW13; GKP+13; GGH+13] that are not subject to impossibility results. However, changing a security definition can lead to an inadequate notion of security and it might not be clear how the resulting schemes can be used. A result from [DIJ+13] shows how to transform a functional encryption scheme for the set of all n-input Boolean circuits satisfying a weak security definition to a scheme that satisfies a stronger security definition in the random oracle model. This provides a bridge between considering weaker definitions and using stronger assumed resources (i.e., including random oracles) to achieve strong security notions.

In Section 5.7, we follow the approach of constructing weaker resources. In particular, we introduce restricted repositories and show that schemes satisfying a definition from [GVW12] can be used to construct such repositories.

5.6 Construction with Random Oracles

We recast the scheme presented in [BSW11] as "the modified 'brute-force' construction" in our framework. It allows us to construct Rep_F for all F that contain only a polynomial (in the security parameter) number of functions that all have domain X and codomain $\{0,1\}^\ell$ for some $\ell \in \mathbb{N}$. The construction is similar to the one for identity-based encryption we described in Section 4.5. As there, the assumed resource in our construction contains a random oracle. In this case, however, we assume the random oracle has codomain $\{0,1\}^\ell$ and only the interfaces A, B, and C.

Definition 5.6.1. The resource RO has interfaces A, B, and C. On input $x \in \{0,1\}^*$ at interface $I \in \{A,B,C\}$, if x has not been input before (at any interface), RO chooses $y \in \{0,1\}^\ell$ uniformly at random, outputs y at interface I, and stores (x,y) internally; if x has been input before, RO outputs the value y at interface I that has been stored together with x.

Remark. As in Section 4.5, the simulator in the proof of the construction will answer queries to the random oracle made by the distinguisher. Since the simulator can answer these queries arbitrarily as long as they are consistent with previous answers and appear to be uniform, the simulator has additional power, which allows us to overcome the impossibility result from Section 5.5.1. This setting is often referred to as *programmable random oracle model* because the simulator can in some sense "reprogram" the random oracle.

Now let X be some nonempty set and $F = (f_0, \dots, f_s)$ with $f_i \colon X \to \{0,1\}^\ell$. Further let (K, E, D) be a semantically secure public-key encryption scheme. Let $\mathfrak{C}$ be the set of $s{+}1$-tuples where the first component is an ℓ-bit string and the other components consist of an element from the range of E and an ℓ-bit string. We define the protocol $\pi^{\mathsf{RO}} = (\pi_A^{\mathsf{RO}}, \pi_B^{\mathsf{RO}}, \pi_C^{\mathsf{RO}})$ to construct Rep_F from $[\mathsf{PRep}_{\mathfrak{C}}, \mathsf{AutC}^{C,A}, \mathsf{SecC}^{C,B}, \mathsf{RO}]$ as follows: At the beginning, π_C^{RO} invokes $(pk_i, sk_i) \leftarrow K(1^\kappa)$ for $i \in \{1, \dots, s\}$, stores the sk_i and sends $(pk_1, \dots, pk_s)$ to Alice over the authenticated channel. The public keys are internally stored by π_A^{RO}. On input $x \in X$ at its outside interface, π_A^{RO} chooses $r_1, \dots, r_s$ uniformly at random from $\{0,1\}^\kappa$. It then outputs r_i at its inside interface to the random oracle and receives the answer r'_i for each $i \in \{1, \dots, s\}$. Finally, it outputs $c \leftarrow (f_0(x), (E(pk_1, r_1), r'_1 \oplus f_1(x)), \dots, (E(pk_s, r_s), r'_s \oplus f_s(x)))$ at its inside interface to the repository and outputs the returned handle h at its outside interface. On input $f_i \in F$ at its outside interface, π_C^{RO} sends (f_i, sk_i) to B over the secure channel. The converter π_B^{RO} then stores the secret key and outputs f_i at its outside interface. On input $(f_i, h) \in F \times H$ at its outside interface, π_B^{RO} outputs h at its inside interface to the repository if it has stored the secret key sk_i or if $i = 0$. If it receives $c = (c_0, \dots, c_s)$ from the repository, it outputs c_0 at its outside interface if $i = 0$, and if $i \neq 0$, it decrypts the first component of c_i with $D(sk_i, \cdot)$ and outputs the result at its inside interface to RO. When RO answers with r, π_B^{RO} outputs the bitwise XOR of r and the second component of c_i at its outside interface. All other inputs are ignored.

Theorem 5.6.2. *We have*

$$\left[\mathsf{PRep}_{\mathfrak{C}}, \mathsf{AutC}^{C,A}, \mathsf{SecC}^{C,B}, \mathsf{RO}\right] \overset{\pi^{\mathsf{RO}}}{\underset{B}{\Longmapsto}} \mathsf{Rep}_F.$$

Proof sketch. Correctness is straightforward to verify. To prove the security condition, we consider the simulator σ_B that initially generates $(pk_i, sk_i) \leftarrow K(1^\kappa)$ for $i \in \{1, \dots, s\}$ and outputs $(pk_1, \dots, pk_s)$ at its outside sub-interface simulating $\mathsf{AutC}^{C,A}$. To answer queries to the simulated random oracle, the simulator maintains a list O of all previous inputs and outputs. Queries in O are answered consistently. For a fresh input, the simulator chooses a value uniformly at random from $\{0,1\}^\ell$, outputs this value at its outside sub-interface simulating RO, and updates O. To simulate $\mathsf{PRep}_{\mathfrak{C}}$, as the simulator in the proof of Lemma 5.4.2, σ_B keeps track of all queried handles and the generated ciphertexts, answers repeated queries consistently, and ignores queries for invalid handles. A fresh ciphertext for handle h is generated as $(y_0, (E(pk_1, r_1), r'_1), \ldots, (E(pk_s, r_s), r'_s))$, where y_0 is obtained from Rep_F via the query (f_0, h) and $r_1, \ldots, r_s \in \{0,1\}^\kappa$, $r'_1, \ldots, r'_s \in \{0,1\}^\ell$ are chosen uniformly at random. Moreover, the simulator obtains y_i from Rep_F via the query (f_i, h) for all f_i for which access has been granted, and adds $(r_i, r'_i \oplus y_i)$ to O.

On input $f_i \in F$ at its inside interface, σ_B outputs (f_i, sk_i) at its outside sub-interface simulating $\mathsf{SecC}^{C,B}$. Furthermore, it obtains $y_{j,i}$ from Rep_F via the query (f_i, h_j) for all h_j for which ciphertexts have been simulated. Let $(y_{j,0}, (E(pk_1, r_{j,1}), r'_{j,1}), \ldots, (E(pk_s, r_{j,s}), r'_{j,s}))$ be these ciphertexts. Finally, σ_B adds $(r_{j,i}, r'_{j,i} \oplus y_{j,i})$ to O. Whenever an entry (x, y) is supposed to be added to O and this list already contains an entry for x, the simulator aborts.

Note that if the simulator does not abort, if all r_i chosen by σ_B are unique and different from previous oracle queries by the distinguisher, and if the distinguisher does not query the random oracle on a value r such that $E(pk_i, r)$ is contained in some ciphertext generated by σ_B for some i, the simulation is perfect. Since (K, E, D) is semantically secure, it can be shown that the probability of these events is negligible. Thus, $\pi_A^{\mathsf{RO}} \pi_C^{\mathsf{RO}} [\mathsf{PRep}_{\mathfrak{C}}, \mathsf{AutC}^{C,A}, \mathsf{SecC}^{C,B}, \mathsf{RO}]$ and $\sigma_B \mathsf{Rep}_F$ are computationally indistinguishable. □

5.7 Weaker Security Definitions

5.7.1 Definitions

We now define restricted variants of repository resources, which may still be sufficient for certain applications and can then be used as sketched in Section 5.9.

Definition 5.7.1. Let $L, Q \in \mathbb{N} \cup \{\infty\}$, X be a nonempty set, and let F be a set of functions with domain X and $f_0 \in F$. We define the resource

$\mathsf{Rep}^{\mathrm{AD}}_{F,L,Q}$ to be identical to Rep_F as in Definition 5.3.1 but only allow up to L inputs at interface A and Q inputs at interface C and ignore further inputs (in case L or Q equals ∞, no restriction is placed on the number of inputs, i.e., $\mathsf{Rep}^{\mathrm{AD}}_{F,\infty,\infty} \equiv \mathsf{Rep}_F$). We further define a nonadaptive variant $\mathsf{Rep}^{\mathrm{NA}}_{F,L,Q}$ that ignores all inputs at interface C after the first input at interface A.

Since the resources only accept a given number of inputs at interfaces A and C, we have to adjust the protocols to behave the same way. We therefore consider a functional encryption scheme $\mathcal{E} = (\mathsf{setup}, \mathsf{keygen}, \mathsf{enc}, \mathsf{dec})$ for a set F of functions with domain X where $\mathfrak{C}$ is the range of enc. We then define the protocol $\pi^{\mathrm{AD},L,Q} := \left(\pi_A^L, \pi_B, \pi_C^{\mathrm{AD},Q}\right)$ as π in Section 5.3.2, but π_A^L and $\pi_C^{\mathrm{AD},Q}$ additionally keep track of the number of inputs at their outside interface and ignore all of them after the first L and Q inputs, respectively.

The following lemma states that this protocol yields a restricted repository with access control if all parties are honest. It follows directly from the correctness of the functional encryption scheme.

Lemma 5.7.2. *For the protocol* $\pi^{\mathrm{AD},L,Q} = \left(\pi_A^L, \pi_B, \pi_C^{\mathrm{AD},Q}\right)$ *defined above, we have*

$$\pi_A^L \pi_B \pi_C^{\mathrm{AD},Q}\left[\mathsf{PRep}_{\mathfrak{C}}, \mathsf{AutC}^{C,A}, \mathsf{SecC}^{C,B}\right] \;\approx\; \mathsf{Rep}^{\mathrm{AD}}_{F,L,Q}.$$

To construct the nonadaptive variant, the protocol at interface C has to ignore all inputs after the first input at interface A. Hence, it needs to know whether there has already been any input at interface A, i.e., whether the repository is still empty. We thus introduce for a nonempty set X the resource $\mathsf{PRep}^{\emptyset}_X$ that is identical to PRep_X as in Definition 5.3.2 but additionally accepts the input `isEmpty` at interface C which is answered with `true` if the repository is empty, and `false` otherwise. We then define $\pi^{\mathrm{NA},L,Q} := \left(\pi_A^L, \pi_B, \pi_C^{\mathrm{NA},Q}\right)$ where $\pi_C^{\mathrm{NA},Q}$ on each input at its outside interface outputs `isEmpty` at its inside interface to the repository and ignores the input (and all subsequent inputs) if the repository answers `false`, and otherwise does the same as $\pi_C^{\mathrm{AD},Q}$.

As above, correctness of the functional encryption scheme implies the following lemma.

Lemma 5.7.3. *For the protocol* $\pi^{\mathrm{NA},L,Q} = \left(\pi_A^L, \pi_B, \pi_C^{\mathrm{NA},Q}\right)$ *defined above, we have*

$$\pi_A^L \pi_B \pi_C^{\mathrm{NA},Q}\left[\mathsf{PRep}_{\mathfrak{C}}^{\emptyset}, \mathsf{AutC}^{C,A}, \mathsf{SecC}^{C,B}\right] \;\approx\; \mathsf{Rep}_{F,L,Q}^{\mathrm{NA}}.$$

Having defined these resources and protocols, we can derive the following security definitions.

Definition 5.7.4. Let $\mathcal{E} = (\mathsf{setup}, \mathsf{keygen}, \mathsf{enc}, \mathsf{dec})$ be a functional encryption scheme for a set F of functions with domain X where $\mathfrak{C}$ is the range of enc and let $L, Q \in \mathbb{N} \cup \{\infty\}$. We say $\mathcal{E}$ is (L,Q)*-AD-CFE secure* if

$$\left[\mathsf{PRep}_{\mathfrak{C}}, \mathsf{AutC}^{C,A}, \mathsf{SecC}^{C,B}\right] \;\overset{\pi^{\mathrm{AD},L,Q}}{\underset{B}{\Longmapsto}}\; \mathsf{Rep}_{F,L,Q}^{\mathrm{AD}},$$

and $\mathcal{E}$ is (L,Q)*-NA-CFE secure* if

$$\left[\mathsf{PRep}_{\mathfrak{C}}^{\emptyset}, \mathsf{AutC}^{C,A}, \mathsf{SecC}^{C,B}\right] \;\overset{\pi^{\mathrm{NA},L,Q}}{\underset{B}{\Longmapsto}}\; \mathsf{Rep}_{F,L,Q}^{\mathrm{NA}}.$$

Note that CFE security corresponds to (∞,∞)-AD-CFE security by Theorem 5.4.4. We now recall a simulation-based single-message security definition from [GVW12], which we will compare to our new notions in the next section.

Definition 5.7.5. Let $\mathcal{E} = (\mathsf{setup}, \mathsf{keygen}, \mathsf{enc}, \mathsf{dec})$ be a functional encryption scheme for a set F of functions with common domain X. We introduce the two experiments in Figure 5.5 for an efficient probabilistic oracle algorithm $\mathcal{A}_1$ and efficient probabilistic algorithms $\mathcal{A}_2$ and S. The advantage of a distinguisher $\mathcal{D}$ in distinguishing the outputs of these experiments is denoted by

$$\mathsf{Adv}_{\mathcal{E},S,\mathcal{A},\mathcal{D}}^{\textsf{NA-SIM}} := \Delta^{\mathcal{D}}\left(\mathsf{Exp}_{\mathcal{E},\mathcal{A}}^{\textsf{NA-SIM-real}}, \mathsf{Exp}_{\mathcal{E},S,\mathcal{A}}^{\textsf{NA-SIM-ideal}}\right).$$

For $Q \in \mathbb{N}$, the scheme $\mathcal{E}$ is Q*-NA-SIM secure* if there exists an efficient probabilistic algorithm S such that $\mathsf{Adv}_{\mathcal{E},S,\mathcal{A},\mathcal{D}}^{\textsf{NA-SIM}}$ is negligible for all efficient probabilistic oracle algorithms $\mathcal{A}_1$ that make at most Q queries, all efficient probabilistic algorithms $\mathcal{A}_2$, and for all efficient distinguishers $\mathcal{D}$.

Experiment $\mathsf{Exp}^{\text{NA-SIM-real}}_{\mathcal{E},\mathcal{A}}$

```
Input: $1^\kappa, \kappa \in \mathbb{N}$
  $(pk, mk) \leftarrow \mathsf{setup}(1^\kappa)$
  $(x, \tau) \leftarrow \mathcal{A}_1^{\mathsf{keygen}(mk,\cdot)}(pk)$
  $c \leftarrow \mathsf{enc}(pk, x)$
  $\alpha \leftarrow \mathcal{A}_2(pk, c, \tau)$
  return $(\alpha, x)$
```

Experiment $\mathsf{Exp}^{\text{NA-SIM-ideal}}_{\mathcal{E},S,\mathcal{A}}$

```
Input: $1^\kappa, \kappa \in \mathbb{N}$
  $(pk, mk) \leftarrow \mathsf{setup}(1^\kappa)$
  $(x, \tau) \leftarrow \mathcal{A}_1^{\mathsf{keygen}(mk,\cdot)}(pk)$
  $(f_1, \dots, f_q) \leftarrow$ oracle queries by $\mathcal{A}_1$
  $(sk_1, \dots sk_q) \leftarrow$ replies from oracle
  $(y_0, \dots, y_q) \leftarrow (f_0(x), \dots, f_q(x))$
  $c \leftarrow S(pk, f_1, \dots, f_q, sk_1, \dots sk_q,$
                                   $y_0, \dots, y_q)$
  $\alpha \leftarrow \mathcal{A}_2(pk, c, \tau)$
  return $(\alpha, x)$
```

Figure 5.5: Experiments for the Q-NA-SIM security definition. Note that $q \le Q$ for $\mathcal{A}_1$ that make at most Q queries.

5.7.2 Sufficiency of NA-SIM Security

In this section, we show that Q-NA-SIM security is sufficient to construct a nonadaptive repository with potentially dishonest B. This shows that the scheme constructed in [GVW12], which satisfies this definition, can be used in a composable framework. In particular, this shows how to construct a nonadaptive repository for the set of all functions that are computed by polynomial-size circuits, when the number of granted access rights is bounded.

Note that in contrast to Definition 5.4.1, all keys the adversary sees in the ideal experiment are generated by the algorithms of the functional encryption scheme and not by a simulator. While this is an artificial restriction for constructing single-input repositories, it interestingly allows us to prove that this definition is sufficient to construct a repository for many inputs. A similar result was already shown in [GVW12] but our result is stronger because we allow subsequent inputs to depend on previous ciphertexts whereas the many-message definition in [GVW12] restricts the adversary to input all messages at once before seeing a ciphertext.

Lemma 5.7.6. *Let $L, Q \in \mathbb{N}$ and let S be an efficient probabilistic algorithm. Then there exists an efficient converter σ_B such that for all efficient distinguishers $\mathcal{D}$ for the resources* $\pi_A^L \pi_C^{\mathrm{NA},Q}[\mathsf{PRep}^{\emptyset}_{\mathfrak{C}}, \mathsf{AutC}^{C,A}, \mathsf{SecC}^{C,B}]$

Converter σ_B

```
Initialization
  $(l, q) \leftarrow (0, 0)$
  $(pk, mk) \leftarrow \mathsf{setup}(1^\kappa)$
  **output** $pk$ at outside sub-interface simulating $\mathsf{AutC}^{C,A}$

Inside interface
Input: $f \in F$
  $q \leftarrow q + 1$
  $f_q \leftarrow f$
  $sk_q \leftarrow \mathsf{keygen}(mk, f)$
  **output** $(f, sk_q)$ at outside sub-interface simulating $\mathsf{SecC}^{C,B}$

Outside interface
Input: $h \in H$
  if $\exists k \in \{1, \ldots, l\}$ $h_k = h$ then
      output $c_k$ at outside sub-interface simulating $\mathsf{PRep}^{\emptyset}_{\mathfrak{C}}$
  else if output $(f_0, h)$ at inside interface not ignored then        ▷ data stored for $h$
      $l \leftarrow l + 1$
      $h_l \leftarrow h$
      for $i = 0, \ldots, q$ do
          output $(f_i, h)$ at inside interface, let $y_i$ be the returned value
      $c_l \leftarrow S(pk, f_1, \ldots, f_q, sk_1, \ldots sk_q, y_0, \ldots, y_q)$
      output $c_l$ at outside sub-interface simulating $\mathsf{PRep}^{\emptyset}_{\mathfrak{C}}$
```

Figure 5.6: Definition of the simulator σ_B for the proof of Lemma 5.7.6.

and $\sigma_B \mathsf{Rep}^{\mathrm{NA}}_{F,L,Q}$, there exists an efficient probabilistic oracle algorithm $\mathcal{A}_1$ that makes at most Q queries, an efficient probabilistic algorithm $\mathcal{A}_2$, and an efficient distinguisher $\mathcal{D}'$ for the NA-SIM experiment such that

$$\Delta^{\mathcal{D}}\left(\pi_A^L \pi_C^{\mathrm{NA},Q}\left[\mathsf{PRep}^{\emptyset}_{\mathfrak{C}}, \mathsf{AutC}^{C,A}, \mathsf{SecC}^{C,B}\right], \sigma_B \mathsf{Rep}^{\mathrm{NA}}_{F,L,Q}\right) = L \cdot \mathsf{Adv}^{\mathsf{NA\text{-}SIM}}_{\mathcal{E},S,\mathcal{A},\mathcal{D}'}.$$

Proof. We define the simulator σ_B in Figure 5.6. Now let $\mathcal{D}$ be an efficient distinguisher for $\pi_A^L \pi_C^{\mathrm{NA},Q}\left[\mathsf{PRep}^{\emptyset}_{\mathfrak{C}}, \mathsf{AutC}^{C,A}, \mathsf{SecC}^{C,B}\right]$ and $\sigma_B \mathsf{Rep}^{\mathrm{NA}}_{F,L,Q}$. We can assume without loss of generality that $\mathcal{D}$ does not make any inputs that are ignored by both resources because this cannot influence the distinguishing advantage. Hence, we can assume that after getting a public key from interface B, $\mathcal{D}$ makes up to Q inputs of the form $f \in F$ at interface C. Afterwards, it makes inputs of the form $x \in X$ at interface A and inputs $h \in H$ at interface B for h that were output at interface A before. As in the proof of Lemma 5.4.2, we can also assume without loss of generality that each h is input at most once, because both resources return the same value for each input of the same h.

We let $\mathcal{A}_1$, $\mathcal{A}_2$, and $\mathcal{D}'$ emulate the distinguisher $\mathcal{D}$. At the beginning, the algorithm $\mathcal{A}_1$ sets $l \leftarrow 0$, draws a number $\hat{l} \in \{1, \ldots, L\}$ uniformly at random and outputs pk at interface B from the authenticated channel emulated for $\mathcal{D}$. When $\mathcal{D}$ inputs $f \in F$ at interface C, $\mathcal{A}_1$ makes the oracle-query f and outputs f and the answer sk at interface B from the secure channel. When $\mathcal{D}$ inputs x at interface A, $\mathcal{A}_1$ invokes `getHandle` and outputs the returned handle h at interface A. It further sets $M[h] \leftarrow x$ for a map M. When $\mathcal{D}$ inputs $h \in H$ at interface B, $\mathcal{A}_1$ increments l by one. If $l < \hat{l}$, $\mathcal{A}_1$ outputs $\mathsf{enc}(pk, M[h])$ at interface B from the repository. If $l \geq \hat{l}$, $\mathcal{A}_1$ saves M, the list $f_1, \ldots, f_q$ of queried functions, the answers $sk_1, \ldots, sk_q$, and the state of $\mathcal{D}$ in τ and returns $(M[h], \tau)$. On input (pk, c, τ), $\mathcal{A}_2$ reads M, $f_1, \ldots, f_q$, $sk_1, \ldots, sk_q$, and the state of $\mathcal{D}$ from τ and continues the simulation of $\mathcal{D}$ by outputting c at interface B from the repository. When $\mathcal{D}$ inputs x at interface A, $\mathcal{A}_2$ invokes `getHandle`, outputs the returned handle h at interface A, and sets $M[h] \leftarrow x$. When $\mathcal{D}$ inputs $h \in H$ at interface B, $\mathcal{A}_2$ computes $y_0 \leftarrow f_0(M[h]), \ldots, y_q \leftarrow f_q(M[h])$ and $c' \leftarrow S(pk, f_1, \ldots, f_q, sk_1, \ldots sk_q, y_0, \ldots, y_q)$ and outputs c' at interface B from the repository. Finally, when $\mathcal{D}$ outputs a bit, $\mathcal{A}_2$ returns the same bit. The distinguisher $\mathcal{D}'$ on input (α, x) simply outputs the value α.

Consider for $i = 0, \ldots, L$ the system H_i that corresponds to the resource $\pi_A^L \pi_C^{\mathrm{NA},Q}\big[\mathsf{PRep}_{\mathfrak{C}}^{\emptyset}, \mathsf{AutC}^{C,A}, \mathsf{SecC}^{C,B}\big]$ for the first i inputs of the form $h \in H$ at interface B, and that subsequently on input $h \in H$ at interface B behaves as follows: If h has been output at interface A but not input at interface B before, the resource outputs $S(pk, f_1, \ldots, f_q, sk_1, \ldots sk_q, y_0, \ldots, y_q)$ at interface B, where q is the number of inputs at interface C, f_j corresponds to the j-th input at interface C, sk_j to the value output in return at interface B together with f_j, and $y_j = f_j(x)$ for the value $x \in X$ that was input at interface A before the resource returned h. Inputs $h \in H$ at interface B for h that have not been output before at interface A are ignored, and inputting the same h more than once always yields the same output as after its initial input. Note that

$$\begin{aligned}
\mathsf{H}_L &\equiv \pi_A^L \pi_C^{\mathrm{NA},Q}\big[\mathsf{PRep}_{\mathfrak{C}}^{\emptyset}, \mathsf{AutC}^{C,A}, \mathsf{SecC}^{C,B}\big], \\
\mathsf{H}_0 &\equiv \sigma_B \mathsf{Rep}_{F,L,Q}^{\mathrm{NA}}.
\end{aligned}$$

Further note that

$$\begin{aligned}
&\Pr\big(\mathcal{D}'\ \mathsf{Exp}^{\text{NA-SIM-real}}_{\mathcal{E},\mathcal{A}} = 1\big)\\
&= \sum_{i=1}^{L} \Pr\big(\hat{l} = i\big) \cdot \Pr\Big(\mathcal{D}'\ \mathsf{Exp}^{\text{NA-SIM-real}}_{\mathcal{E},\mathcal{A}} = 1 \mid \hat{l} = i\Big) = \frac{1}{L}\sum_{i=1}^{L} \Pr(\mathcal{D}\mathsf{H}_i = 1)
\end{aligned}$$

and similarly

$$\Pr\Big(\mathcal{D}'\ \mathsf{Exp}^{\text{NA-SIM-ideal}}_{\mathcal{E},S,\mathcal{A}} = 1\Big) = \frac{1}{L}\sum_{i=1}^{L} \Pr(\mathcal{D}\mathsf{H}_{i-1} = 1).$$

We therefore have

$$\begin{aligned}
&\Delta^{\mathcal{D}}\Big(\pi_A^L \pi_C^{\mathrm{NA},Q}\big[\mathsf{PRep}^{\emptyset}_{\mathfrak{C}}, \mathsf{AutC}^{C,A}, \mathsf{SecC}^{C,B}\big], \sigma_B \mathsf{Rep}^{\mathrm{NA}}_{F,L,Q}\Big)\\
&= \Delta^{\mathcal{D}}(\mathsf{H}_L, \mathsf{H}_0)\\
&= \Pr(\mathcal{D}\mathsf{H}_L = 1) - \Pr(\mathcal{D}\mathsf{H}_0 = 1)\\
&= \sum_{i=1}^{L} \Pr(\mathcal{D}\mathsf{H}_i = 1) - \sum_{i=1}^{L} \Pr(\mathcal{D}\mathsf{H}_{i-1} = 1)\\
&= L \cdot \Pr\Big(\mathcal{D}'\ \mathsf{Exp}^{\text{NA-SIM-real}}_{\mathcal{E},\mathcal{A}} = 1\Big) - L \cdot \Pr\Big(\mathcal{D}'\ \mathsf{Exp}^{\text{NA-SIM-ideal}}_{\mathcal{E},S,\mathcal{A}} = 1\Big)\\
&= L \cdot \Delta^{\mathcal{D}'}\Big(\mathsf{Exp}^{\text{NA-SIM-real}}_{\mathcal{E},\mathcal{A}}, \mathsf{Exp}^{\text{NA-SIM-ideal}}_{\mathcal{E},S,\mathcal{A}}\Big)\\
&= L \cdot \mathsf{Adv}^{\text{NA-SIM}}_{\mathcal{E},S,\mathcal{A},\mathcal{D}'}. \qquad \square
\end{aligned}$$

Note that since an efficient distinguisher can only make a polynomial number of inputs at interface A, Lemmata 5.7.3 and 5.7.6 imply the following theorem.

Theorem 5.7.7. *Let $Q \in \mathbb{N}$, $\mathcal{E}$ be a Q-NA-SIM-secure functional encryption scheme, and let $\pi^{\mathrm{NA},\infty,Q}$ be the protocol defined above for $\mathcal{E}$. We then have*

$$\big[\mathsf{PRep}^{\emptyset}_{\mathfrak{C}}, \mathsf{AutC}^{C,A}, \mathsf{SecC}^{C,B}\big] \overset{\pi^{\mathrm{NA},\infty,Q}}{\underset{B}{\Longmapsto}} \mathsf{Rep}^{\mathrm{NA}}_{F,\infty,Q}.$$

Stated equivalently, Q-NA-SIM security implies (∞, Q)-NA-CFE security.

In Figure 5.7, we provide an overview of the implications among the security definitions considered in this chapter.

$$\begin{tikzcd}
\boxed{\text{CFE}} \arrow[r, shift left, "\text{Theorem 5.4.4}"] \arrow[d] & \boxed{(\infty,\infty)\text{-AD-CFE}} \arrow[l, shift left] \arrow[r, shift left] & \boxed{(\infty,Q)\text{-NA-CFE}} \arrow[l, shift left, "/" marking] \\
\boxed{\text{BSW-SIM}} & & \boxed{Q\text{-NA-SIM}} \arrow[ll, "/" marking] \arrow[u, "\text{Theorem 5.7.7}"']
\end{tikzcd}$$

Figure 5.7: Implications among security definitions. BSW-SIM corresponds to [BSW11, Definition 4] and Q-NA-SIM corresponds to the non-adaptive case of [GVW12, Definition 1]. The separations follow from the impossibility of BSW-SIM and CFE security and the possibility of Q-NA-SIM security.

5.8 More General Notions of FE

Recent papers generalize the notion of functional encryption to support functions of several variables [GGJS13; GKL+13; GGG+14; ABF+13] or randomized functions [GJKS15; GGJS13; ABF+13]. While a detailed treatment of these extensions is beyond the scope of this thesis, we sketch how to capture them and further extensions in our model. Using our approach, one only needs to specify the involved resources; the corresponding security definitions are then implied by the constructive cryptography framework and one can extract equivalent traditional security definitions as we have done to obtain CFE security. In contrast to that, adjusting a traditional security definition appropriately to support additional features is a challenging task and it is often unclear which guarantees a specific definition provides. Whether existing definitions are sufficient to achieve the constructions we propose in this section is an open question, but we conjecture that this is not the case.

5.8.1 Dishonest Senders

So far, we have assumed that Alice is always honest, that is in the real world, dishonest users do not encrypt data that is decrypted by someone else. Dropping this assumption is a very natural extension, which is indeed required for many applications. This scenario can be captured by simply considering dishonest users at interface A. According to a general principle of the constructive cryptography framework, security then means that interactions at interface A can also be simulated. Otherwise, all definitions remain unchanged. While in the context of randomized functions [GJKS15; GGJS13], security definitions have been considered that prevent the sender from tampering with the randomness, we are not aware of any results that provide guarantees against dishonest senders beyond preventing them from manipulating randomness.

5.8.2 Randomized Functions

It is straightforward to extend the repositories defined in Definition 5.7.1 to support randomized functions as follows: For a set F of functions with domain $R \times X$, the repository on input (f, h) at interface B chooses $r \in R$ uniformly at random and outputs $f(r, M[h])$ at interface B if the right to access f has been granted to B and $M[h] \neq \bot$. One can consider two cases here: Either the randomness r is chosen freshly for each input at interface B or the resource remembers all inputs and only samples r for fresh inputs (and uses the same randomness if the same pair (f, h) is input again). This can, of course, be combined with dishonest senders as explained above.

5.8.3 Functions of Several Variables

To compute on a database where each entry is encrypted separately, one needs to support functions f with domain X^n. Our constructed resources can be generalized to capture that by answering inputs $(f, h_1, \ldots, h_n) \in F \times H^n$ at interface B with $f(M[h_1], \ldots, M[h_n])$ if the right to access f has been granted. The setting considered in [GGJS13] is even more general: The functions there have domain $X_1 \times \ldots \times X_n$ and the key used to encrypt $x_i \in X_i$ can be different from the one used to encrypt $x_j \in X_j$ for $i \neq j$. To capture this setting, we extend our constructed resource to have interfaces $A_1, \ldots, A_n$ instead of interface A where elements in X_i can be input at interface A_i. Moreover, [GGJS13; GKL+13] allow some of the encryption keys to be secret while others are public. This can be covered easily in our model by adjusting the assumed resource: For the keys that are supposed to be secret, the channel from the key authority C to the corresponding A_i has to be secure instead of only authenticated.

This ensures that dishonest parties cannot learn these keys, in contrast to the (public) keys sent over only authenticated channels.

Remark. Note that the impossibility result in Section 5.5.1 does not rely on the fact that the encryption key is public. Hence, the impossibility extends to the case where all encryption keys are secret.

5.9 Application of Constructed Repository

In this section, we demonstrate how a repository with access control can be used in applications and how this relates to the composition theorem of the constructive cryptography framework. As a simple application, consider a university that wants its students to learn the results of their exams as well as the average result of each exam. This can be accomplished by entering the results into a repository and granting the students appropriate access rights. To formalize this, we introduce the resource ExamDB that can directly be used by the university to enter results such that students are automatically notified about their results. We construct this resource from a repository with access control $\mathsf{Rep}^{\mathrm{NA}}_{F,\infty,Q}$ as introduced in Section 5.7.[4] To notify the students about the results and communicate the handles needed to access the data in the repository, our construction additionally requires authenticated channels to the students.

For simplicity, we describe a resource that allows one student called Bob to access his results. As mentioned in Section 5.3.1, analyzing the security with a single potentially dishonest party is sufficient to guarantee security against several colluding dishonest parties, so the results here can be applied to a real world application with many students. The described protocols can be extended to such a setting in an obvious way by granting each student the appropriate rights and sending the notifications to every student.

We now describe the resource ExamDB in more detail. After the examiner Alice inputs the ID of a lecture together with a list of students and the number of points they were awarded, Bob receives the lecture ID, the number of participants, his number of points, and the average points. More concretely, let $\mathcal{L}$ be the set of lectures and $\mathcal{S}$ be the set of students

[4]Of course, one can instead use Rep_F if it is available, since $\mathsf{Rep}^{\mathrm{NA}}_{F,\infty,Q}$ is more restricted than Rep_F.

registered at the university. Now consider the resource ExamDB that on input $(\texttt{lecture}, ((s_1, p_1), \ldots, (s_n, p_n))) \in \mathcal{L} \times (\mathcal{S} \times \mathbb{N})^*$ at interface A, outputs $(\texttt{lecture}, n, p, \bar{p}) \in \mathcal{L} \times \mathbb{N}^2 \times \mathbb{Q}$ at interface B, where $p = p_j$ with s_j being Bob's student ID and $\bar{p} = \frac{1}{n} \sum_{i=1}^{n} p_i$.

We will construct the resource ExamDB from a repository with access control. To this end, let $X := \mathcal{L} \times (\mathcal{S} \times \mathbb{N})^*$. We do not require the repository to hide the length of stored entries, i.e., we set $f_0 \colon X \to \mathbb{N}, (\texttt{lecture}, ((s_1, p_1), \ldots, (s_n, p_n))) \mapsto n$. We further consider for every student $s \in \mathcal{S}$ the function $f_s \colon X \to \mathbb{N} \cup \{\perp\}$ that maps an entry to the number of points for student s if s occurs in the list, and $\perp$ otherwise. Finally, let $f_{\mathsf{avg}} \colon X \to \mathbb{Q}, (\texttt{lecture}, ((s_1, p_1), \ldots, (s_n, p_n))) \mapsto \frac{1}{n} \sum_{i=1}^{n} p_i$. We set $F := \{f_0, f_{\mathsf{avg}}\} \cup \{f_s \mid s \in \mathcal{S}\}$ and consider $\left[\mathsf{Rep}^{\mathrm{NA}}_{F,\infty,Q}, \mathsf{AutC}^{A,B}\right]$ for $Q \geq 2|\mathcal{S}|$ (because every $s \in \mathcal{S}$ is granted access to f_{avg} and f_s) as the assumed resource in our construction.

The protocol $\pi^{\mathrm{Ex}} := \left(\pi^{\mathrm{Ex}}_A, \pi^{\mathrm{Ex}}_B, \pi^{\mathrm{Ex}}_C\right)$ is defined as follows: Initially, π^{Ex}_C outputs f_{Bob} and f_{avg} at its inside interface. The converter π^{Ex}_A on input $x = \left(\texttt{lecture}, ((s_1, p_1), \ldots, (s_n, p_n))\right) \in \mathcal{L} \times (\mathcal{S} \times \mathbb{N})^*$ at its outside interface, outputs x at its inside interface to the repository. When the repository returns a handle h, π^{Ex}_A sends $(\texttt{lecture}, h)$ to Bob over the authenticated channel. On input $(\texttt{lecture}, h)$ at its inside interface, π^{Ex}_B inputs (f_0, h), (f_{Bob}, h), and (f_{avg}, h) at its inside interface to the repository. Let the returned values from the repository be n, p, and $\bar{p}$, respectively. The converter then outputs $(\texttt{lecture}, n, p, \bar{p})$ at its outside interface.

Proposition 5.9.1. *We have*

$$[\mathsf{Rep}^{\mathrm{NA}}_{F,\infty,Q}, \mathsf{AutC}^{A,B}] \xRightarrow[B]{\pi^{\mathrm{Ex}}} \mathsf{ExamDB}.$$

Proof. It is easy to see that

$$\pi^{\mathrm{Ex}}_A \pi^{\mathrm{Ex}}_B \pi^{\mathrm{Ex}}_C [\mathsf{Rep}^{\mathrm{NA}}_{F,\infty,Q}, \mathsf{AutC}^{A,B}] \equiv \mathsf{ExamDB}.$$

We define a simulator σ_B as follows: Initially, it outputs f_{Bob} and f_{avg} at its outside sub-interface simulating $\mathsf{Rep}^{\mathrm{NA}}_{F,\infty,Q}$. On input $(\texttt{lecture}, n, p, \bar{p})$ at its inside interface, it invokes $h \leftarrow \texttt{getHandle}$, stores $(h, n, p, \bar{p})$ internally, and outputs $(\texttt{lecture}, h)$ at its outside sub-interface simulating $\mathsf{AutC}^{A,B}$. On input (f_0, h), (f_{Bob}, h), or (f_{avg}, h) at its outside interface for some h that has been stored, it outputs the corresponding stored n, p, or $\bar{p}$, respectively, at its outside sub-interface simulating $\mathsf{Rep}^{\mathrm{NA}}_{F,\infty,Q}$. Other inputs are ignored. We then have

$$\pi^{\mathrm{Ex}}_A \pi^{\mathrm{Ex}}_C [\mathsf{Rep}^{\mathrm{NA}}_{F,\infty,Q}, \mathsf{AutC}^{A,B}] \equiv \sigma_B \mathsf{ExamDB}$$

and the claim follows. □

By the composition theorem of constructive cryptography (see Section 2.4.4), one can first construct $\mathsf{Rep}^{\mathrm{NA}}_{F,\infty,Q}$ from a public repository via a protocol π' as demonstrated in Section 5.7 and then use π^{Ex} to construct ExamDB. Due to this modularity, the protocol π^{Ex} does not need to consider functional encryption or know how $\mathsf{Rep}^{\mathrm{NA}}_{F,\infty,Q}$ was constructed. This makes the protocol and its security analysis very simple. The composition theorem guarantees that the overall construction (corresponding to an application of the protocol $\pi^{\mathrm{Ex}} \circ \pi'$) is secure. Moreover, security is still guaranteed if $\mathsf{AutC}^{A,B}$ is in parallel constructed from an insecure channel using some secure authentication protocol. Traditional security definitions for functional encryption do not offer such guarantees.

Note that π^{Ex}_C has to grant Bob the rights before A makes any inputs. Using the constructive approach, this is obvious since $\mathsf{Rep}^{\mathrm{NA}}_{F,\infty,Q}$ does not allow any inputs at interface C after an input at interface A. In a real world application, this corresponds to sending all keys to the students before the exam session starts (and using fresh keys every semester). If a student forgot to register and asks for his key after some results have already been published, the university cannot give him a key without destroying all security guarantees. When a functional encryption scheme is used to build ExamDB from scratch using traditional security definitions, this fact might be less obvious.

Chapter 6

Access Control Encryption

6.1 Introduction

6.1.1 Model and Security Requirements

The concept of *access control encryption (ACE)* has been 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 is 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 has been 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's role.

2. A dishonest sender must be unable to communicate with a (potentially 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 an 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; see also Chapter 5. 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.

6.1.2 Contributions

While the proposal of the ACE concept and of efficient ACE schemes were important first steps toward outsourcing access control, the existing security definition turns out to be insufficient for several realistic attack scenarios. Hence, existing schemes also cannot be used to construct a meaningful ideal resource. The main contributions in this chapter 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.

As we discuss in Section 6.7, our stronger notions are still insufficient to provide the expected guarantees in the constructive cryptography framework. This is not because the definitions are too weak, but due to inherent limitations we discuss in that section.

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 this is disallowed by the security policy, but they do not enforce allowed information flow. For example, if user A only has one 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, nor from 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 that needs to receive all sent messages), 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 sanitizer parameters and a master secret key. The master secret key is needed to generate a public/private key pair and the sanitizer 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 from 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 by Sahai to achieve full CCA security if the zero-knowledge proof is simulation-sound [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 sanitizer parameters, and let senders sign their ciphertexts. To preserve 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.

6.1.3 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], by Tan et al. [TZMT17], and by Kim and Wu [KW17] focused on new schemes that are more efficient, based on different assumptions, or support more fine-grained access control policies. In contrast to our work, they did not attempt to strengthen the security guarantees provided by ACE.

6.2 Existing Definitions for ACE

6.2.1 Access Control Encryption

We recall the definition of access control encryption by Damgård et al. [DHO16]. 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 6.2.1. 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.

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

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

Decryption: 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

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

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. The scheme $\mathcal{E}$ is called *correct* if $\mathsf{Adv}^{\mathsf{ACE\text{-}CORR}}_{\mathcal{E},\mathcal{A}}$ is negligible for all efficient $\mathcal{A}$, and *perfectly correct* if $\mathsf{Adv}^{\mathsf{ACE\text{-}CORR}}_{\mathcal{E},\mathcal{A}} = 0$ for all $\mathcal{A}$.

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 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]. We additionally define strong detectability, in which the randomness for the encryption is adversarially chosen.

Definition 6.2.2. 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 $\mathsf{Exp}^{\mathsf{ACE\text{-}DTCT}}_{\mathcal{E},\mathcal{A}}$ 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

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

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. The scheme $\mathcal{E}$ is called *detectable* if $\mathsf{Adv}^{\mathsf{ACE\text{-}DTCT}}_{\mathcal{E},\mathcal{A}}$ is negligible for all efficient $\mathcal{A}$. The experiment $\mathsf{Exp}^{\mathsf{ACE\text{-}sDTCT}}_{\mathcal{E},\mathcal{A}}$ is identical to $\mathsf{Exp}^{\mathsf{ACE\text{-}DTCT}}_{\mathcal{E},\mathcal{A}}$ except that $\mathcal{A}$ returns (m, r, i, j). The *strong detectability advantage* of $\mathcal{A}$ is defined as

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

where the probability is over the randomness in $\mathsf{Exp}^{\text{ACE-sDTCT}}_{\mathcal{E},\mathcal{A}}$ and the random coins of San and Dec. The scheme $\mathcal{E}$ is called *strongly detectable* if $\mathsf{Adv}^{\text{ACE-sDTCT}}_{\mathcal{E},\mathcal{A}}$ is negligible for all efficient $\mathcal{A}$.

6.2.2 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 its length, or about 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 for roles for which the adversary does not possess 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 next state the definition of the no-read rule.[1]

Definition 6.2.3. Let $\mathcal{E} = (\mathsf{Setup}, \mathsf{Gen}, \mathsf{Enc}, \mathsf{San}, \mathsf{Dec})$ be an ACE scheme and let $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ be a pair of probabilistic algorithms. Consider the experiment $\mathsf{Exp}^{\text{ACE-no-read}}_{\mathcal{E},\mathcal{A}}$ in Figure 6.1 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 $\mathcal{O}_G$. The *payload-privacy advantage* and the *sender-anonymity advantage* of $\mathcal{A}$ are defined as

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

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

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

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

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

```
Input: $(1^\kappa, P), \kappa \in \mathbb{N}$,
       $P\colon [n] \times [n] \to \{0,1\}$
  $(sp, msk) \leftarrow \mathsf{Setup}(1^\kappa, P)$
  $(c_0, i', st) \leftarrow \mathcal{A}_1^{\mathcal{O}_G(\cdot,\cdot),\mathcal{O}_{ES}(\cdot,\cdot)}(sp)$
  $b \twoheadleftarrow \{0,1\}$
  $m' \twoheadleftarrow \mathcal{M}$
  $c_1 \leftarrow \mathsf{Enc}(\mathsf{Gen}(msk, i', \mathtt{sen}), m')$
  $b' \leftarrow \mathcal{A}_2^{\mathcal{O}_G(\cdot,\cdot),\mathcal{O}_{ES}(\cdot,\cdot)}(st, \mathsf{San}(sp, c_b))$
```

Figure 6.1: The no-read and no-write experiments for an ACE scheme $\mathcal{E}$ and an adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$. The oracles are defined as $\mathcal{O}_G(\cdot,\cdot) := \mathsf{Gen}(msk,\cdot,\cdot)$, $\mathcal{O}_E(\cdot,\cdot) := \mathsf{Enc}(\mathsf{Gen}(msk,\cdot,\mathtt{sen}),\cdot)$, and $\mathcal{O}_{ES}(\cdot,\cdot) := \mathsf{San}(sp, \mathsf{Enc}(\mathsf{Gen}(msk,\cdot,\mathtt{sen}),\cdot))$.

no-read rule and the *sender-anonymity no-read rule* if $\mathsf{Adv}^{\mathsf{ACE\text{-}no\text{-}read,priv}}_{\mathcal{E},\mathcal{A}}$ and $\mathsf{Adv}^{\mathsf{ACE\text{-}no\text{-}read,anon}}_{\mathcal{E},\mathcal{A}}$ are negligible for all efficient $\mathcal{A}$, respectively. If it satisfies both, it is said to satisfy the *no-read rule.*

The no-write rule of ACE is the core property to capture access control. In a nutshell, if the adversary only possesses encryption keys for roles i and decryption keys for roles j with $P(i,j) = 0$, then he should not be able to create a ciphertext from which, after being sanitized, he can retrieve any information. Technically, in the corresponding 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 unsanitized ciphertexts generated for roles for which he does not possess 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 encryption of a random message, and if he does not obtain the encryption key 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 6.2.4. Let $\mathcal{E} = (\mathsf{Setup}, \mathsf{Gen}, \mathsf{Enc}, \mathsf{San}, \mathsf{Dec})$ be an ACE scheme and let $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ be a pair of probabilistic algorithms. Consider the experiment $\mathsf{Exp}_{\mathcal{E},\mathcal{A}}^{\mathsf{ACE\text{-}no\text{-}write}}$ in Figure 6.1, let I_1 be the set of all i such that $\mathcal{A}_1$ issued the query $(i, \mathtt{sen})$ to $\mathcal{O}_G$, 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 $\mathcal{O}_G$. We define the *no-write advantage* of $\mathcal{A}$ as

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

where the probability is over the randomness of all algorithms in the experiment $\mathsf{Exp}_{\mathcal{E},\mathcal{A}}^{\mathsf{ACE\text{-}no\text{-}write}}$. The scheme $\mathcal{E}$ satisfies the *no-write rule* if $\mathsf{Adv}_{\mathcal{E},\mathcal{A}}^{\mathsf{ACE\text{-}no\text{-}write}}$ is negligible for all efficient $\mathcal{A}$.

Remark. Our definition follows the one by Fuchsbauer et al. [FGKO17] by requiring $\mathsf{San}(sp, c_0) \neq \bot$ in the winning condition for the no-write rule, which was not required in the original definition by Damgård et al. [DHO16]. Schemes can be made secure with respect to the original definition by letting the algorithm San create a fresh ciphertext for a random message when given an invalid ciphertext.

The condition $i' \in I_1$ together with $\forall i \in I_1 \ \forall j \in J \ P(i,j) = 0$ ensures that $\mathcal{A}$ does not have a key to decrypt c_1, which would trivially allow to distinguish. Requiring that $\mathcal{A}$ obtains a key for i' however excludes adversaries that obtain no key at all. The original definitions [DHO16] therefore include a special role 0 with $P(0,j) = 0$ for all j. One can then assume without loss of generality that anyone obtains a key for this role. Since assuming the existence of such a role appears to be a technicality that is only needed for the no-write rule, we do not make this assumption and present new security definitions in Section 6.4.2 that do not rely on such a role.

6.3 Ciphertext-Revealing Attacks

6.3.1 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 finally 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.

Consider the policy defined by

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

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,

$$\mathsf{maul}_{\mathcal{E}}\big(\mathsf{Enc}(ek_i, m; r), sp, m, m'\big) = \mathsf{Enc}(ek_i, m'; r). \tag{6.1}$$

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].

6.3.2 DHO Scheme Based on ElGamal

We briefly recall the ElGamal based ACE scheme for a single identity. The sanitizer 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

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

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

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

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 (6.1) 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.

6.3.3 FGKO Scheme Based on ElGamal

Description of the scheme. In that scheme, the sanitizer 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 $L := \{x \mid \exists w\ (x, w) \in R\}$, 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 $ek := (g^x, \sigma^{\mathsf{Sig}})$, $dk := x$ 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 $(c_0, c_1, c_2, c_3) := (g^r, g^{x\cdot r}, g^s, m \cdot g^{x\cdot s})$. 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 $c := (c_0, c_1, c_2, c_3, \pi)$.

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

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

This function satisfies equation (6.1) 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 $w = \big(g^x, \sigma^{\mathsf{Sig}}, r, s\big)$, $(x, w) \in R'$ 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:

$$\mathsf{NIZK.Gen}(1^\kappa) := \mathsf{NIZK}'\mathsf{.Gen}(1^\kappa),$$

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

and

$$\mathsf{NIZK.Ver}\big(crs^{\mathsf{NIZK}}, (vk^{\mathsf{Sig}}, c_0, c_1, c_2, c_3), \pi^{\mathsf{NIZK}}\big) := \\ \mathsf{NIZK'.Ver}\big(crs^{\mathsf{NIZK}}, (vk^{\mathsf{Sig}}, c_0, c_1, c_2), \pi^{\mathsf{NIZK}}\big).$$

Correctness and zero-knowledge of NIZK follow straightforwardly from the underlying scheme $\mathsf{NIZK'}$. For knowledge-extraction, assume that $\mathsf{NIZK'}$ is capable of extracting a valid witness $(g^x, \sigma^{\mathsf{Sig}}, r, s)$ given a valid proof for the statement $(vk^{\mathsf{Sig}}, c_0, c_1, c_2)$. Given a statement $\big(vk^{\mathsf{Sig}}, c_0, c_1, c_2, c_3\big)$ in the original language L, we can obtain a valid message encoded in c_3 by computing $m := c_3 \cdot (g^{x \cdot s})^{-1}$, and thus also a witness $\big(g^x, \sigma^{\mathsf{Sig}}, m, r, s\big)$ for the given statement. 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 (6.1) 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.

6.4 A Stronger Notion of ACE

In this section, we introduce our new security definitions, which exclude the issues we have discovered.

6.4.1 ACE with Modification Detection

To be resilient against the ciphertext-revealing attacks described in Section 6.3, 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 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 6.4.1. 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 6.4.3, where we show that our new definitions imply the existing ones, we will from now on 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 the 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 c such that a ciphertext produced by Enc for a message of the adversary's choice is detected to be a modification of c. In particular, independent encryptions of messages collide only with negligible probability.

The first requirement is captured by role-respecting security as defined in Definition 6.4.5, the second one by non-detection of fresh encryptions defined in Definition 6.4.4.

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.

6.4.2 New 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 6.4.2. 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 $\mathsf{Exp}_{\mathcal{E},\mathcal{A}}^{\mathsf{ACE\text{-}PRV\text{-}ANON\text{-}CCA}}$ in Figure 6.2 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 under chosen-ciphertext attacks advantage* and the *sender-anonymity under chosen-*

Exper. $\mathsf{Exp}_{\mathcal{E},\mathcal{A}}^{\mathsf{ACE\text{-}PRV\text{-}ANON\text{-}CCA}}$

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

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

```
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
        $m_{0,j} \leftarrow \mathsf{Dec}\big(\mathsf{Gen}(msk, j, \mathtt{rec}), c_0'\big)$
        $m_{1,j} \leftarrow \mathsf{Dec}\big(\mathsf{Gen}(msk, j, \mathtt{rec}), c_1'\big)$
```

Exper. $\mathsf{Exp}_{\mathcal{E},\mathcal{A}}^{\mathsf{ACE\text{-}NDTCT\text{-}FENC}}$

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

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

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

Definitions of oracles

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

Figure 6.2: 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.

ciphertext attacks advantages of $\mathcal{A}$ as

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

respectively, where all probabilities are over the randomness in the experiment $\mathsf{Exp}^{\mathsf{ACE\text{-}PRV\text{-}ANON\text{-}CCA}}_{\mathcal{E},\mathcal{A}}$. We call the scheme $\mathcal{E}$ *private under chosen-ciphertext attacks (PRV-CCA secure)*, *weakly sender-anonymous under chosen-ciphertext attacks (wANON-CCA secure)*, and *strongly sender-anonymous under chosen-ciphertext attacks (sANON-CCA secure)* if $\mathsf{Adv}^{\mathsf{ACE\text{-}PRV\text{-}CCA}}_{\mathcal{E},\mathcal{A}}$, $\mathsf{Adv}^{\mathsf{ACE\text{-}wANON\text{-}CCA}}_{\mathcal{E},\mathcal{A}}$, and $\mathsf{Adv}^{\mathsf{ACE\text{-}sANON\text{-}CCA}}_{\mathcal{E},\mathcal{A}}$ are negligible for all efficient $\mathcal{A}$, respectively.

Remark. Weak anonymity corresponds to the anonymity notion 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. This provides no security guarantees if the adversary can decrypt the ciphertexts, which does not seem to be an issue since in this case, the parties can anyway directly communicate via the messages. However, we additionally consider a stronger variant, where the adversary is allowed to possess a decryption key that decrypts the ciphertexts, as long as they both decrypt to the same message. This stronger variant excludes subliminal channels, i.e., even if the involved parties are allowed to communicated by the policy, they cannot exchange information via ciphertexts beyond the encrypted message.

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 (in the weak variant) that the adversary should not be able to decrypt by only considering the policy and the obtained decryption keys, as in the no-write rule in Definition 6.2.4. Instead, we require that the decryption algorithm returns $\perp$ for all decryption keys the adversary possesses. For this to provide the intended security, the decrypt algorithm should return $\perp$ 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 6.4.3. 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 6.2 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 under chosen-ciphertext attacks advantage* and the *strong sanitization under chosen-ciphertext attacks advantage* of $\mathcal{A}$ as

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

and

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

respectively, where both probabilities are over the randomness in the experiment $\mathsf{Exp}^{\mathsf{ACE\text{-}SAN\text{-}CCA}}_{\mathcal{E},\mathcal{A}}$. The scheme $\mathcal{E}$ is called *sanitization under chosen-ciphertext attacks secure (SAN-CCA secure)* and *strongly sanitization under chosen-ciphertext attacks secure (sSAN-CCA secure)* if $\mathsf{Adv}^{\mathsf{ACE\text{-}SAN\text{-}CCA}}_{\mathcal{E},\mathcal{A}}$ and $\mathsf{Adv}^{\mathsf{ACE\text{-}sSAN\text{-}CCA}}_{\mathcal{E},\mathcal{A}}$ are negligible for all efficient $\mathcal{A}$, respectively.

Non-detection of fresh encryptions. 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 6.4.4. 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 $\mathsf{Exp}_{\mathcal{E},\mathcal{A}}^{\mathsf{ACE\text{-}NDTCT\text{-}FENC}}$ in Figure 6.2. We define the *non-detection of fresh encryptions advantage* of $\mathcal{A}$ as

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

where the probability is over the randomness in $\mathsf{Exp}_{\mathcal{E},\mathcal{A}}^{\mathsf{ACE\text{-}NDTCT\text{-}FENC}}$. The scheme $\mathcal{E}$ is said to have *non-detecting fresh encryptions (NDTCT-FENC)* if $\mathsf{Adv}_{\mathcal{E},\mathcal{A}}^{\mathsf{ACE\text{-}NDTCT\text{-}FENC}}$ is negligible for all efficient $\mathcal{A}$.

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.[2] 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

[2]Detectability (Definition 6.2.2) provides this guarantee for honest encryptions, role-respecting security extends this to maliciously generated ciphertexts. Note, however, that detectability is not implied by role-respecting security: If an adversary has encryption keys for two roles i and i', role-respecting security does not exclude that encrypting some message (depending on i') with the key for role i can be decrypted with keys for roles that are allowed to receive from i'.

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 6.4.5. 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 6.2 and let I and J be the sets of all i and j such that $\mathcal{A}$ issued the query $(i, \texttt{sen})$ and $(j, \texttt{rec})$ to the oracle $\mathcal{O}_G$, respectively. We define the *role-respecting advantage* and the *uniform-decryption advantage* of $\mathcal{A}$ as

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

respectively, where both probabilities are over the randomness in the epxeriment $\mathsf{Exp}^{\mathsf{ACE\text{-}URR}}_{\mathcal{E},\mathcal{A}}$. The scheme $\mathcal{E}$ is *role-respecting (RR secure)* and *uniform-decryption (UDEC) secure* if $\mathsf{Adv}^{\mathsf{ACE\text{-}RR}}_{\mathcal{E},\mathcal{A}}$ and $\mathsf{Adv}^{\mathsf{ACE\text{-}UDEC}}_{\mathcal{E},\mathcal{A}}$ are negligible for all efficient $\mathcal{A}$, respectively.

Remark. Note that in Definition 6.4.5, 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.

6.4.3 Relation to the Original Security Notions

In this section, we discuss how our notions relate to the original security definitions (see Section 6.2.2). First note that we assume the scheme has an additional algorithm DMod. As explained in Section 6.4.1, the intended usage of such a scheme is that the sanitizer discards ciphertexts that are detected to be a modification of a previous ciphertext. This means that if dishonest parties want to communicate even though disallowed by the policy (i.e., they want to break the no-write rule), the sender must produce a ciphertext that is not detected as a modification of a previous ciphertext. With this in mind, it is natural to adjust the no-write rule such that an adversary only wins if the ciphertext he outputs is not detected to be a modification of a ciphertext generated by the oracle $\mathcal{O}_{ES}$ (before sanitizing it).

Definition 6.4.6. Let 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. The experiment $\mathsf{Exp}_{\mathcal{E},\mathcal{A}}^{\mathsf{ACE\text{-}MD\text{-}no\text{-}write}}$ is identical to $\mathsf{Exp}_{\mathcal{E},\mathcal{A}}^{\mathsf{ACE\text{-}no\text{-}write}}$ in Figure 6.1 except that after $\mathcal{A}_1$ returns (c_0, i', st), it is checked whether the oracle $\mathcal{O}_{ES}$ has generated some $\tilde{c}$ and returned its sanitization such that $\mathsf{DMod}(sp, \tilde{c}, c_0) = 1$. If this is the case, set $\mathtt{dct} \leftarrow \mathtt{true}$, else $\mathtt{dct} \leftarrow \mathtt{false}$. Let I_1 be the set of all i such that $\mathcal{A}_1$ issued the query $(i, \mathtt{sen})$ to $\mathcal{O}_G$, 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 $\mathcal{O}_G$. We define the *no-write with modification detection advantage* of $\mathcal{A}$ as

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

where the probability is over the randomness in $\mathsf{Exp}_{\mathcal{E},\mathcal{A}}^{\mathsf{ACE\text{-}MD\text{-}no\text{-}write}}$. We say the scheme $\mathcal{E}$ satisfies the *no-write with modification detection rule* if $\mathsf{Adv}_{\mathcal{E},\mathcal{A}}^{\mathsf{ACE\text{-}MD\text{-}no\text{-}write}}$ is negligible for all efficient $\mathcal{A}$.

We show that our new security definitions from Section 6.4.2 imply the no-read rule and the no-write with modification detection rule. We have to assume that the policy P allows for all i that one can efficiently find some j with $P(i,j) = 1$. This seems to be the case for all practically relevant policies, though. The results are summarized in the following theorem.

Theorem 6.4.7. *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. If $\mathcal{E}$ is correct and PRV-CCA, sANON-CCA, SAN-CCA, and RR secure, then it satisfies the the no-write with modification detection rule for policies P such that for all i, one can efficiently find some j with $P(i, j) = 1$, and $\mathcal{E}'$ satisfies the no-read rule. More precisely, for all adversaries $\mathcal{A}$, $\mathcal{A}'$, and $\mathcal{A}''$, there exist adversaries $\mathcal{A}_{\mathsf{PRV}}$ and $\mathcal{A}_{\mathsf{wANON}}$ (both roughly as efficient as emulating an execution of $\mathsf{Exp}^{\mathsf{ACE\text{-}no\text{-}read}}_{\mathcal{E}',\mathcal{A}}$), an adversary $\mathcal{A}'_{\mathsf{sANON}}$ (roughly as efficient as emulating an execution of $\mathsf{Exp}^{\mathsf{ACE\text{-}no\text{-}read}}_{\mathcal{E}',\mathcal{A}'}$), and adversaries $\mathcal{A}''_{\mathsf{SAN}}$, $\mathcal{A}''_{\mathsf{RR}}$, and $\mathcal{A}''_{\mathsf{CORR}}$ (all roughly as efficient as emulating an execution of $\mathsf{Exp}^{\mathsf{ACE\text{-}MD\text{-}no\text{-}write}}_{\mathcal{E},\mathcal{A}''}$) such that

$$\begin{aligned}
\mathsf{Adv}^{\mathsf{ACE\text{-}no\text{-}read,priv}}_{\mathcal{E}',\mathcal{A}} &\le \mathsf{Adv}^{\mathsf{ACE\text{-}PRV\text{-}CCA}}_{\mathcal{E},\mathcal{A}_{\mathsf{PRV}}} + \mathsf{Adv}^{\mathsf{ACE\text{-}wANON\text{-}CCA}}_{\mathcal{E},\mathcal{A}_{\mathsf{wANON}}}, \\
\mathsf{Adv}^{\mathsf{ACE\text{-}no\text{-}read,anon}}_{\mathcal{E}',\mathcal{A}'} &= \mathsf{Adv}^{\mathsf{ACE\text{-}sANON\text{-}CCA}}_{\mathcal{E},\mathcal{A}'_{\mathsf{sANON}}}, \\
\mathsf{Adv}^{\mathsf{ACE\text{-}MD\text{-}no\text{-}write}}_{\mathcal{E},\mathcal{A}''} &\le \mathsf{Adv}^{\mathsf{ACE\text{-}SAN\text{-}CCA}}_{\mathcal{E},\mathcal{A}''_{\mathsf{SAN}}} + 4 \cdot \mathsf{Adv}^{\mathsf{ACE\text{-}RR}}_{\mathcal{E},\mathcal{A}''_{\mathsf{RR}}} + 2 \cdot \mathsf{Adv}^{\mathsf{ACE\text{-}CORR}}_{\mathcal{E},\mathcal{A}''_{\mathsf{CORR}}}.
\end{aligned}$$

We first sketch the proof idea, a detailed proof can be found below. To prove the claim about the payload-privacy no-read rule, consider the hybrid experiment H that is identical to $\mathsf{Exp}^{\mathsf{ACE\text{-}no\text{-}read}}_{\mathcal{E}',\mathcal{A}}$ except that after $\mathcal{A}_1$ returns (m_0, m_1, i_0, i_1, st), i_1 is replaced by i_0. If $\mathcal{A}$ wins the no-read privacy game, $P(i_0, j) = P(i_1, j) = 0$ for all j for which $\mathcal{A}$ obtained a decryption key. Hence, in this case $\mathsf{Exp}^{\mathsf{ACE\text{-}no\text{-}read}}_{\mathcal{E}',\mathcal{A}}$ and H are indistinguishable by weak sender-anonymity. If $\mathcal{A}$ wins in H, one can construct an adversary against PRV-CCA security by running $\mathcal{A}$, returning (m_0, m_1, i_0, i_0, st) when $\mathcal{A}_1$ returns (m_0, m_1, i_0, i_1, st), and returning the same guess as $\mathcal{A}_2$. Note that $\mathcal{A}$ has access to an encryption oracle $\mathcal{O}_E$ in $\mathsf{Exp}^{\mathsf{ACE\text{-}no\text{-}read}}_{\mathcal{E}',\mathcal{A}}$, which is not available in $\mathsf{Exp}^{\mathsf{ACE\text{-}PRV\text{-}ANON\text{-}CCA}}_{\mathcal{E},\mathcal{A}}$. However, since the winning conditions do not restrict the encryption keys obtained from $\mathcal{O}_G$, the oracle $\mathcal{O}_E$ can be emulated by obtaining the encryption key and then encrypting the message.

Relating the sender-anonymity no-read rule to sANON-CCA security is a straightforward reduction.

To prove the claim about the no-write rule, assume $\mathcal{A}''$ wins the corresponding game. If $\mathcal{A}''$ does not obtain a decryption key that decrypts c_0 or c_1 to a message different from $\perp$, this adversary can be used to break SAN-CCA security as follows: when $\mathcal{A}''_1$ returns (c_0, i', st), output c_0 and the encryption of a uniformly chosen message for sender role i' as c_1; finally output the same guess b' as $\mathcal{A}''_2$. Correctness ensures that

c_1 does not sanitize to $\bot$,[3] so the winning condition of the SAN-CCA game is satisfied. If $\mathcal{A}''$ does obtain a decryption key that decrypts c_0 or c_1 to a message different from $\bot$, one can construct an adversary against role-respecting security.

We now formally prove the three claims in Theorem 6.4.7 as separate lemmata, starting with the payload-privacy no-read rule.

Lemma 6.4.8. *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. Further let $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ be a pair of probabilistic algorithms. Then, there exist adversaries $\mathcal{A}_{\mathsf{PRV}}$ and $\mathcal{A}_{\mathsf{wANON}}$ (both roughly as efficient as emulating an execution of $\mathsf{Exp}^{\mathsf{ACE\text{-}no\text{-}read}}_{\mathcal{E},\mathcal{A}}$) such that*

$$\mathsf{Adv}^{\mathsf{ACE\text{-}no\text{-}read,priv}}_{\mathcal{E}',\mathcal{A}} \le \mathsf{Adv}^{\mathsf{ACE\text{-}PRV\text{-}CCA}}_{\mathcal{E},\mathcal{A}_{\mathsf{PRV}}} + \mathsf{Adv}^{\mathsf{ACE\text{-}wANON\text{-}CCA}}_{\mathcal{E},\mathcal{A}_{\mathsf{wANON}}}.$$

Proof. We assume without loss of generality that $\mathcal{A}$ ensures $|m_0| = |m_1|$ and $P(i_0, j) = P(i_1, j) = 0$ for all $j \in J$, where J is the set of all j such that $\mathcal{A}_1$ or $\mathcal{A}_2$ issued the query $(j, \mathtt{rec})$ to the oracle $\mathcal{O}_G$. Let H be identical to $\mathsf{Adv}^{\mathsf{ACE\text{-}no\text{-}read,priv}}_{\mathcal{E}',\mathcal{A}}$ except that after $\mathcal{A}_1$ returns (m_0, m_1, i_0, i_1, st), i_1 is replaced by i_0. We first show that the probability that b is guessed correctly in H and $\mathsf{Adv}^{\mathsf{ACE\text{-}no\text{-}read,priv}}_{\mathcal{E}',\mathcal{A}}$ differ only negligibly if the scheme satisfies weak anonymity. Note that if $b = 0$, the two experiments are identical, which implies

$$\Pr^{\mathsf{Exp}^{\mathsf{ACE\text{-}no\text{-}read}}_{\mathcal{E}',\mathcal{A}}}\big[b' = b \mid b = 0\big] = \Pr^{H}\big[b' = b \mid b = 0\big]. \tag{6.2}$$

Claim 1. *There exists an adversary $\mathcal{A}_{\mathsf{wANON}}$ such that*

$$\Pr^{\mathsf{Exp}^{\mathsf{ACE\text{-}no\text{-}read}}_{\mathcal{E}',\mathcal{A}}}\big[b' = b \mid b = 1\big] - \Pr^{H}\big[b' = b \mid b = 1\big] = \mathsf{Adv}^{\mathsf{ACE\text{-}wANON\text{-}CCA}}_{\mathcal{E},\mathcal{A}_{\mathsf{wANON}}}.$$

Proof of claim. We construct $\mathcal{A}_{\mathsf{wANON}}$ as follows. On input sp, it emulates an execution of $\mathsf{Exp}^{\mathsf{ACE\text{-}no\text{-}read}}_{\mathcal{E}',\mathcal{A}}$, where the oracles for $\mathcal{A}$ are emulated as follows.

$\mathcal{O}_G(\cdot,\cdot)$**:** Relay queries to the oracle $\mathcal{O}_G$ of $\mathsf{Exp}^{\mathsf{ACE\text{-}PRV\text{-}ANON\text{-}CCA}}_{\mathcal{E},\mathcal{A}_{\mathsf{wANON}}}$.

[3] This is the only place where we need that one can efficiently find j with $P(i', j) = 1$ since the adversary in the correctness game has to provide such a role j.

$\mathcal{O}_E(\cdot,\cdot)$: On query (j, m), query $(j, \texttt{sen})$ to the oracle $\mathcal{O}_G$ to receive the encryption key ek_j. Then compute $c \leftarrow \mathsf{Enc}(ek_j, m)$ and return c.

When $\mathcal{A}$ outputs (m_0, m_1, i_0, i_1, st), $\mathcal{A}_{\mathsf{wANON}}$ gives (m_1, m_1, i_0, i_1) to the challenger to obtain a ciphertext c^*, which is given to $\mathcal{A}_2$. When $\mathcal{A}_2$ returns b', $\mathcal{A}_{\mathsf{wANON}}$ returns the same bit b'. Note that if $b = 0$, $\mathcal{A}_{\mathsf{wANON}}$ perfectly emulates H with $b = 1$, and if $b = 1$, $\mathcal{A}_{\mathsf{wANON}}$ perfectly emulates $\mathsf{Exp}^{\mathsf{ACE\text{-}no\text{-}read}}_{\mathcal{E}',\mathcal{A}}$ with $b = 1$. Hence,

$$\begin{aligned}
&\Pr^{\mathsf{Exp}^{\mathsf{ACE\text{-}no\text{-}read}}_{\mathcal{E}',\mathcal{A}}}\big[b' = b \mid b = 1\big] - \Pr^{H}\big[b' = b \mid b = 1\big] \\
&= \Pr^{\mathsf{Exp}^{\mathsf{ACE\text{-}PRV\text{-}ANON\text{-}CCA}}_{\mathcal{E},\mathcal{A}_{\mathsf{wANON}}}}\big[b' = 1 \mid b = 1\big] - \Pr^{\mathsf{Exp}^{\mathsf{ACE\text{-}PRV\text{-}ANON\text{-}CCA}}_{\mathcal{E},\mathcal{A}_{\mathsf{wANON}}}}\big[b' = 1 \mid b = 0\big] \\
&= 2 \cdot \bigg(\frac{1}{2} \Pr^{\mathsf{Exp}^{\mathsf{ACE\text{-}PRV\text{-}ANON\text{-}CCA}}_{\mathcal{E},\mathcal{A}_{\mathsf{wANON}}}}\big[b' = b \mid b = 1\big] \\
&\qquad - \frac{1}{2}\Big(1 - \Pr^{\mathsf{Exp}^{\mathsf{ACE\text{-}PRV\text{-}ANON\text{-}CCA}}_{\mathcal{E},\mathcal{A}_{\mathsf{wANON}}}}\big[b' = b \mid b = 0\big]\Big)\bigg) \\
&= 2 \cdot \Pr^{\mathsf{Exp}^{\mathsf{ACE\text{-}PRV\text{-}ANON\text{-}CCA}}_{\mathcal{E},\mathcal{A}_{\mathsf{wANON}}}}\big[b' = b\big] - 1.
\end{aligned}$$

Note that $\mathcal{A}_{\mathsf{wANON}}$ returns the same message m_1 twice and we have $P(i_0, j) = P(i_1, j) = 0$ for all $j \in J$ by the assumption on $\mathcal{A}$. This implies $\mathsf{Adv}^{\mathsf{ACE\text{-}wANON\text{-}CCA}}_{\mathcal{E},\mathcal{A}_{\mathsf{wANON}}} = 2 \cdot \Pr^{\mathsf{Exp}^{\mathsf{ACE\text{-}PRV\text{-}ANON\text{-}CCA}}_{\mathcal{E},\mathcal{A}_{\mathsf{wANON}}}}\big[b' = b\big] - 1$ and concludes the proof of the claim. $\Diamond$

Combining Claim 1 and equation (6.2), we obtain

$$\begin{aligned}
&\Pr^{\mathsf{Exp}^{\mathsf{ACE\text{-}no\text{-}read}}_{\mathcal{E}',\mathcal{A}}}\big[b' = b\big] \\
&= \frac{1}{2} \cdot \Pr^{\mathsf{Exp}^{\mathsf{ACE\text{-}no\text{-}read}}_{\mathcal{E}',\mathcal{A}}}\big[b' = b \mid b = 0\big] + \frac{1}{2} \cdot \Pr^{\mathsf{Exp}^{\mathsf{ACE\text{-}no\text{-}read}}_{\mathcal{E}',\mathcal{A}}}\big[b' = b \mid b = 1\big] \\
&= \frac{1}{2} \Pr^{H}\big[b' = b \mid b = 0\big] + \frac{1}{2}\Big(\mathsf{Adv}^{\mathsf{ACE\text{-}wANON\text{-}CCA}}_{\mathcal{E},\mathcal{A}_{\mathsf{wANON}}} + \Pr^{H}\big[b' = b \mid b = 1\big]\Big) \\
&= \Pr^{H}\big[b' = b\big] + \frac{1}{2} \cdot \mathsf{Adv}^{\mathsf{ACE\text{-}wANON\text{-}CCA}}_{\mathcal{E},\mathcal{A}_{\mathsf{wANON}}}.
\end{aligned}$$

Hence,

$$\mathsf{Adv}^{\mathsf{ACE\text{-}no\text{-}read,priv}}_{\mathcal{E}',\mathcal{A}} = 2 \cdot \Pr^{H}\big[b' = b\big] - 1 + \mathsf{Adv}^{\mathsf{ACE\text{-}wANON\text{-}CCA}}_{\mathcal{E},\mathcal{A}_{\mathsf{wANON}}}. \tag{6.3}$$

We now construct the adversary $\mathcal{A}_{\mathsf{PRV}}$. When invoked on input sp, it starts an emulation of H by passing sp to $\mathcal{A}$. The oracles for $\mathcal{A}$ are emulated as in the proof of Claim 1. When $\mathcal{A}_1$ returns (m_0, m_1, i_0, i_1, st), $\mathcal{A}_{\mathsf{wANON}}$ gives (m_0, m_1, i_0, i_0) to the challenger to obtain a ciphertext c^*, which is then given to $\mathcal{A}_2$. When $\mathcal{A}_2$ returns b', $\mathcal{A}_{\mathsf{PRV}}$ returns the same bit b'. Note that the view of $\mathcal{A}$ in this emulation is identical to its view in H. Since $\mathcal{A}_{\mathsf{PRV}}$ returns the same role i_0 twice and $P(i_0, j) = 0$ for all $j \in J$ by the assumption on $\mathcal{A}$, we have

$$\mathsf{Adv}^{\mathsf{ACE\text{-}PRV\text{-}CCA}}_{\mathcal{E},\mathcal{A}_{\mathsf{PRV}}} = 2 \cdot \Pr^{\mathsf{Exp}^{\mathsf{ACE\text{-}PRV\text{-}ANON\text{-}CCA}}_{\mathcal{E},\mathcal{A}_{\mathsf{PRV}}}}\big[b' = b\big] - 1 = 2 \cdot \Pr^{H}\big[b' = b\big] - 1.$$

Using equation (6.3), we conclude

$$\mathsf{Adv}^{\mathsf{ACE\text{-}no\text{-}read,priv}}_{\mathcal{E}',\mathcal{A}} = \mathsf{Adv}^{\mathsf{ACE\text{-}PRV\text{-}CCA}}_{\mathcal{E},\mathcal{A}_{\mathsf{PRV}}} + \mathsf{Adv}^{\mathsf{ACE\text{-}wANON\text{-}CCA}}_{\mathcal{E},\mathcal{A}_{\mathsf{wANON}}}. \qquad \square$$

We next show that the sender-anonymity no-read rule is implied by strong sender anonymity.

Lemma 6.4.9. *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. Further let* $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ *be a pair of probabilistic algorithms. Then, there exists an adversary* $\mathcal{A}_{\mathsf{sANON}}$ *(roughly as efficient as emulating an execution of* $\mathsf{Exp}^{\mathsf{ACE\text{-}no\text{-}read}}_{\mathcal{E}',\mathcal{A}'}$*) such that*

$$\mathsf{Adv}^{\mathsf{ACE\text{-}no\text{-}read,anon}}_{\mathcal{E}',\mathcal{A}} = \mathsf{Adv}^{\mathsf{ACE\text{-}sANON\text{-}CCA}}_{\mathcal{E},\mathcal{A}_{\mathsf{sANON}}}.$$

Proof. We construct $\mathcal{A}_{\mathsf{sANON}}$ as follows. On input sp, it emulates an execution of $\mathsf{Exp}^{\mathsf{ACE\text{-}no\text{-}read}}_{\mathcal{E}',\mathcal{A}}$, where the oracles $\mathcal{O}_G$ and $\mathcal{O}_E$ for $\mathcal{A}$ are emulated as in the proof of Lemma 6.4.8. When $\mathcal{A}_1$ returns (m_0, m_1, i_0, i_1, st), $\mathcal{A}_{\mathsf{sANON}}$ gives (m_0, m_1, i_0, i_1) to the challenger to obtain the ciphertext c^*. Then, $\mathcal{A}_2$ is invoked on input (st, c^*) and the oracles are emulated as before. When $\mathcal{A}_2$ terminates with output b', $\mathcal{A}_{\mathsf{sANON}}$ returns the same bit b'. We observe that the view $\mathcal{A}_{\mathsf{sANON}}$ emulates toward $\mathcal{A}$ is identical to the view of $\mathcal{A}$ in the experiment $\mathsf{Exp}^{\mathsf{ACE\text{-}no\text{-}read}}_{\mathcal{E}',\mathcal{A}}$. Thus,

$$\begin{aligned}
\mathsf{Adv}^{\mathsf{ACE\text{-}no\text{-}read,anon}}_{\mathcal{E}',\mathcal{A}} &= 2 \cdot \Pr^{\mathsf{Exp}^{\mathsf{ACE\text{-}no\text{-}read}}_{\mathcal{E}',\mathcal{A}}}\big[b' = b \ \wedge \ m_0 = m_1 \\
&\qquad\qquad \wedge \ \forall j \in J \ P(i_0, j) = P(i_1, j)\big] - 1 \\
&= 2 \cdot \Pr^{\mathsf{Exp}^{\mathsf{ACE\text{-}PRV\text{-}ANON\text{-}CCA}}_{\mathcal{E},\mathcal{A}_{\mathsf{sANON}}}}\big[b' = b \ \wedge \ m_0 = m_1 \\
&\qquad\qquad \wedge \ \forall j \in J \ P(i_0, j) = P(i_1, j)\big] - 1 \\
&= \mathsf{Adv}^{\mathsf{ACE\text{-}sANON\text{-}CCA}}_{\mathcal{E},\mathcal{A}_{\mathsf{sANON}}}.
\end{aligned} \qquad \square$$

To conclude the proof of Theorem 6.4.7, we prove the claim about the no-write with modification detection rule.

Lemma 6.4.10. *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. Then, there exist adversaries $\mathcal{A}_{\mathsf{SAN}}$, $\mathcal{A}_{\mathsf{RR}}$, and $\mathcal{A}_{\mathsf{CORR}}$ (all roughly as efficient as emulating an execution of $\mathsf{Exp}_{\mathcal{E},\mathcal{A}}^{\mathsf{ACE\text{-}MD\text{-}no\text{-}write}}$) such that for policies P where for all i, one can efficiently find some j with $P(i,j) = 1$,*

$$\mathsf{Adv}_{\mathcal{E},\mathcal{A}}^{\mathsf{ACE\text{-}MD\text{-}no\text{-}write}} \leq \mathsf{Adv}_{\mathcal{E},\mathcal{A}_{\mathsf{SAN}}}^{\mathsf{ACE\text{-}SAN\text{-}CCA}} + 4 \cdot \mathsf{Adv}_{\mathcal{E},\mathcal{A}_{\mathsf{RR}}}^{\mathsf{ACE\text{-}RR}} + 2 \cdot \mathsf{Adv}_{\mathcal{E},\mathcal{A}_{\mathsf{CORR}}}^{\mathsf{ACE\text{-}CORR}}.$$

Proof. We first construct the adversary $\mathcal{A}_{\mathsf{SAN}}$. When invoked on input sp, it gives sp to $\mathcal{A}_1$ and emulates $\mathsf{Exp}_{\mathcal{E},\mathcal{A}}^{\mathsf{ACE\text{-}MD\text{-}no\text{-}write}}$. The oracles for $\mathcal{A}$ are emulated as follows.

$\mathcal{O}_G(\cdot,\cdot)$: Relay queries to the oracle $\mathcal{O}_G$ of $\mathsf{Exp}_{\mathcal{E},\mathcal{A}_{\mathsf{SAN}}}^{\mathsf{ACE\text{-}SAN\text{-}CCA}}$.

$\mathcal{O}_{ES}(\cdot,\cdot)$: On query (j, m), $\mathcal{A}_{\mathsf{SAN}}$ queries $(j, \mathtt{sen})$ to its oracle $\mathcal{O}_G$ to receive the encryption key ek_j,[4] computes $c' \leftarrow \mathsf{San}(sp, \mathsf{Enc}(ek_j, m))$, and outputs c' to $\mathcal{A}$.

When $\mathcal{A}_1$ outputs (c_0, i', st), $\mathcal{A}_{\mathsf{SAN}}$ chooses a uniformly random message $m \twoheadleftarrow \mathcal{M}$, queries $(i', \mathtt{sen})$ to its oracle $\mathcal{O}_G$ to receive the encryption key $ek_{i'}$, and computes $c_1 \leftarrow \mathsf{Enc}(ek_{i'}, m)$. Then, $\mathcal{A}_{\mathsf{SAN}}$ gives (c_0, c_1) to the challenger to obtain a sanitized ciphertext c'_b. It then invokes $\mathcal{A}_2$ on input (st, c'_b) and emulates the oracles as above. When $\mathcal{A}_2$ outputs its guess b', $\mathcal{A}_{\mathsf{SAN}}$ outputs the same bit b' as its own guess. Note that the view $\mathcal{A}_{\mathsf{SAN}}$ emulates toward $\mathcal{A}$ is identical to the view of $\mathcal{A}$ in the experiment $\mathsf{Exp}_{\mathcal{E},\mathcal{A}}^{\mathsf{ACE\text{-}MD\text{-}no\text{-}write}}$. Let W_{noW} and W_{san} be the events that $\mathcal{A}$ wins in the no-write with modification detection experiment and $\mathcal{A}_{\mathsf{SAN}}$ wins the sanitization experiment, respectively, i.e.,

$$\begin{aligned} W_{\mathrm{noW}} &:= \big[b' = b \ \wedge \ \mathtt{dct} = \mathtt{false} \ \wedge \ i' \in I_1 \\ &\qquad\qquad \wedge \ \forall i \in I_1 \ \forall j \in J \ P(i,j) = 0 \ \wedge \ \mathsf{San}(sp, c_0) \neq \bot\big], \\ W_{\mathrm{san}} &:= \big[b' = b \ \wedge \ c'_0 \neq \bot \neq c'_1 \ \wedge \ \forall j \in J \ m_{0,j} = m_{1,j} = \bot\big]. \end{aligned}$$

[4]Looking ahead, we note that obtaining additional encryption keys is not problematic in the sanitization game, since the winning condition does not restrict the obtained encryption keys.

Further consider the events

$$\begin{aligned} C &:= [\mathsf{San}(sp, c_1) \neq \bot], \\ R &:= [\forall j \in J \; \mathsf{Dec}\big(\mathsf{Gen}(msk, j, \mathtt{rec}), \mathsf{San}(sp, c_0)\big) \\ &\qquad\qquad\qquad = \mathsf{Dec}\big(\mathsf{Gen}(msk, j, \mathtt{rec}), \mathsf{San}(sp, c_1)\big) = \bot]. \end{aligned}$$

We then have

$$\Pr^{\mathsf{Exp}^{\mathsf{ACE\text{-}SAN\text{-}CCA}}_{\mathcal{E},\mathcal{A}_{\mathsf{SAN}}}}[W_{\mathrm{san}}] \geq \Pr^{\mathsf{Exp}^{\mathsf{ACE\text{-}MD\text{-}no\text{-}write}}_{\mathcal{E},\mathcal{A}}}[W_{\mathrm{noW}} \cap C \cap R]. \tag{6.4}$$

We next show that the events $\neg C$ and $\neg R$ only occur with negligible probability if the ACE scheme is correct and role-respecting, respectively.

Claim 1. *There exists an adversary $\mathcal{A}_{\mathsf{CORR}}$ (roughly as efficient as emulating an execution of $\mathsf{Exp}^{\mathsf{ACE\text{-}MD\text{-}no\text{-}write}}_{\mathcal{E},\mathcal{A}}$) such that*

$$\Pr^{\mathsf{Exp}^{\mathsf{ACE\text{-}MD\text{-}no\text{-}write}}_{\mathcal{E},\mathcal{A}}}[\neg C] \leq \mathsf{Adv}^{\mathsf{ACE\text{-}CORR}}_{\mathcal{E},\mathcal{A}_{\mathsf{CORR}}}.$$

Proof of claim. On input sp, the adversary $\mathcal{A}_{\mathsf{CORR}}$ begins an emulation of $\mathsf{Exp}^{\mathsf{ACE\text{-}MD\text{-}no\text{-}write}}_{\mathcal{E},\mathcal{A}}$ as $\mathcal{A}_{\mathsf{SAN}}$ above. When $\mathcal{A}_1$ outputs (c_0, i', st), $\mathcal{A}_{\mathsf{CORR}}$ chooses a uniformly random message $m \twoheadleftarrow \mathcal{M}$ and finds j with $P(i', j) = 1$. It finally returns (m, i', j). By definition of Dec, we have $\mathsf{Dec}(\mathsf{Gen}(msk, j, \mathtt{rec}), \bot) = \bot$. Hence, if $\neg C$ occurs, then encrypting m for role i' and sanitizing and decrypting the result yields $\bot \neq m$. Therefore, $\mathcal{A}_{\mathsf{CORR}}$ wins the correctness game in this case, which implies the claim. ◊

Claim 2. *There exists an adversary $\mathcal{A}_{\mathsf{RR}}$ (roughly as efficient as emulating an execution of $\mathsf{Exp}^{\mathsf{ACE\text{-}MD\text{-}no\text{-}write}}_{\mathcal{E},\mathcal{A}}$) such that*

$$\Pr^{\mathsf{Exp}^{\mathsf{ACE\text{-}MD\text{-}no\text{-}write}}_{\mathcal{E},\mathcal{A}}}[W_{\mathrm{noW}} \cap C \cap \neg R] \leq 2 \cdot \mathsf{Adv}^{\mathsf{ACE\text{-}RR}}_{\mathcal{E},\mathcal{A}_{\mathsf{RR}}}.$$

Proof of claim. When invoked on input sp, $\mathcal{A}_{\mathsf{RR}}$ internally emulates an execution of $\mathcal{A}$ on input sp and emulates the oracles as follows.

$\mathcal{O}_G(\cdot,\cdot)$**:** Relay queries to the oracle $\mathcal{O}_G$ of $\mathsf{Exp}^{\mathsf{ACE\text{-}URR}}_{\mathcal{E},\mathcal{A}_{\mathsf{RR}}}$.

$\mathcal{O}_{ES}(\cdot,\cdot)$**:** On query (j, m), query (j, m) to $\mathcal{O}_E$ to receive the ciphertext c. Then, compute $c' \leftarrow \mathsf{San}(sp, c)$ and return c'.

When $\mathcal{A}_1$ outputs (c_0, i', st), $\mathcal{A}_{\mathsf{RR}}$ chooses a uniformly random message $m \twoheadleftarrow \mathcal{M}$, queries $(i', \mathtt{sen})$ to its oracle $\mathcal{O}_G$ to receive the encryption key $ek_{i'}$, and computes $c_1 \leftarrow \mathsf{Enc}(ek_{i'}, m)$. Then, $\mathcal{A}_{\mathsf{RR}}$ chooses $c \twoheadleftarrow \{c_0, c_1\}$ uniformly at random and outputs c to the challenger. Let W_{noW} be the event that $\mathcal{A}_{\mathsf{RR}}$ wins the role-respecting game, i.e.,

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

Note that W_{noW} and C imply that $c' \neq \bot$, $\mathtt{dct} = \mathtt{false}$, and $\forall i \in I\ \forall j \in J\ P(i,j) = 0$. Hence, if we additionally have $\neg R$, at least one of the two possible choices for c yield $m_j \neq \bot$ for some $j \in J$, and thus

$$\Pr^{\mathsf{Exp}^{\mathsf{ACE\text{-}MD\text{-}no\text{-}write}}_{\mathcal{E},\mathcal{A}}}[W_{\mathrm{noW}} \cap C \cap \neg R] \leq 2 \cdot \Pr^{\mathsf{Exp}^{\mathsf{ACE\text{-}URR}}_{\mathcal{E},\mathcal{A}_{\mathsf{RR}}}}[W_{\mathrm{RR}}] = 2 \cdot \mathsf{Adv}^{\mathsf{ACE\text{-}RR}}_{\mathcal{E},\mathcal{A}_{\mathsf{RR}}}. \ \Diamond$$

Combining equation (6.4) and Claims 1 and 2, we obtain

$$\begin{aligned}
&\Pr^{\mathsf{Exp}^{\mathsf{ACE\text{-}MD\text{-}no\text{-}write}}_{\mathcal{E},\mathcal{A}}}[W_{\mathrm{noW}}] \\
&\leq \Pr^{\mathsf{Exp}^{\mathsf{ACE\text{-}MD\text{-}no\text{-}write}}_{\mathcal{E},\mathcal{A}}}[W_{\mathrm{noW}} \cap C \cap R] \\
&\qquad + \Pr^{\mathsf{Exp}^{\mathsf{ACE\text{-}MD\text{-}no\text{-}write}}_{\mathcal{E},\mathcal{A}}}[W_{\mathrm{noW}} \cap C \cap \neg R] + \Pr^{\mathsf{Exp}^{\mathsf{ACE\text{-}MD\text{-}no\text{-}write}}_{\mathcal{E},\mathcal{A}}}[\neg C] \\
&\leq \Pr^{\mathsf{Exp}^{\mathsf{ACE\text{-}SAN\text{-}CCA}}_{\mathcal{E},\mathcal{A}_{\mathsf{SAN}}}}[W_{\mathrm{san}}] + 2 \cdot \mathsf{Adv}^{\mathsf{ACE\text{-}RR}}_{\mathcal{E},\mathcal{A}_{\mathsf{RR}}} + \mathsf{Adv}^{\mathsf{ACE\text{-}CORR}}_{\mathcal{E},\mathcal{A}_{\mathsf{CORR}}}.
\end{aligned}$$

We can thus conclude

$$\begin{aligned}
&\mathsf{Adv}^{\mathsf{ACE\text{-}MD\text{-}no\text{-}write}}_{\mathcal{E},\mathcal{A}} \\
&= 2 \cdot \Pr^{\mathsf{Exp}^{\mathsf{ACE\text{-}MD\text{-}no\text{-}write}}_{\mathcal{E},\mathcal{A}}}[W_{\mathrm{noW}}] - 1 \\
&\leq \underbrace{2 \cdot \Pr^{\mathsf{Exp}^{\mathsf{ACE\text{-}SAN\text{-}CCA}}_{\mathcal{E},\mathcal{A}_{\mathsf{SAN}}}}[W_{\mathrm{san}}] - 1}_{= \mathsf{Adv}^{\mathsf{ACE\text{-}SAN\text{-}CCA}}_{\mathcal{E},\mathcal{A}_{\mathsf{SAN}}}} + 4 \cdot \mathsf{Adv}^{\mathsf{ACE\text{-}RR}}_{\mathcal{E},\mathcal{A}_{\mathsf{RR}}} + 2 \cdot \mathsf{Adv}^{\mathsf{ACE\text{-}CORR}}_{\mathcal{E},\mathcal{A}_{\mathsf{CORR}}}. \quad \square
\end{aligned}$$

Relation to original no-write rule. Perhaps surprisingly, one can also show that our new notions imply the original no-write rule if DMod is symmetric in the sense that

$$\Pr[\mathsf{DMod}(sp, c_0, c_1) = 1] = \Pr[\mathsf{DMod}(sp, c_1, c_0) = 1],$$

which is the case for all schemes considered in this thesis. The proof idea is to construct adversaries against correctness, and sanitization and role-respecting security as above. Now, the role-respecting game is not won if the adversary $\mathcal{A}$ returns a ciphertext c_0 that is detected to be a modification of a ciphertext generated by $\mathcal{O}_{ES}$. We show that in this case, we can break sSAN-CCA security. Note that $\mathcal{O}_{ES}$ only gives $\mathcal{A}$ the sanitized ciphertexts. The proof idea is as follows. $\mathcal{A}_1$ makes several queries to $\mathcal{O}_{ES}$. For a uniformly chosen one, encrypt the message twice, give the resulting ciphertexts $\tilde{c}_0, \tilde{c}_1$ to the sSAN-CCA challenger, and give the obtained sanitized ciphertext $\tilde{c}'_b$ to $\mathcal{A}_1$. For all other queries, encrypt and sanitize the message normally. When $\mathcal{A}_1$ returns a ciphertext c_0, check whether c_0 is detected to be a modification of $\tilde{c}_0$ or $\tilde{c}_1$. Since the ciphertext $\tilde{c}_{1-b}$ is information-theoretically hidden from $\mathcal{A}$, it can be considered to be a fresh encryption. By our assumption, the probability that c_0 is detected to be a modification of $\tilde{c}_{1-b}$ is equal to the probability that $\tilde{c}_{1-b}$ is detected to be a modification of c_0, which contradicts non-detection of fresh encryptions. Hence, by checking which of the two ciphertexts is detected, one can guess b and thus break sSAN-CCA security. Note that $\mathcal{A}$ is allowed by the winning condition to obtain a decryption key that decrypts $\tilde{c}'_b$, which is why we need *strong* sanitization security.

Theorem 6.4.11. *Let* $\mathcal{E} = (\mathsf{Setup}, \mathsf{Gen}, \mathsf{Enc}, \mathsf{San}, \mathsf{Dec}, \mathsf{DMod})$ *be an ACE with modification detection scheme such that* $\Pr[\mathsf{DMod}(sp, c_0, c_1) = 1] = \Pr[\mathsf{DMod}(sp, c_1, c_0) = 1]$ *for all sp returned by* Setup *and all ciphertexts* $c_0, c_1 \in \mathcal{C}$*. Further let* $\mathcal{E}' = (\mathsf{Setup}, \mathsf{Gen}, \mathsf{Enc}, \mathsf{San}, \mathsf{Dec})$ *be the corresponding ACE scheme. If* $\mathcal{E}$ *is correct, detectable, has NDTCT-FENC, and is sSAN-CCA and RR secure, then* $\mathcal{E}'$ *satisfies the no-write rule for policies* P *such that for all* i*, one can efficiently find some* j *with* $P(i, j) = 1$*. More precisely, for all adversaries* $\mathcal{A}$ *that make at most* q_{ES} *queries to the oracle* $\mathcal{O}_{ES}$ *and at most* q_{dk} *queries of the form* $(\cdot, \mathtt{rec})$ *to* $\mathcal{O}_G$*, there exist adversaries* $\mathcal{A}_{\mathsf{SAN}}$*,* $\mathcal{A}_{\mathsf{RR}}$*,* $\mathcal{A}_{\mathsf{sSAN}}$*,* $\mathcal{A}_{\mathsf{NDTCT}}$*,* $\mathcal{A}_{\mathsf{CORR}}$*, and* $\mathcal{A}_{\mathsf{dtct}}$ *(all roughly as efficient as emulating an execution of* $\mathsf{Exp}^{\mathsf{ACE\text{-}no\text{-}write}}_{\mathcal{E},\mathcal{A}}$*) such that*

$$\begin{aligned}\mathsf{Adv}^{\mathsf{ACE\text{-}no\text{-}write}}_{\mathcal{E}',\mathcal{A}} \leq \mathsf{Adv}^{\mathsf{ACE\text{-}SAN\text{-}CCA}}_{\mathcal{E},\mathcal{A}_{\mathsf{SAN}}} + 4 \cdot \mathsf{Adv}^{\mathsf{ACE\text{-}RR}}_{\mathcal{E},\mathcal{A}_{\mathsf{RR}}} + 2q_{ES} \cdot \mathsf{Adv}^{\mathsf{ACE\text{-}sSAN\text{-}CCA}}_{\mathcal{E},\mathcal{A}_{\mathsf{sSAN}}} \\ + 4q_{ES} \cdot \mathsf{Adv}^{\mathsf{ACE\text{-}NDTCT\text{-}FENC}}_{\mathcal{E},\mathcal{A}_{\mathsf{NDTCT}}} + (8q_{ES}q_{dk} + 2) \cdot \mathsf{Adv}^{\mathsf{ACE\text{-}CORR}}_{\mathcal{E},\mathcal{A}_{\mathsf{CORR}}} \\ + 8q_{ES}q_{dk} \cdot \mathsf{Adv}^{\mathsf{ACE\text{-}DTCT}}_{\mathcal{E},\mathcal{A}_{\mathsf{dtct}}}.\end{aligned}$$

Proof. Let $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ be an adversary and consider $\mathsf{Exp}^{\mathsf{ACE\text{-}MD\text{-}no\text{-}write}}_{\mathcal{E},\mathcal{A}}$. Let W_{noW} and $W_{\mathrm{MD\text{-}noW}}$ be the events that $\mathcal{A}$ wins the no-write and no-write with modification detection game, respectively. Note that we have $W_{\text{MD-noW}} = W_{\text{noW}} \cap [\texttt{dct} = \texttt{false}]$. Hence,

$$\begin{aligned}\mathsf{Adv}^{\text{ACE-no-write}}_{\mathcal{E}',\mathcal{A}} &= 2 \cdot \Pr^{\mathsf{Exp}^{\text{ACE-MD-no-write}}_{\mathcal{E},\mathcal{A}}}[W_{\text{noW}}] - 1 \\ &= 2 \cdot \Big(\Pr^{\mathsf{Exp}^{\text{ACE-MD-no-write}}_{\mathcal{E},\mathcal{A}}}[W_{\text{MD-noW}}] \\ &\qquad + \Pr^{\mathsf{Exp}^{\text{ACE-MD-no-write}}_{\mathcal{E},\mathcal{A}}}\big[W_{\text{noW}} \cap [\texttt{dct} = \texttt{true}]\big]\Big) - 1 \\ &\leq \mathsf{Adv}^{\text{ACE-MD-no-write}}_{\mathcal{E},\mathcal{A}} + 2 \cdot \Pr^{\mathsf{Exp}^{\text{ACE-MD-no-write}}_{\mathcal{E},\mathcal{A}}}[\texttt{dct} = \texttt{true}].\end{aligned}$$

Lemma 6.4.10 implies that there exist adversaries $\mathcal{A}_{\mathsf{SAN}}$, $\mathcal{A}_{\mathsf{RR}}$, and $\mathcal{A}'_{\mathsf{CORR}}$ (all roughly as efficient as emulating an execution of $\mathsf{Exp}^{\text{ACE-MD-no-write}}_{\mathcal{E},\mathcal{A}}$) such that

$$\begin{aligned}\mathsf{Adv}^{\text{ACE-no-write}}_{\mathcal{E}',\mathcal{A}} &\leq \mathsf{Adv}^{\text{ACE-SAN-CCA}}_{\mathcal{E},\mathcal{A}_{\mathsf{SAN}}} + 4 \cdot \mathsf{Adv}^{\text{ACE-RR}}_{\mathcal{E},\mathcal{A}_{\mathsf{RR}}} + 2 \cdot \mathsf{Adv}^{\text{ACE-CORR}}_{\mathcal{E},\mathcal{A}'_{\mathsf{CORR}}} \\ &\quad + 2 \cdot \Pr^{\mathsf{Exp}^{\text{ACE-MD-no-write}}_{\mathcal{E},\mathcal{A}}}[\texttt{dct} = \texttt{true}].\end{aligned} \tag{6.5}$$

To bound the probability of $[\texttt{dct} = \texttt{true}]$, we construct the adversary $\mathcal{A}_{\mathsf{sSAN}}$. When invoked on input sp, it first chooses $q_0 \twoheadleftarrow \{1, \ldots, q_{ES}\}$ uniformly at random, sets $k \leftarrow 1$ and internally emulates an execution of $\mathcal{A}$ on input sp. Oracle queries by $\mathcal{A}$ are answered as follows:

$\mathcal{O}_G(\cdot,\cdot)$: Relay queries to the oracle $\mathcal{O}_G$ of $\mathsf{Exp}^{\text{ACE-SAN-CCA}}_{\mathcal{E},\mathcal{A}_{\mathsf{sSAN}}}$.

$\mathcal{O}_{ES}(\cdot,\cdot)$: On query (i, m), if $k \neq q_0$, $\mathcal{A}_{\mathsf{sSAN}}$ queries $(i, \texttt{sen})$ to its oracle $\mathcal{O}_G$ to receive the encryption key ek_i. It then computes $c' \leftarrow \mathsf{San}(sp, \mathsf{Enc}(ek_i, m))$ and outputs c' to $\mathcal{A}$. Finally, it sets $k \leftarrow k + 1$.

If $k = q_0$, then $\mathcal{A}_{\mathsf{sSAN}}$ queries $(i, \texttt{sen})$ to its oracle $\mathcal{O}_G$ to receive the encryption key ek_i. It then creates two independent encryptions of m by computing $\tilde{c}_0 \leftarrow \mathsf{Enc}(ek_i, m)$ and $\tilde{c}_1 \leftarrow \mathsf{Enc}(ek_i, m)$, sets $i_{q_0} \leftarrow i$, $m_{q_0} \leftarrow m$, $k \leftarrow k + 1$, and gives $\tilde{c}_0, \tilde{c}_1$ to the challenger to obtain $\tilde{c}'_b$.

If $\mathcal{A}_1$ terminates before $k = q_0$ is reached, $\mathcal{A}_{\mathsf{sSAN}}$ gives two fresh encryptions of some fixed message m_{q_0} for a fixed role i_{q_0} to the challenger and then returns a uniform bit $b' \twoheadleftarrow \{0, 1\}$. Otherwise, when $\mathcal{A}_1$ returns i' and c_0, $\mathcal{A}_{\mathsf{sSAN}}$ evaluates $d_0 \leftarrow \mathsf{DMod}(sp, \tilde{c}_0, c_0)$ and $d_1 \leftarrow \mathsf{DMod}(sp, \tilde{c}_1, c_0)$. If $d_0 = d_1$, then $\mathcal{A}_{\mathsf{sSAN}}$ also returns a uniform bit; if $d_{b'} = 1$ for exactly one $b' \in \{0, 1\}$, $\mathcal{A}_{\mathsf{sSAN}}$ returns b'.

Let Q be the event that $d_0 = 1$ or $d_1 = 1$ and let D be the event that $d_{1-b} = 1$. Note that if Q and $\neg D$ occur, $\mathcal{A}_{\mathsf{sSAN}}$ returns the correct bit $b' = b$. Moreover, if Q does not occur, $\mathcal{A}_{\mathsf{sSAN}}$ returns a uniform bit. Hence,

$$\begin{aligned}\Pr[b' = b] &= \Pr\big[[b' = b] \cap Q \cap \neg D\big] + \Pr\big[[b' = b] \cap \neg(Q \cap \neg D)\big] \\ &\geq \Pr[Q \cap \neg D] + \Pr\big[[b' = b] \cap \neg Q\big] \\ &= \Pr[Q \cap \neg D] + \Pr[b' = b \mid \neg Q] \cdot \Pr[\neg Q] \\ &= \Pr[Q \cap \neg D] + \tfrac{1}{2} \cdot (1 - \Pr[Q]),\end{aligned}$$

where all probabilities are in $\mathsf{Exp}_{\mathcal{E},\mathcal{A}_{\mathsf{sSAN}}}^{\mathsf{ACE\text{-}SAN\text{-}CCA}}$. This implies

$$\Pr[Q] = \Pr[Q \cap D] + \Pr[Q \cap \neg D] \leq \Pr[D] + \Pr[b' = b] - \tfrac{1}{2} + \tfrac{1}{2}\Pr[Q]$$

and therefore

$$\Pr[Q] \leq 2 \cdot \Pr[D] + 2 \cdot \Pr[b' = b] - 1.$$

Note that if $[\mathtt{dct} = \mathtt{true}]$ occurs in $\mathsf{Exp}_{\mathcal{E},\mathcal{A}}^{\mathsf{ACE\text{-}MD\text{-}no\text{-}write}}$, then Q occurs in $\mathsf{Exp}_{\mathcal{E},\mathcal{A}_{\mathsf{sSAN}}}^{\mathsf{ACE\text{-}SAN\text{-}CCA}}$ with probability at least $1/q_{ES}$. We can therefore conclude that

$$\begin{aligned}\Pr^{\mathsf{Exp}_{\mathcal{E},\mathcal{A}}^{\mathsf{ACE\text{-}MD\text{-}no\text{-}write}}}[\mathtt{dct} = \mathtt{true}] &\leq q_{ES} \cdot \Pr^{\mathsf{Exp}_{\mathcal{E},\mathcal{A}_{\mathsf{sSAN}}}^{\mathsf{ACE\text{-}SAN\text{-}CCA}}}[Q] \\ &\leq 2q_{ES} \cdot \Pr^{\mathsf{Exp}_{\mathcal{E},\mathcal{A}_{\mathsf{sSAN}}}^{\mathsf{ACE\text{-}SAN\text{-}CCA}}}[D] \\ &\qquad + q_{ES}\Big(2 \cdot \Pr^{\mathsf{Exp}_{\mathcal{E},\mathcal{A}_{\mathsf{sSAN}}}^{\mathsf{ACE\text{-}SAN\text{-}CCA}}}[b' = b] - 1\Big).\end{aligned}$$

Let W_{sSAN} be the event that $\mathcal{A}_{\mathsf{sSAN}}$ wins the strong sanitization game and consider the events

$$\begin{aligned}C &:= \big[\tilde{c}_0' \neq \bot \neq \tilde{c}_1' \ \wedge\ \forall j \in J\ (P(i_{q_0}, j) = 1 \rightarrow m_{0,j} = m_{1,j} = m_{q_0})\big], \\ R &:= \big[\forall j \in J\ (P(i_{q_0}, j) = 0 \rightarrow m_{0,j} = m_{1,j} = \bot)\big].\end{aligned}$$

We then have that $[b' = b]$, C, and R together imply W_{sSAN}. Thus,

$$\begin{aligned}&\Pr^{\mathsf{Exp}_{\mathcal{E},\mathcal{A}_{\mathsf{sSAN}}}^{\mathsf{ACE\text{-}SAN\text{-}CCA}}}[b' = b] \\ &= \Pr^{\mathsf{Exp}_{\mathcal{E},\mathcal{A}_{\mathsf{sSAN}}}^{\mathsf{ACE\text{-}SAN\text{-}CCA}}}\big[[b' = b] \cap C \cap R\big] + \Pr^{\mathsf{Exp}_{\mathcal{E},\mathcal{A}_{\mathsf{sSAN}}}^{\mathsf{ACE\text{-}SAN\text{-}CCA}}}\big[[b' = b] \cap \neg(C \cap R)\big] \\ &\leq \Pr^{\mathsf{Exp}_{\mathcal{E},\mathcal{A}_{\mathsf{sSAN}}}^{\mathsf{ACE\text{-}SAN\text{-}CCA}}}[W_{\text{sSAN}}] + \Pr^{\mathsf{Exp}_{\mathcal{E},\mathcal{A}_{\mathsf{sSAN}}}^{\mathsf{ACE\text{-}SAN\text{-}CCA}}}[\neg C \cup \neg R].\end{aligned}$$

Together with the previous result, this yields

$$\begin{aligned}
&\Pr^{\mathsf{Exp}^{\textsf{ACE-MD-no-write}}_{\mathcal{E},\mathcal{A}}}[\texttt{dct} = \texttt{true}] \\
&\le 2q_{ES} \cdot \Pr^{\mathsf{Exp}^{\textsf{ACE-SAN-CCA}}_{\mathcal{E},\mathcal{A}_{\textsf{sSAN}}}}[D] + q_{ES}\Big[2 \cdot \Pr^{\mathsf{Exp}^{\textsf{ACE-SAN-CCA}}_{\mathcal{E},\mathcal{A}_{\textsf{sSAN}}}}[W_{\text{sSAN}}] - 1 \\
&\qquad\qquad + 2 \cdot \Pr^{\mathsf{Exp}^{\textsf{ACE-SAN-CCA}}_{\mathcal{E},\mathcal{A}_{\textsf{sSAN}}}}[\neg C \cup \neg R]\Big] \\
&= 2q_{ES} \cdot \Pr^{\mathsf{Exp}^{\textsf{ACE-SAN-CCA}}_{\mathcal{E},\mathcal{A}_{\textsf{sSAN}}}}[D] + q_{ES} \cdot \mathsf{Adv}^{\textsf{ACE-sSAN-CCA}}_{\mathcal{E},\mathcal{A}_{\textsf{sSAN}}} \\
&\qquad\qquad + 2q_{ES} \cdot \Pr^{\mathsf{Exp}^{\textsf{ACE-SAN-CCA}}_{\mathcal{E},\mathcal{A}_{\textsf{sSAN}}}}[\neg C \cup \neg R].
\end{aligned}$$

Now consider the adversary $\mathcal{A}_{\textsf{NDTCT}}$ that emulates $\mathsf{Exp}^{\textsf{ACE-SAN-CCA}}_{\mathcal{E},\mathcal{A}_{\textsf{sSAN}}}$ and outputs (m_{q_0}, i_{q_0}, c_0). Note that the view of $\mathcal{A}_1$ in the emulation is independent of $\tilde{c}_{1-b}$. One can therefore assume that $\tilde{c}_{1-b}$ is generated after $\mathcal{A}_1$ outputs c_0, as c^* in $\mathsf{Exp}^{\textsf{ACE-NDTCT-FENC}}_{\mathcal{E},\mathcal{A}_{\textsf{NDTCT}}}$. By assumption, we have $\Pr[\mathsf{DMod}(sp, \tilde{c}_{1-b}, c_0) = 1] = \Pr[\mathsf{DMod}(sp, c_0, \tilde{c}_{1-b}) = 1]$, and therefore

$$\Pr^{\mathsf{Exp}^{\textsf{ACE-SAN-CCA}}_{\mathcal{E},\mathcal{A}_{\textsf{sSAN}}}}[D] = \mathsf{Adv}^{\textsf{ACE-NDTCT-FENC}}_{\mathcal{E},\mathcal{A}_{\textsf{NDTCT}}}.$$

Further consider $\mathcal{A}''_{\textsf{CORR}}$ that emulates $\mathsf{Exp}^{\textsf{ACE-SAN-CCA}}_{\mathcal{E},\mathcal{A}_{\textsf{sSAN}}}$ and if there exists $j \in J$ with $P(i_{q_0}, j) = 1$, it chooses $j \twoheadleftarrow J$ uniformly at random and outputs (m_{q_0}, i_{q_0}, j). If such a $j \in J$ does not exist, $\mathcal{A}''_{\textsf{CORR}}$ finds $j \in [n]$ with $P(i_{q_0}, j) = 1$ and then outputs (m_{q_0}, i_{q_0}, j). Note that $m_{0,j} = m_{1,j} = m_{q_0}$ implies $\tilde{c}'_0 \neq \bot \neq \tilde{c}'_1$ since $\bot$ decrypts to $\bot$. Hence, if such a $j \in J$ exists and $\neg C$ occurs, $\mathcal{A}''_{\textsf{CORR}}$ wins the correctness game with probability at least $1/(2|J|) \geq 1/(2q_{dk})$; and if no such $j \in J$ exists and $\neg C$ occurs, $\mathcal{A}''_{\textsf{CORR}}$ wins with probability at least $1/2$. The factor $1/2$ is due to the fact that the message is encrypted twice in $\mathsf{Exp}^{\textsf{ACE-SAN-CCA}}_{\mathcal{E},\mathcal{A}_{\textsf{sSAN}}}$ but only once in the correctness experiment. Overall, we get

$$\Pr^{\mathsf{Exp}^{\textsf{ACE-SAN-CCA}}_{\mathcal{E},\mathcal{A}_{\textsf{sSAN}}}}[\neg C] \leq 2q_{dk} \cdot \mathsf{Adv}^{\textsf{ACE-CORR}}_{\mathcal{E},\mathcal{A}''_{\textsf{CORR}}}.$$

Finally consider $\mathcal{A}_{\textsf{dtct}}$ that emulates $\mathsf{Exp}^{\textsf{ACE-SAN-CCA}}_{\mathcal{E},\mathcal{A}_{\textsf{sSAN}}}$, chooses $j \twoheadleftarrow J$ uniformly at random and outputs (m_{q_0}, i_{q_0}, j). If $\neg R$ occurs, $\mathcal{A}_{\textsf{dtct}}$ wins the detectability game with probability at least $1/(2q_{dk})$. Hence,

$$\Pr^{\mathsf{Exp}^{\textsf{ACE-SAN-CCA}}_{\mathcal{E},\mathcal{A}_{\textsf{sSAN}}}}[\neg R] \leq 2q_{dk} \cdot \mathsf{Adv}^{\textsf{ACE-DTCT}}_{\mathcal{E},\mathcal{A}_{\textsf{dtct}}}.$$

Combining our results, we obtain

$$\Pr^{\mathsf{Exp}^{\mathsf{ACE\text{-}MD\text{-}no\text{-}write}}_{\mathcal{E},\mathcal{A}}}[\mathtt{dct} = \mathtt{true}] \leq 2q_{ES} \cdot \mathsf{Adv}^{\mathsf{ACE\text{-}NDTCT\text{-}FENC}}_{\mathcal{E},\mathcal{A}_{\mathsf{NDTCT}}} \\ + q_{ES} \cdot \mathsf{Adv}^{\mathsf{ACE\text{-}sSAN\text{-}CCA}}_{\mathcal{E},\mathcal{A}_{\mathsf{sSAN}}} + 4q_{ES}q_{dk} \cdot \Big(\mathsf{Adv}^{\mathsf{ACE\text{-}CORR}}_{\mathcal{E},\mathcal{A}''_{\mathsf{CORR}}} + \mathsf{Adv}^{\mathsf{ACE\text{-}DTCT}}_{\mathcal{E},\mathcal{A}_{\mathsf{dtct}}}\Big).$$

Together with equation (6.5), this yields

$$\mathsf{Adv}^{\mathsf{ACE\text{-}no\text{-}write}}_{\mathcal{E}',\mathcal{A}} \leq \mathsf{Adv}^{\mathsf{ACE\text{-}SAN\text{-}CCA}}_{\mathcal{E},\mathcal{A}_{\mathsf{SAN}}} + 4 \cdot \mathsf{Adv}^{\mathsf{ACE\text{-}RR}}_{\mathcal{E},\mathcal{A}_{\mathsf{RR}}} + 2 \cdot \mathsf{Adv}^{\mathsf{ACE\text{-}CORR}}_{\mathcal{E},\mathcal{A}'_{\mathsf{CORR}}} \\ + 4q_{ES} \cdot \mathsf{Adv}^{\mathsf{ACE\text{-}NDTCT\text{-}FENC}}_{\mathcal{E},\mathcal{A}_{\mathsf{NDTCT}}} + 2q_{ES} \cdot \mathsf{Adv}^{\mathsf{ACE\text{-}sSAN\text{-}CCA}}_{\mathcal{E},\mathcal{A}_{\mathsf{sSAN}}} \\ + 8q_{ES}q_{dk} \cdot \mathsf{Adv}^{\mathsf{ACE\text{-}CORR}}_{\mathcal{E},\mathcal{A}''_{\mathsf{CORR}}} + 8q_{ES}q_{dk} \cdot \mathsf{Adv}^{\mathsf{ACE\text{-}DTCT}}_{\mathcal{E},\mathcal{A}_{\mathsf{dtct}}}.$$

For the adversary $\mathcal{A}_{\mathsf{CORR}}$ that runs $\mathcal{A}'_{\mathsf{CORR}}$ with probability $\frac{2}{8q_{ES}q_{dk}+2}$ and $\mathcal{A}''_{\mathsf{CORR}}$ with probability $\frac{8q_{ES}q_{dk}}{8q_{ES}q_{dk}+2}$, we have $(8q_{ES}q_{dk} + 2) \cdot \mathsf{Adv}^{\mathsf{ACE\text{-}CORR}}_{\mathcal{E},\mathcal{A}_{\mathsf{CORR}}} = 2 \cdot \mathsf{Adv}^{\mathsf{ACE\text{-}CORR}}_{\mathcal{E},\mathcal{A}'_{\mathsf{CORR}}} + 8q_{ES}q_{dk} \cdot \mathsf{Adv}^{\mathsf{ACE\text{-}CORR}}_{\mathcal{E},\mathcal{A}''_{\mathsf{CORR}}}$ and the claim of the theorem follows. □

6.5 Enhanced Sanitizable PKE

6.5.1 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 6.5.1. 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 *sanitizer parameters sp*, and a *master secret key msk*. We implicitly assume that all parameters and keys include the finite *message space* $\mathcal{M}$ and the *ciphertext spaces* $\mathcal{C}, \mathcal{C}'$.

Key generation: The algorithm Gen on input a master secret key msk, outputs an *encryption key* ek and a *decryption key* dk.

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

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

Decryption: 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 negligible 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 6.5.2. For an sPKE scheme $\mathcal{E} = (\mathsf{Setup}, \mathsf{Gen}, \mathsf{Enc}, \mathsf{San}, \mathsf{Dec})$ and 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 *(unrestricted strong)*

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

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

Exper. $\mathsf{Exp}^{\mathsf{sPKE\text{-}UPD\text{-}CTXT}}_{\mathcal{E},\mathcal{A}}$

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

Figure 6.3: Security experiments $\mathsf{Exp}^{\mathsf{sPKE\text{-}IND\text{-}CCA}}_{\mathcal{E},\mathcal{A}}$ and $\mathsf{Exp}^{\mathsf{sPKE\text{-}UPD\text{-}CTXT}}_{\mathcal{E},\mathcal{A}}$ for an sPKE scheme $\mathcal{E}$ and an adversary $\mathcal{A}$, where $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ in the former experiment. The oracle $\mathcal{O}_G$ on input `getNew`, outputs a fresh key pair $(ek, dk) \leftarrow \mathsf{Gen}(msk)$, and the oracle $\mathcal{O}_{SD}$ is defined as $\mathcal{O}_{SD}(c) = \mathsf{Dec}(dk, \mathsf{San}(sp, c))$.

robustness advantage of $\mathcal{A}$ as

$$\begin{aligned}\mathsf{Adv}^{\mathsf{sPKE\text{-}USROB}}_{\mathcal{E},\mathcal{A}} := \Pr\big[&1 \le i_0, i_1 \le q \;\wedge\; i_0 \ne i_1 \\ &\wedge\; \mathsf{Dec}\big(dk_{i_0}, \mathsf{San}(sp, c)\big) \ne \bot \ne \mathsf{Dec}\big(dk_{i_1}, \mathsf{San}(sp, c)\big)\big],\end{aligned}$$

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 say the scheme $\mathcal{E}$ is *(unrestricted strongly) robust (USROB secure)* if $\mathsf{Adv}^{\mathsf{sPKE\text{-}USROB}}_{\mathcal{E},\mathcal{A}}$ is negligible for all efficient $\mathcal{A}$.

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

Definition 6.5.3. For an sPKE scheme $\mathcal{E} = (\mathsf{Setup}, \mathsf{Gen}, \mathsf{Enc}, \mathsf{San}, \mathsf{Dec})$ and a pair of probabilistic algorithms $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, consider the experiment $\mathsf{Exp}^{\mathsf{sPKE\text{-}IND\text{-}CCA}}_{\mathcal{E},\mathcal{A}}$ in Figure 6.3 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

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

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

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

Figure 6.4: The anonymity security experiment for an sPKE scheme $\mathcal{E}$ and an adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$. The oracle $\mathcal{O}_G$ on input `getNew`, outputs a fresh key pair $(ek, dk) \leftarrow \mathsf{Gen}(msk)$, and the oracle $\mathcal{O}_{SD_j}$ is defined as $\mathcal{O}_{SD_j}(c) = \mathsf{Dec}(dk_j, \mathsf{San}(sp, c))$.

where the probability is over the randomness in $\mathsf{Exp}^{\mathsf{sPKE\text{-}IND\text{-}CCA}}_{\mathcal{E},\mathcal{A}}$. The scheme $\mathcal{E}$ has *indistinguishable ciphertexts under chosen-ciphertext attacks (is IND-CCA secure)* if $\mathsf{Adv}^{\mathsf{sPKE\text{-}IND\text{-}CCA}}_{\mathcal{E},\mathcal{A}}$ is negligible for all efficient $\mathcal{A}$.

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 here obtains the decryption key.

Definition 6.5.4. For an sPKE scheme $\mathcal{E} = (\mathsf{Setup}, \mathsf{Gen}, \mathsf{Enc}, \mathsf{San}, \mathsf{Dec})$ and probabilistic algorithm $\mathcal{A}$, consider the experiment $\mathsf{Exp}^{\mathsf{sPKE\text{-}UPD\text{-}CTXT}}_{\mathcal{E},\mathcal{A}}$ in Figure 6.3. We define the *ciphertext unpredictability advantage* of $\mathcal{A}$ as

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

where the probability is over the randomness in $\mathsf{Exp}^{\mathsf{sPKE\text{-}UPD\text{-}CTXT}}_{\mathcal{E},\mathcal{A}}$. We say the scheme $\mathcal{E}$ has *unpredictable ciphertexts (is UPD-CTXT secure)* if $\mathsf{Adv}^{\mathsf{sPKE\text{-}UPD\text{-}CTXT}}_{\mathcal{E},\mathcal{A}}$ is negligible for all efficient $\mathcal{A}$.

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

Definition 6.5.5. For an sPKE scheme $\mathcal{E} = (\mathsf{Setup}, \mathsf{Gen}, \mathsf{Enc}, \mathsf{San}, \mathsf{Dec})$ and a pair of probabilistic algorithms $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, consider the experiment $\mathsf{Exp}^{\mathsf{sPKE\text{-}IK\text{-}CCA}}_{\mathcal{E},\mathcal{A}}$ in Figure 6.4 and let $C_{\mathcal{A}_2}$ be the set of all ciphertexts

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

```
**Input:** $1^\kappa, \kappa \in \mathbb{N}$
    $(sp, msk) \leftarrow \mathsf{Setup}(1^\kappa)$
    $(ek_0, dk_0) \leftarrow \mathsf{Gen}(msk)$; $(ek_1, dk_1) \leftarrow \mathsf{Gen}(msk)$
    $(c_0, c_1, st) \leftarrow \mathcal{A}_1^{\mathcal{O}_G(\cdot), \mathcal{O}_{SD_0}(\cdot), \mathcal{O}_{SD_1}(\cdot)}(sp, ek_0, ek_1)$
    $c_0' \leftarrow \mathsf{San}(sp, c_0)$; $c_1' \leftarrow \mathsf{San}(sp, c_1)$
    $m_{0,0} \leftarrow \mathsf{Dec}(dk_0, c_0')$; $m_{0,1} \leftarrow \mathsf{Dec}(dk_1, c_0')$
    $m_{1,0} \leftarrow \mathsf{Dec}(dk_0, c_1')$; $m_{1,1} \leftarrow \mathsf{Dec}(dk_1, c_1')$
    $b \twoheadleftarrow \{0,1\}$
    $b' \leftarrow \mathcal{A}_2^{\mathcal{O}_G(\cdot), \mathcal{O}_{SD_0}(\cdot), \mathcal{O}_{SD_1}(\cdot)}(st, c_b')$
```

Figure 6.5: Sanitization security experiment for an sPKE scheme $\mathcal{E}$ and an adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$. The oracle $\mathcal{O}_G$ on input `getNew`, outputs a fresh key pair $(ek, dk) \leftarrow \mathsf{Gen}(msk)$, and the oracle $\mathcal{O}_{SD_j}$ is defined as $\mathcal{O}_{SD_j}(c) = \mathsf{Dec}(dk_j, \mathsf{San}(sp, c))$.

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

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

where the probability is over the randomness in $\mathsf{Exp}^{\text{sPKE-IK-CCA}}_{\mathcal{E},\mathcal{A}}$. The scheme $\mathcal{E}$ has *indistinguishable keys under chosen-ciphertext attacks (is IK-CCA secure)* if $\mathsf{Adv}^{\text{sPKE-IK-CCA}}_{\mathcal{E},\mathcal{A}}$ is negligible for all efficient $\mathcal{A}$.

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 are encryptions relative to the two challenge public keys ek_0 and ek_1. Otherwise, the adversary could use the oracle $\mathcal{O}_G$ to obtain a fresh key-pair (ek, dk) and encrypt two different messages under ek. It could then decrypt the challenge ciphertext using dk and win the game.

Definition 6.5.6. For an sPKE scheme $\mathcal{E} = (\mathsf{Setup}, \mathsf{Gen}, \mathsf{Enc}, \mathsf{San}, \mathsf{Dec})$ and a pair of probabilistic algorithms $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, consider the experiment $\mathsf{Exp}^{\text{sPKE-SAN-CCA}}_{\mathcal{E},\mathcal{A}}$ in Figure 6.5. We define the *sanitization under chosen-ciphertext attacks advantage* of $\mathcal{A}$ as

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

where the probability is over the randomness in $\mathsf{Exp}_{\mathcal{E},\mathcal{A}}^{\mathsf{sPKE\text{-}IK\text{-}CCA}}$. The scheme $\mathcal{E}$ is *sanitization under chosen-ciphertext attacks (SAN-CCA) secure* if $\mathsf{Adv}_{\mathcal{E},\mathcal{A}}^{\mathsf{sPKE\text{-}SAN\text{-}CCA}}$ is negligible for all efficient $\mathcal{A}$.

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 obtain a new key pair from $\mathcal{O}_G$ 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 6.5.7. For an sPKE scheme $\mathcal{E} = (\mathsf{Setup}, \mathsf{Gen}, \mathsf{Enc}, \mathsf{San}, \mathsf{Dec})$, we define the *encryption-key collision probability* $\mathsf{Col}_{\mathcal{E}}^{\mathsf{ek}}$ as the probability of $[ek_0 = ek_1]$ in the experiment that runs $(sp, msk) \leftarrow \mathsf{Setup}(1^\kappa)$ and then $(ek_0, dk_0) \leftarrow \mathsf{Gen}(msk)$ and $(ek_1, dk_1) \leftarrow \mathsf{Gen}(msk)$ with independent random coins.

6.5.2 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 use a variant of ElGamal encryption and 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 one-time simulation sound by Sahai [Sah99].

Let PKE be a (IND-CPA-secure) public-key encryption scheme, let Sig be a (EUF-CMA-secure) signature scheme, and let NIZK be a (one-time simulation sound) NIZK proof system for the language $L := \{x \mid \exists w \ (x, w) \in R\}$, where the relation R is defined as follows: for $x = \big(g, ek^{\mathsf{PKE}}, vk^{\mathsf{Sig}}, c_1, c_2, c_\sigma\big)$ and $w = (m, g^a, g^b, r_1, s_1, r_2, s_2, \sigma, r)$, we have $(x, w) \in R$ if and only if

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

We define an sPKE scheme as follows:

Setup: The setup algorithm $\mathsf{sPKE.Setup}$ first generates

$$\begin{aligned}\big(ek^{\mathsf{PKE}}, dk^{\mathsf{PKE}}\big) &\leftarrow \mathsf{PKE.Gen}(1^\kappa),\\ \big(vk^{\mathsf{Sig}}, sk^{\mathsf{Sig}}\big) &\leftarrow \mathsf{Sig.Gen}(1^\kappa),\\ crs &\leftarrow \mathsf{NIZK.Gen}(1^\kappa).\end{aligned}$$

Let $G = \langle g \rangle$ be a cyclic group with prime order p generated by g, with $p \geq 2^\kappa$, and let $\mathcal{M} \subseteq G$ such that $|\mathcal{M}|/p \leq 2^{-\kappa}$. The sanitizer parameters sp^{sPKE} contain ek^{PKE}, vk^{Sig}, crs, and a description of G, including g and p. The master secret key msk^{sPKE} consists of ek^{PKE}, vk^{Sig}, sk^{Sig}, crs, and a description of G, including g and p.

Key generation: The algorithm $\mathsf{sPKE.Gen}$ on input msk^{sPKE}, samples two elements $dk_1, dk_2 \twoheadleftarrow \mathbb{Z}_p^*$ and computes $ek_1 \leftarrow g^{dk_1}$, $ek_2 \leftarrow g^{dk_2}$, as well as $\sigma \leftarrow \mathsf{Sig.Sign}\big(sk^{\mathsf{Sig}}, (ek_1, ek_2)\big)$. Finally, it outputs $ek^{\mathsf{sPKE}} := \big(g, p, crs, ek^{\mathsf{PKE}}, vk^{\mathsf{Sig}}, ek_1, ek_2, \sigma\big)$ and $dk^{\mathsf{sPKE}} := (dk_1, dk_2)$.

Encryption: On input an encryption key $ek^{\mathsf{sPKE}} = \big(g, p, crs, ek^{\mathsf{PKE}}, vk^{\mathsf{Sig}}, ek_1, ek_2, \sigma\big)$ and a message $m \in \mathcal{M}$, the algorithm $\mathsf{sPKE.Enc}$ samples randomness r, chooses $r_1, s_1, r_2, s_2 \twoheadleftarrow \mathbb{Z}_p^*$ uniformly at random, and computes

$$\begin{aligned}c_1 &\leftarrow \big(g^{r_1}, ek_1^{r_1}, g^{s_1}, ek_1^{s_1} \cdot m\big),\\ c_2 &\leftarrow \big(g^{r_2}, ek_2^{r_2}, g^{s_2}, ek_2^{s_2} \cdot m\big),\\ c_\sigma &\leftarrow \mathsf{PKE.Enc}\big(ek^{\mathsf{PKE}}, (ek_1, ek_2, \sigma); r\big).\end{aligned}$$

It generates $\pi \leftarrow \mathsf{NIZK.Prove}\big(crs, x := (g, ek^{\mathsf{PKE}}, vk^{\mathsf{Sig}}, c_1, c_2, c_\sigma), w := (m, ek_1, ek_2, r_1, s_1, r_2, s_2, \sigma, r)\big)$, and finally outputs the ciphertext $c := (c_1, c_2, c_\sigma, \pi)$.

Sanitization: On input sanitizer parameters sp^{sPKE} and a ciphertext $c = (c_1, c_2, c_\sigma, \pi)$, the algorithm $\mathsf{sPKE.San}$ first verifies the NIZK proof by evaluating $\mathsf{NIZK.Ver}\big(crs, x := (g, ek^{\mathsf{PKE}}, vk^{\mathsf{Sig}}, c_1, c_2, c_\sigma), \pi\big)$. It then parses $(c_{1,1}, c_{1,2}, c_{1,3}, c_{1,4}) \leftarrow c_1$. If the verification succeeds and $c_{1,1} \neq 1 \neq c_{1,2}$, then it chooses a random $t \twoheadleftarrow \mathbb{Z}_p^*$ and outputs the sanitized ciphertext

$$c' := \big((c_{1,1})^t \cdot c_{1,3},\ (c_{1,2})^t \cdot c_{1,4}\big).$$

If the verification fails or if $c_{1,1} = 1$ or $c_{1,2} = 1$, it outputs $\bot$.

Decryption: On input a decryption key $dk^{\mathsf{sPKE}} = (dk_1, dk_2)$ and a sanitized ciphertext $c' = (c'_1, c'_2)$, the algorithm $\mathsf{sPKE.Dec}$ computes the message $m \leftarrow c'_2 \cdot \big((c'_1)^{dk_1}\big)^{-1}$. It outputs m if $m \in \mathcal{M}$, and otherwise it outputs $\bot$. On input dk^{sPKE} and $\bot$, it outputs $\bot$.

We first prove correctness and other straightforward properties of the scheme.

Proposition 6.5.8. *If* Sig *is correct and* NIZK *has perfect completeness, the scheme* sPKE *from above is correct, robust, has unpredictable ciphertexts, and negligible encryption-key collision probability.*

Proof. For correctness, let $(sp^{\mathsf{sPKE}}, msk^{\mathsf{sPKE}})$ in the range of $\mathsf{sPKE.Setup}$, $(ek^{\mathsf{sPKE}}, dk^{\mathsf{sPKE}})$ in the range of $\mathsf{sPKE.Gen}(msk^{\mathsf{sPKE}})$, and let $m \in \mathcal{M}$. By correctness of Sig and completeness of NIZK, the NIZK verification in sPKE.San in the correctness experiment succeeds with probability 1. Moreover, since g generates G and $r_1, dk_1 \in \mathbb{Z}_p^*$, we have $c_{1,1} = g^{r_1} \neq 1$ and $c_{1,2} = ek_1^{r_1} = g^{dk_1 \cdot r_1} \neq 1$. Hence,

$$\begin{aligned} c' &= \mathsf{sPKE.San}\big(sp^{\mathsf{sPKE}}, \mathsf{sPKE.Enc}(ek^{\mathsf{sPKE}}, m)\big) \\ &= \big((c_{1,1})^t \cdot c_{1,3},\, (c_{1,2})^t \cdot c_{1,4}\big) \\ &= \big(g^{r_1 \cdot t + s_1},\, ek_1^{r_1 \cdot t + s_1} \cdot m\big). \end{aligned}$$

and

$$\begin{aligned} \mathsf{sPKE.Dec}\big(dk^{\mathsf{sPKE}}, c'\big) &= ek_1^{r_1 \cdot t + s_1} \cdot m \cdot \Big(\big(g^{r_1 \cdot t + s_1}\big)^{dk_1}\Big)^{-1} \\ &= g^{dk_1(r_1 \cdot t + s_1)} \cdot m \cdot \Big(g^{dk_1(r_1 \cdot t + s_1)}\Big)^{-1} \\ &= m. \end{aligned}$$

This shows that sPKE is correct.

For ciphertext unpredictability, note that each ciphertext contains g^{r_1}, g^{s_1}, g^{r_2}, and g^{s_2} for uniformly chosen $r_1, s_1, r_2, s_2 \in \mathbb{Z}_p^*$. Each of these elements can only be guessed with probability $1/|\mathbb{Z}_p^*| = 1/(p-1)$, where $p \geq 2^\kappa$. We can therefore conclude that for any $\mathcal{A}$,

$$\mathsf{Adv}^{\mathsf{sPKE\text{-}UPD\text{-}CTXT}}_{\mathsf{sPKE},\mathcal{A}} \leq \frac{1}{(p-1)^4} \leq \frac{1}{(2^\kappa - 1)^4}.$$

Similarly, since the encryption keys contain the pairs $(ek_1 = g^{dk_1}, ek_2 = g^{dk_2})$ for uniformly chosen $dk_1, dk_2 \in \mathbb{Z}_p^*$, we have

$$\mathsf{Col}^{\mathsf{ek}}_{\mathsf{sPKE}} \leq \frac{1}{(p-1)^2} \leq \frac{1}{(2^\kappa - 1)^2}.$$

We finally prove robustness. To this end, let $\mathcal{A}$ be a probabilistic algorithm that makes at most q queries to $\mathcal{O}_G$ and consider $\mathsf{Exp}^{\mathsf{sPKE\text{-}USROB}}_{\mathcal{E},\mathcal{A}}$. Further let ek_i^{sPKE} and $dk_i^{\mathsf{sPKE}} = (dk_{i,1}, dk_{i,2})$ be the keys returned from $\mathcal{O}_G$ for the i-th query and let (c, i_0, i_1) be the output of $\mathcal{A}$, where $c := (c_1, c_2, c_\sigma, \pi)$ and $c_1 = (g^a, g^b, g^c, g^d)$. Assume that $i_0 \neq i_1$ and that c passes sanitization, since $\mathcal{A}$ cannot win otherwise. This implies $a \neq 0 \neq b$ and sanitizing and decrypting the ciphertext with the two decryption keys yield $m_0 = g^{bt_0 + d - dk_{i_0,1}(at_0 + c)}$ and $m_1 = g^{bt_1 + d - dk_{i_1,1}(at_1 + c)}$, respectively, where $t_0, t_1 \in \mathbb{Z}_p^*$ are chosen uniformly during sanitization. We then have that $\mathcal{A}$ wins if $m_0, m_1 \in \mathcal{M}$. Assume that $m_0 \in \mathcal{M}$. Then,

$$\begin{aligned} m_1 &= m_0 \cdot g^{-b \cdot t_0 - d + dk_{i_0,1}(a \cdot t_0 + c)} \cdot g^{b \cdot t_1 + d - dk_{i_1,1}(a \cdot t_1 + c)} \\ &= m_0 \cdot g^{(dk_{i_0,1} - dk_{i_1,1})c} \cdot g^{(a \cdot dk_{i_0,1} - b)t_0} \cdot g^{(b - a \cdot dk_{i_1,1})t_1}. \end{aligned}$$

Note that if $dk_{i_0,1} \neq dk_{i_1,1}$ and $a \neq 0 \neq b$, then $a \cdot dk_{i_0,1} - b$ and $b - a \cdot dk_{i_1,1}$ cannot both be 0. Hence, in this case, m_1 is a uniformly random element in the group G. The probability that $m_1 \in \mathcal{M}$ is therefore $|\mathcal{M}|/|G| \leq 2^{-\kappa}$. Since $\mathcal{A}$ obtains at most q decryption keys and the $dk_{i,1}$ are uniform elements in $\mathbb{Z}_p^*$, the probability that $dk_{i_0,1} = dk_{i_1,1}$ is bounded by $q^2 \cdot 1/(p-1) \leq q^2 \cdot 1/(2^\kappa - 1)$. We can therefore conclude that

$$\mathsf{Adv}^{\mathsf{sPKE\text{-}USROB}}_{\mathsf{sPKE},\mathcal{A}} \leq 2^{-\kappa} + \frac{q^2}{2^\kappa - 1} \leq \frac{q^2 + 1}{2^\kappa - 1}. \qquad \square$$

The main result of this section is the security of the scheme, summarized in the following theorem.

Theorem 6.5.9. *If the DDH assumption holds in the group G,* PKE *is IND-CPA secure,* Sig *is EUF-CMA secure, and if* NIZK *is zero-knowledge, computationally sound, and one-time simulation sound, then the scheme* sPKE *from above is IND-CCA secure, IK-CCA secure, and SAN-CCA secure.*

On a high level, our proof proceeds as follows. It is rather straightforward to show that our variant of ElGamal encryption satisfies the CPA versions of the three properties. The proof of CCA security follows the proof by Sahai for public-key encryption [Sah99]: Since the NIZK ensures that both ciphertext components are encryptions of the same message, it does not matter which component is decrypted. In a reduction, where we assume an adversary $\mathcal{A}$ against the CCA variants of the desired properties, and we want to break the corresponding CPA variants, we only get one public key and no decryption oracle from the challenger. In order to emulate the view toward $\mathcal{A}$, 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 emulate a decryption oracle. To generate a challenge ciphertext, the reduction obtains one challenge ciphertexts from its CPA challenger, and encrypts another, arbitrary message to get 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 *one-time simulation sound* (see Definition 2.3.11). This ensures that even if the adversary sees one simulated (accepting) proof of a wrong statement, it is not capable of producing accepting proofs of wrong statements, except by reproducing the exact proof obtained within the challenge, but which $\mathcal{A}$ is not allowed to ask to the decryption oracle by the CCA definition. The fundamental result of Sahai [Sah99] is that the above strategy successfully simulates a complete CCA attack toward $\mathcal{A}$.

An additional obstacle we have is that to preserve anonymity, the NIZK needs to be verified without knowing which encryption keys were used. On the other hand, the reduction only works if the two used keys "match", since otherwise, the emulated decryption oracle would use an incorrect key to decrypt. To prevent an adversary from mixing different key pairs for encryptions, the key-generation process signs valid key pairs, and the NIZK ensures that a signed pair was used. Due to anonymity, this signature cannot be directly contained in the ciphertexts. Instead, it is part of the witness. To prove that if a ciphertext is accepted, the used key pair was indeed signed by the key-generation process, we show that if $\mathcal{A}$ manages to produce a ciphertext that is accepted but the keys were not signed, we can break EUF-CMA security of the signature scheme. In this reduction, we have to provide a forgery. Hence, the reduction needs to extract the signature and the used encryption keys from the ciphertext. This could be achieved by assuming that the NIZK is extractable. Extractability and simulation-soundness at the same time is, however, a quite strong assumption. Instead, we add an encryption of the signature and the key pair under a separate PKE scheme to the ciphertexts. The reduction can then generate the keys for this PKE scheme itself and perform extraction by decrypting that ciphertext.

Since the proofs for IND-CCA and IK-CCA security closely follow the proof by Sahai [Sah99], we first prove SAN-CCA security.

Lemma 6.5.10. *Let* sPKE *be the scheme from above and let* $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ *be a pair of probabilistic algorithms such that* $\mathcal{A}_1$ *and* $\mathcal{A}_2$ *together make at most* q_G *queries to* $\mathcal{O}_G$ *and at most* q_{SD} *queries to* $\mathcal{O}_{SD_0}$ *and* $\mathcal{O}_{SD_1}$ *combined. Then, there exist adversaries* $\mathcal{A}_{\mathsf{DDH}}$*,* $\mathcal{A}_{\mathsf{snd}}$*, and* $\mathcal{A}_{\mathsf{Sig}}$ *(which are all roughly as efficient as emulating an execution of* $\mathsf{Exp}^{\mathsf{sPKE\text{-}SAN\text{-}CCA}}_{\mathsf{sPKE},\mathcal{A}}$*) such that*

$$\begin{aligned}\mathsf{Adv}^{\mathsf{sPKE\text{-}SAN\text{-}CCA}}_{\mathsf{sPKE},\mathcal{A}} \le 8 \cdot \mathsf{Adv}^{\mathsf{DDH}}_{g,\mathcal{A}_{\mathsf{DDH}}} + (24 q_{SD} + 48) \cdot \mathsf{Adv}^{\mathsf{NIZK\text{-}snd}}_{\mathsf{NIZK},\mathcal{A}_{\mathsf{snd}}} \\ + 24 \cdot \mathsf{Adv}^{\mathsf{Sig\text{-}EUF\text{-}CMA}}_{\mathsf{Sig},\mathcal{A}_{\mathsf{Sig}}} + \frac{52 q_G^2 + 192 q_G + 196}{2^{\kappa} - 1}.\end{aligned}$$

Proof. Let W_{san} be the event that $\mathcal{A}$ wins the sanitization game, i.e.,

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

We define hybrid experiments H_0 to H_2 as follows:

- $H_0 := \mathsf{Exp}^{\mathsf{sPKE\text{-}SAN\text{-}CCA}}_{\mathsf{sPKE},\mathcal{A}}$ is the sanitization experiment.
- H_1 is identical to H_0, except that if $c_0' \neq \perp$, then c_0' is replaced by two uniformly random group elements (g^b, g^c).
- H_2 is identical to H_1, except that if $c_1' \neq \perp$, then c_1' is replaced by two uniformly random group elements (g^b, g^c).

In H_2, if $c_0' \neq \perp$ and $c_1' \neq \perp$, the view of $\mathcal{A}$ is independent of the bit b. Hence, $\mathcal{A}$ cannot guess b with probability more than 1/2 in this case. On the other hand, if $c_0' = \perp$ or $c_1' = \perp$, then $m_{0,0} = m_{0,1} = \perp$ or $m_{1,0} = m_{1,1} = \perp$, respectively, since $\perp$ decrypts to $\perp$. By definition of the sanitization advantage, $\mathcal{A}$ cannot win in this case. Thus,

$$\Pr^{H_2}[W_{\mathrm{san}}] \le \frac{1}{2}. \tag{6.6}$$

To conclude the proof, we show that the probability of W_{san} in H_0 differs only negligibly from its probability in H_2. To this end, we first prove that three bad events occur only with negligible probability in any of the hybrids.

Claim 1. *Let $i \in \{0, 1, 2\}$ and consider the experiment H_i. Further let B_1 be the event that $\mathcal{A}$ outputs as c_0 or c_1 or queries at least one of its decryption oracles with a valid but improper ciphertext $(c_1, c_2, c_\sigma, \pi)$, i.e., $\big(g, ek^{\mathsf{PKE}}, vk^{\mathsf{Sig}}, c_1, c_2, c_\sigma\big) \notin L$, but where π is an accepting proof, i.e., $\mathsf{NIZK.Ver}\big(crs, x := (g, ek^{\mathsf{PKE}}, vk^{\mathsf{Sig}}, c_1, c_2, c_\sigma), \pi\big) = 1$. Then, there exists an adversary $\mathcal{A}^i_{\mathsf{snd}}$ such that*

$$\Pr^{H_i}[B_1] \leq (q_{SD} + 2) \cdot \mathsf{Adv}^{\mathsf{NIZK\text{-}snd}}_{\mathsf{NIZK}, \mathcal{A}^i_{\mathsf{snd}}}.$$

Proof of claim. On input crs from the soundness challenger, $\mathcal{A}^i_{\mathsf{snd}}$ uses this CRS, generates all needed keys itself, and emulates an execution of H_i toward $\mathcal{A}$. It initially chooses $q_0 \twoheadleftarrow \{-1, 0, 1, \ldots, q_{SD}\}$ uniformly at random. If $q_0 > 0$ and when $\mathcal{A}$ submits the q_0-th query to a decryption oracle, $\mathcal{A}^i_{\mathsf{snd}}$ outputs the corresponding statement and proof to the challenger. If $q_0 \leq 0$ and when $\mathcal{A}$ outputs (c_0, c_1, st), then $\mathcal{A}^i_{\mathsf{snd}}$ submits the statement and proof from c_{q_0+1} to the challenger. If B_1 occurs, then for some q_0, $\mathcal{A}^i_{\mathsf{snd}}$ outputs an accepting proof for an incorrect statement. Hence, the claim follows. $\diamondsuit$

Claim 2. *Let $i \in \{0, 1, 2\}$ and consider the experiment H_i. Further let B_2 be the event that $\mathcal{A}$ outputs as c_0 or c_1 or queries at least one of its decryption oracles with a valid and proper ciphertext $(c_1, c_2, c_\sigma, \pi)$, i.e., $\big(g, ek^{\mathsf{PKE}}, vk^{\mathsf{Sig}}, c_1, c_2, c_\sigma\big) \in L$ 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 $\mathcal{O}_G$. Then, there exists an adversary $\mathcal{A}^i_{\mathsf{Sig}}$ such that*

$$\Pr^{H_i}[B_2] \leq \mathsf{Adv}^{\mathsf{Sig\text{-}EUF\text{-}CMA}}_{\mathsf{Sig}, \mathcal{A}^i_{\mathsf{Sig}}}.$$

Proof of claim. On input a signature verification key vk^{Sig}, $\mathcal{A}^i_{\mathsf{Sig}}$ generates all keys except for vk^{Sig} and sk^{Sig}, and emulates an execution of H_i. To generate the encryption keys ek_0^{sPKE} and ek_1^{sPKE} and to answer queries to $\mathcal{O}_G$, $\mathcal{A}^i_{\mathsf{Sig}}$ obtains the needed signature using the signing oracle of

$\mathsf{Exp}^{\mathsf{Sig\text{-}EUF\text{-}CMA}}_{\mathsf{Sig},\mathcal{A}^i_{\mathsf{Sig}}}$. The rest of H_i is straightforward to emulate since $\mathcal{A}^i_{\mathsf{Sig}}$ possesses all keys except for sk^{Sig}. Whenever $\mathcal{A}$ returns or submits a ciphertext $(c_1, c_2, c_\sigma, \pi)$ to one of the decryption oracles, $\mathcal{A}^i_{\mathsf{Sig}}$ decrypts c_σ to obtain a pair (ek_1', ek_2') and a signature σ'. If it has never queried (ek_1', ek_2') to its signing oracle and if the signature is valid, then it outputs $((ek_1', ek_2'), \sigma')$ as its forgery. Note that if B_2 occurs, $\mathcal{A}^i_{\mathsf{Sig}}$ obtains a forgery, so the claim follows. $\diamondsuit$

Claim 3. *Let $i \in \{0,1,2\}$, consider the experiment H_i, and let B_3 be the event that H_i generates two different encryption keys $ek^{\mathsf{sPKE}} = \big(g, p, crs, ek^{\mathsf{PKE}}, vk^{\mathsf{Sig}}, ek_1, ek_2, \sigma\big)$ and $\big(ek^{\mathsf{sPKE}}\big)' = \big(g, p, crs, ek^{\mathsf{PKE}}, vk^{\mathsf{Sig}}, ek_1', ek_2', \sigma'\big)$ such that $ek_1 = ek_1'$ or $ek_2 = ek_2'$. Then,*

$$\Pr^{H_i}[B_3] \le \frac{2(q_G+2)^2}{2^\kappa - 1}.$$

Proof of claim. The experiment H_i initially generates two encryption keys and then one for each query to $\mathcal{O}_G$. Hence, there are at most $(q_G+2)^2$ such pairs. For each of these pairs, the probability that one of the two components collides is at most $2 \cdot (1/|\mathbb{Z}_p^*|) = 2/(p-1)$. Using $p \ge 2^\kappa$ and the union bound implies the claim. $\diamondsuit$

We now bound the difference of the probabilities of W_{san} in different hybrids. To this end, let $B := B_1 \cup B_2 \cup B_3$.

Claim 4. *For all $i \in \{0,1\}$, there exists an adversary $\mathcal{A}^i_{\mathsf{DDH}}$ such that*

$$\Pr^{H_i}[W_{\mathsf{san}}] - \Pr^{H_{i+1}}[W_{\mathsf{san}}] \le 2 \cdot \mathsf{Adv}^{\mathsf{DDH}}_{g,\mathcal{A}^i_{\mathsf{DDH}}} + 2 \cdot \Pr^{H_i}[B] \\ + 4 \cdot \Pr^{H_{i+1}}[B] + \frac{q_G^2+1}{2^\kappa - 1}.$$

Proof of claim. Let $i \in \{0,1\}$ and let G_0 and G_1 be the events that c_i output by $\mathcal{A}$ is an encryption under ek_0^{sPKE} and ek_1^{sPKE}, respectively. If B, G_0, and G_1 all do not occur, then c_i is either invalid or a valid encryption under a key different from ek_0^{sPKE} and ek_1^{sPKE}. Since W_{san} can only occur if the ciphertext decrypts to a message different from $\perp$ under one of these keys, this only happens if robustness is violated. Using the result on robustness derived in the proof of Proposition 6.5.8, this implies

$$\Pr^{H_i}[W_{\mathsf{san}} \cap \neg B \cap \neg G_1 \cap \neg G_2] \le \Pr^{H_i}[W_{\mathsf{san}} \mid \neg B \cap \neg G_1 \cap \neg G_2] \le \frac{q_G^2 + 1}{2^\kappa - 1}.$$

We also have

$$\begin{aligned}
&\Pr^{H_i}[W_{\mathsf{san}}] - \Pr^{H_{i+1}}[W_{\mathsf{san}}] \\
&= \Pr^{H_i}[W_{\mathsf{san}} \cap \neg B \cap (G_1 \cup G_2)] + \Pr^{H_i}[W_{\mathsf{san}} \cap (B \cup \neg(G_1 \cup G_2))] \\
&\qquad - \Pr^{H_{i+1}}[W_{\mathsf{san}} \cap \neg B \cap (G_1 \cup G_2)] \\
&\qquad - \Pr^{H_{i+1}}[W_{\mathsf{san}} \cap (B \cup \neg(G_1 \cup G_2))] \\
&\le \Pr^{H_i}[W_{\mathsf{san}} \cap \neg B \cap (G_1 \cup G_2)] - \Pr^{H_{i+1}}[W_{\mathsf{san}} \cap \neg B \cap (G_1 \cup G_2)] \\
&\qquad + \Pr^{H_i}[B] + \Pr^{H_i}[W_{\mathsf{san}} \cap \neg(G_1 \cup G_2)],
\end{aligned}$$

and

$$\Pr^{H_i}[W_{\mathsf{san}} \cap \neg(G_1 \cup G_2)] \le \Pr^{H_i}[W_{\mathsf{san}} \cap \neg B \cap \neg(G_1 \cup G_2)] + \Pr^{H_i}[B].$$

This implies

$$\begin{aligned}
\Pr^{H_i}[W_{\mathsf{san}}] - \Pr^{H_{i+1}}[W_{\mathsf{san}}] &\le \Pr^{H_i}[W_{\mathsf{san}} \cap \neg B \cap (G_1 \cup G_2)] \\
&\quad - \Pr^{H_{i+1}}[W_{\mathsf{san}} \cap \neg B \cap (G_1 \cup G_2)] + 2 \cdot \Pr^{H_i}[B] + \frac{q_G^2 + 1}{2^\kappa - 1}.
\end{aligned} \tag{6.7}$$

We now define the adversary $\mathcal{A}^i_{\mathsf{DDH}}$. On input (X, Y, T), $\mathcal{A}^i_{\mathsf{DDH}}$ chooses $j \twoheadleftarrow \{0,1\}$ uniformly at random and sets $ek_{j,1} \leftarrow X$. All remaining keys, including $ek_{j,2}$, are generated as in H_i, and $\mathcal{A}$ is invoked on $\big(sp^{\mathsf{sPKE}}, ek_0^{\mathsf{sPKE}} = (ek_{0,1}, ek_{0,2}), ek_1^{\mathsf{sPKE}} = (ek_{1,1}, ek_{1,2})\big)$. The adversary $\mathcal{A}^i_{\mathsf{DDH}}$ then emulates an execution of H_i. Since it has all keys except for the decryption key $dk_{j,1}$, only the emulation of the decryption oracle $\mathcal{O}_{SD_j}$ is nontrivial. To answer queries to this oracle, $\mathcal{A}^i_{\mathsf{DDH}}$ sanitizes and decrypts the second ciphertext component instead of the first one using $dk_{j,2}$. When $\mathcal{A}$ outputs $\big(c_0, c_1, st\big)$, both ciphertexts are sanitized and decrypted as in the emulation of the decryption oracles, except that $m_{i,j}$ is not set to $\perp$ during decryption if $m_{i,j} \notin \mathcal{M}$. If $c'_i \neq \perp$, it is replaced by $c'_i \leftarrow (Y, T \cdot m_{i,j})$. Moreover, $\mathcal{A}^i_{\mathsf{DDH}}$ decrypts $c_{i,\sigma}$ and checks whether it contains the encryption keys corresponding to ek_j^{sPKE}. If this is not the case, it terminates and returns 0. Otherwise, it continues with the emulation. Finally, when $\mathcal{A}$ terminates, $\mathcal{A}^i_{\mathsf{DDH}}$ outputs $d = 1$ if W_{san} occurs, and $d = 0$ otherwise.

Note that B not occurring implies that $\mathcal{A}^i_{\mathsf{DDH}}$ emulates the decryption oracle perfectly since in this case, all submitted valid ciphertexts contain two encryptions of the same message under a signed key pair. Moreover, due to $\neg B_3$, the first encryption key matches the first key of the oracle if and only if the second keys match. If they match, decryption with either key yields the correct message with probability 1. Otherwise, the message (before potentially being set to $\perp$) is a uniform group element for both keys, as shown in the robustness proof of Proposition 6.5.8.

Furthermore, if (X, Y, T) are three independent uniform group elements, c'_i gets replaced by two uniformly random group elements if $c'_i \neq \perp$, as in H_{i+1}. On the other hand, if $\neg B$ and G_j occur and if $c'_i \neq \perp$, then $c_i = (c_{i,1}, c_{i,2}, c_{i,\sigma}, \pi_i)$ is a valid encryption of $m_{i,j}$ under ek_j^{sPKE}. Hence, there exist $r_1, s_1 \in \mathbb{Z}_p^*$ such that

$$c_{i,1} = \big(g^{r_1}, (ek_{j,1})^{r_1}, g^{s_1}, (ek_{j,1})^{s_1} \cdot m_{i,j}\big) = \big(g^{r_1}, X^{r_1}, g^{s_1}, X^{s_1} \cdot m_{i,j}\big).$$

In H_i, this ciphertext is sanitized to $c'_i = \big(g^{r_1 \cdot t + s_1}, X^{r_1 \cdot t + s_1} \cdot m_{i,j}\big)$ for $t \twoheadleftarrow \mathbb{Z}_p^*$. If we further have $X = g^a$, $Y = g^b$, and $T = g^{ab}$, then this corresponds to $c'_i = \big(g^{r_1 \cdot t + s_1}, g^{a \cdot (r_1 \cdot t + s_1)} \cdot m_{i,j}\big)$, which is equally distributed as the sanitization $(Y, T \cdot m_{i,j})$ generated by $\mathcal{A}^i_{\mathsf{DDH}}$. Since we also have that the probability of $\neg B \cap G_j$ is equal in $\mathsf{DDH}^{\text{real}}_{g,\mathcal{A}^i_{\mathsf{DDH}}}$ and H_i, as well as in $\mathsf{DDH}^{\text{rand}}_{g,\mathcal{A}^i_{\mathsf{DDH}}}$ and H_{i+1}, we can conclude

$$\begin{aligned}
\Pr^{\mathsf{DDH}^{\text{real}}_{g,\mathcal{A}^i_{\mathsf{DDH}}}}[d = 1 \cap \neg B \cap G_j] &= \Pr^{H_i}[W_{\text{san}} \cap \neg B \cap G_j], \\
\Pr^{\mathsf{DDH}^{\text{rand}}_{g,\mathcal{A}^i_{\mathsf{DDH}}}}[d = 1 \cap \neg B \cap G_j] &= \Pr^{H_{i+1}}[W_{\text{san}} \cap \neg B \cap G_j].
\end{aligned}$$

Using this together with

$$\begin{aligned}
&\mathsf{Adv}^{\mathsf{DDH}}_{g,\mathcal{A}^i_{\mathsf{DDH}}} \\
&= \Pr^{\mathsf{DDH}^{\text{real}}_{g,\mathcal{A}^i_{\mathsf{DDH}}}}[d = 1] - \Pr^{\mathsf{DDH}^{\text{rand}}_{g,\mathcal{A}^i_{\mathsf{DDH}}}}[d = 1] \\
&= \Pr^{\mathsf{DDH}^{\text{real}}_{g,\mathcal{A}^i_{\mathsf{DDH}}}}[d = 1 \cap \neg B \cap G_j] + \Pr^{\mathsf{DDH}^{\text{real}}_{g,\mathcal{A}^i_{\mathsf{DDH}}}}[d = 1 \cap (B \cup \neg G_j)] \\
&\qquad - \Pr^{\mathsf{DDH}^{\text{rand}}_{g,\mathcal{A}^i_{\mathsf{DDH}}}}[d = 1 \cap \neg B \cap G_j] - \Pr^{\mathsf{DDH}^{\text{rand}}_{g,\mathcal{A}^i_{\mathsf{DDH}}}}[d = 1 \cap (B \cup \neg G_j)],
\end{aligned}$$

we obtain

$$\begin{aligned}\mathsf{Adv}^{\mathsf{DDH}}_{g,\mathcal{A}^i_{\mathsf{DDH}}} \geq{}& \Pr^{H_i}[W_{\text{san}} \cap \neg B \cap G_j] - \Pr^{H_{i+1}}[W_{\text{san}} \cap \neg B \cap G_j]\\ &- \Pr^{\mathsf{DDH}^{\text{rand}}_{g,\mathcal{A}^i_{\mathsf{DDH}}}}[d = 1 \cap B] - \Pr^{\mathsf{DDH}^{\text{rand}}_{g,\mathcal{A}^i_{\mathsf{DDH}}}}[d = 1 \cap \neg G_j].\end{aligned}$$

If $\neg B$ occurs, then $c_{i,\sigma}$ contains the correct encryption keys and thus, if also $\neg G_j$ occurs, $\mathcal{A}^i_{\mathsf{DDH}}$ always returns 0. This implies $\Pr^{\mathsf{DDH}^{\text{rand}}_{g,\mathcal{A}^i_{\mathsf{DDH}}}}[d = 1 \cap \neg G_j \cap \neg B] = 0$, and therefore

$$\begin{aligned}\Pr^{\mathsf{DDH}^{\text{rand}}_{g,\mathcal{A}^i_{\mathsf{DDH}}}}[d = 1 \cap \neg G_j] &= \Pr^{\mathsf{DDH}^{\text{rand}}_{g,\mathcal{A}^i_{\mathsf{DDH}}}}[d = 1 \cap \neg G_j \cap \neg B]\\ &\quad + \Pr^{\mathsf{DDH}^{\text{rand}}_{g,\mathcal{A}^i_{\mathsf{DDH}}}}[d = 1 \cap \neg G_j \cap B]\\ &\leq \Pr^{\mathsf{DDH}^{\text{rand}}_{g,\mathcal{A}^i_{\mathsf{DDH}}}}[B].\end{aligned}$$

Using $\Pr^{\mathsf{DDH}^{\text{rand}}_{g,\mathcal{A}^i_{\mathsf{DDH}}}}[B] = \Pr^{H_{i+1}}[B]$, we obtain
$\mathsf{Adv}^{\mathsf{DDH}}_{g,\mathcal{A}^i_{\mathsf{DDH}}} \geq \Pr^{H_i}[W_{\text{san}} \cap \neg B \cap G_j] - \Pr^{H_{i+1}}[W_{\text{san}} \cap \neg B \cap G_j] - 2 \cdot \Pr^{H_{i+1}}[B]$.

Combining this with equation (6.7) and the fact that given $G_1 \cup G_2$ and $\neg B$, G_j occurs with probability 1/2 (independently of W_{san}), we can conclude

$$\begin{aligned}&\Pr^{H_i}[W_{\mathsf{san}}] - \Pr^{H_{i+1}}[W_{\mathsf{san}}]\\ &\leq 2 \cdot \Pr^{H_i}[W_{\mathsf{san}} \cap \neg B \cap G_j] - 2 \cdot \Pr^{H_{i+1}}[W_{\mathsf{san}} \cap \neg B \cap G_j]\\ &\qquad + 2 \cdot \Pr^{H_i}[B] + \frac{q_G^2 + 1}{2^\kappa - 1}\\ &\leq 2 \cdot \mathsf{Adv}^{\mathsf{DDH}}_{g,\mathcal{A}^i_{\mathsf{DDH}}} + 4 \cdot \Pr^{H_{i+1}}[B] + 2 \cdot \Pr^{H_i}[B] + \frac{q_G^2 + 1}{2^\kappa - 1}. \qquad \Diamond\end{aligned}$$

Using Claim 4 and equation (6.6), we obtain

$$\begin{aligned}\mathsf{Adv}^{\mathsf{sPKE\text{-}SAN\text{-}CCA}}_{\mathsf{sPKE},\mathcal{A}} &= 2 \cdot \Pr^{H_0}[W_{\text{san}}] - 1\\ &= 2 \cdot \big(\Pr^{H_0}[W_{\text{san}}] - \Pr^{H_1}[W_{\text{san}}] + \Pr^{H_1}[W_{\text{san}}]\\ &\qquad - \Pr^{H_2}[W_{\text{san}}] + \Pr^{H_2}[W_{\text{san}}]\big) - 1\\ &\leq 4 \cdot \mathsf{Adv}^{\mathsf{DDH}}_{g,\mathcal{A}^0_{\mathsf{DDH}}} + 4 \cdot \mathsf{Adv}^{\mathsf{DDH}}_{g,\mathcal{A}^1_{\mathsf{DDH}}} + 4 \cdot \Pr^{H_0}[B]\\ &\qquad + 12 \cdot \Pr^{H_1}[B] + 8 \cdot \Pr^{H_2}[B] + 4 \cdot \frac{q_G^2 + 1}{2^\kappa - 1}.\end{aligned}$$

Claims 1 to 3 further imply

$$\Pr^{H_i}[B] \le (q_{SD}+2)\cdot \mathsf{Adv}^{\text{NIZK-snd}}_{\text{NIZK},\mathcal{A}^i_{\text{snd}}} + \mathsf{Adv}^{\text{Sig-EUF-CMA}}_{\text{Sig},\mathcal{A}^i_{\text{Sig}}} + \frac{2(q_G+2)^2}{2^\kappa-1}.$$

Hence,

$$\begin{aligned}
&\mathsf{Adv}^{\text{sPKE-SAN-CCA}}_{\text{sPKE},\mathcal{A}}\\
&\le 4\cdot \mathsf{Adv}^{\text{DDH}}_{g,\mathcal{A}^0_{\text{DDH}}} + 4\cdot \mathsf{Adv}^{\text{DDH}}_{g,\mathcal{A}^1_{\text{DDH}}} + (4q_{SD}+8)\cdot \mathsf{Adv}^{\text{NIZK-snd}}_{\text{NIZK},\mathcal{A}^0_{\text{snd}}}\\
&\quad + (12q_{SD}+24)\cdot \mathsf{Adv}^{\text{NIZK-snd}}_{\text{NIZK},\mathcal{A}^1_{\text{snd}}} + (8q_{SD}+16)\cdot \mathsf{Adv}^{\text{NIZK-snd}}_{\text{NIZK},\mathcal{A}^2_{\text{snd}}}\\
&\quad + 4\cdot \mathsf{Adv}^{\text{Sig-EUF-CMA}}_{\text{Sig},\mathcal{A}^0_{\text{Sig}}} + 12\cdot \mathsf{Adv}^{\text{Sig-EUF-CMA}}_{\text{Sig},\mathcal{A}^1_{\text{Sig}}} + 8\cdot \mathsf{Adv}^{\text{Sig-EUF-CMA}}_{\text{Sig},\mathcal{A}^2_{\text{Sig}}}\\
&\quad + \frac{48(q_G+2)^2+4q_G^2+4}{2^\kappa-1}.
\end{aligned}$$

We define the adversary $\mathcal{A}_{\text{DDH}}$ as running $\mathcal{A}^0_{\text{DDH}}$ and $\mathcal{A}^1_{\text{DDH}}$ with probability $\frac{1}{2}$ each, the adversary $\mathcal{A}_{\text{snd}}$ as running $\mathcal{A}^0_{\text{snd}}$ with probability $\frac{4q_{SD}+8}{24q_{SD}+48}$, $\mathcal{A}^1_{\text{snd}}$ with probability $\frac{12q_{SD}+24}{24q_{SD}+48}$, and $\mathcal{A}^2_{\text{snd}}$ with probability $\frac{8q_{SD}+16}{24q_{SD}+48}$, and the adversary $\mathcal{A}_{\text{Sig}}$ as running $\mathcal{A}^0_{\text{Sig}}$ with probability $\frac{4}{24}$, $\mathcal{A}^1_{\text{Sig}}$ with probability $\frac{12}{24}$, and $\mathcal{A}^2_{\text{Sig}}$ with probability $\frac{8}{24}$. Using the result above, we finally conclude

$$\begin{aligned}
\mathsf{Adv}^{\text{sPKE-SAN-CCA}}_{\text{sPKE},\mathcal{A}} &\le 8\cdot \mathsf{Adv}^{\text{DDH}}_{g,\mathcal{A}_{\text{DDH}}} + (24q_{SD}+48)\cdot \mathsf{Adv}^{\text{NIZK-snd}}_{\text{NIZK},\mathcal{A}_{\text{snd}}}\\
&\quad + 24\cdot \mathsf{Adv}^{\text{Sig-EUF-CMA}}_{\text{Sig},\mathcal{A}_{\text{Sig}}} + \frac{52q_G^2+192q_G+196}{2^\kappa-1}. \qquad \square
\end{aligned}$$

We next show that our sPKE scheme is IND-CCA secure. The proof follows Lindell's proof for the construction of an IND-CCA secure public-key encryption scheme from a IND-CPA secure one [Lin06].

Lemma 6.5.11. *Let* sPKE *be the scheme from Section 6.5.2 and let* $\mathcal{A}=(\mathcal{A}_1,\mathcal{A}_2)$ *be a pair of probabilistic algorithms such that* $\mathcal{A}_1$ *and* $\mathcal{A}_2$ *together make at most* q_G *queries to* $\mathcal{O}_G$ *and at most* q_{SD} *queries to* $\mathcal{O}_{SD}$*. Then, there exist adversaries* $\mathcal{A}_{\text{DDH}}$*,* $\mathcal{A}_{\text{ZK}}$*,* $\mathcal{A}_{\text{snd}}$*, and* $\mathcal{A}_{\text{Sig}}$ *(which are all roughly as efficient as emulating an execution of* $\mathsf{Exp}^{\text{sPKE-IND-CCA}}_{\text{sPKE},\mathcal{A}}$*) such that*

$$\begin{aligned}
\mathsf{Adv}^{\text{sPKE-IND-CCA}}_{\text{sPKE},\mathcal{A}} &\le 4\cdot \mathsf{Adv}^{\text{DDH}}_{g,\mathcal{A}_{\text{DDH}}} + 2\cdot \mathsf{Adv}^{\text{NIZK-ZK}}_{\text{NIZK},\mathcal{A}_{\text{ZK}}} + 2q_{SD}\cdot \mathsf{Adv}^{\text{NIZK-sim-snd}}_{\text{NIZK},\mathcal{A}_{\text{snd}}}\\
&\quad + 2\cdot \mathsf{Adv}^{\text{Sig-EUF-CMA}}_{\text{Sig},\mathcal{A}_{\text{Sig}}} + \frac{4(q_G+1)^2+8}{2^\kappa-1}.
\end{aligned}$$

Proof. We assume without loss of generality that $\mathcal{A}_2$ does not query the challenge ciphertext c^* to its decryption oracle $\mathcal{O}_{SD}$ since doing so can only decrease the advantage. For $b_1, b_2 \in \{0,1\}$, we define the hybrid experiment H_{b_1,b_2} as follows: Let H_{b_1,b_2} be as $\mathsf{Exp}^{\mathsf{sPKE\text{-}IND\text{-}CCA}}_{\mathsf{sPKE},\mathcal{A}}$, but where the common reference string *crs* is obtained via $(crs, \tau) \leftarrow S_1(1^\kappa)$ (instead of an invocation of $\mathsf{NIZK.Gen}$). When $\mathcal{A}_1$ outputs (m_0, m_1, st), H_{b_1,b_2} computes c_1 as the encryption of m_{b_1} under ek_1, c_2 as the encryption of m_{b_2} under ek_2, and c_σ as in the real experiment (namely as the encryption of the two ElGamal public keys and the accompanying signature). It then simulates the proof π using S_2 and invokes $\mathcal{A}_2$ on input st and $c^* := (c_1, c_2, c_\sigma, \pi)$.

Claim 1. *There exist adversaries $\mathcal{A}'_{\mathsf{ZK}}$ and $\mathcal{A}''_{\mathsf{ZK}}$ such that*

$$\Pr^{H_{0,0}}[b' = 1] - \Pr^{\mathsf{Exp}^{\mathsf{sPKE\text{-}IND\text{-}CCA}}_{\mathsf{sPKE},\mathcal{A}}}[b' = 1 \mid b = 0] = \mathsf{Adv}^{\mathsf{NIZK\text{-}ZK}}_{\mathsf{NIZK},\mathcal{A}'_{\mathsf{ZK}}},$$
$$\Pr^{\mathsf{Exp}^{\mathsf{sPKE\text{-}IND\text{-}CCA}}_{\mathsf{sPKE},\mathcal{A}}}[b' = 1 \mid b = 1] - \Pr^{H_{1,1}}[b' = 1] = \mathsf{Adv}^{\mathsf{NIZK\text{-}ZK}}_{\mathsf{NIZK},\mathcal{A}''_{\mathsf{ZK}}}.$$

Proof of claim. We only prove the first part of the claim, the second one can be shown analogously. The adversary $\mathcal{A}'_{\mathsf{ZK}}$ on input *crs*, emulates toward $\mathcal{A}$ the experiment $H_{0,0}$. To this end, it generates all required keys. When generating the challenge ciphertext $c^* = (c_1, c_2, c_\sigma, \pi)$, it obtains π via the proof oracle. Note that in $H_{0,0}$, this ciphertext is a valid encryption of m_0, so the statement is correct and the proof oracle consequently returns a valid proof. When $\mathcal{A}$ returns a bit b', $\mathcal{A}'_{\mathsf{ZK}}$ returns $1 - b'$. Observe that if the CRS and the proofs are real, then this emulation is equivalent to the experiment $\mathsf{Exp}^{\mathsf{sPKE\text{-}IND\text{-}CCA}}_{\mathsf{sPKE},\mathcal{A}}$ when $b = 0$, and if the CRS and the proofs are simulated, then it is equivalent to $H_{0,0}$. We therefore have

$$\begin{aligned}
\mathsf{Adv}^{\mathsf{NIZK\text{-}ZK}}_{\mathsf{NIZK},\mathcal{A}'_{\mathsf{ZK}}} &= \Pr^{crs \leftarrow \mathsf{Gen}(1^\kappa)}\big[\mathcal{A}^{\mathsf{Prove}(crs,\cdot,\cdot)}(crs) = 1\big] \\
&\quad - \Pr^{(crs,\tau) \leftarrow S_1(1^\kappa)}\big[\mathcal{A}^{S'(crs,\tau,\cdot,\cdot)}(crs) = 1\big] \\
&= 1 - \Pr^{\mathsf{Exp}^{\mathsf{sPKE\text{-}IND\text{-}CCA}}_{\mathsf{sPKE},\mathcal{A}}}[b' = 1 \mid b = 0] - \big(1 - \Pr^{H_{0,0}}[b' = 1]\big) \\
&= \Pr^{H_{0,0}}[b' = 1] - \Pr^{\mathsf{Exp}^{\mathsf{sPKE\text{-}IND\text{-}CCA}}_{\mathsf{sPKE},\mathcal{A}}}[b' = 1 \mid b = 0],
\end{aligned}$$

which implies the claim. ◊

Analogous to the proof of Lemma 6.5.10, we define three bad events: Let B_1 be the event that $\mathcal{A}$ queries its decryption oracle with a valid but improper ciphertext $(c_1, c_2, c_\sigma, \pi)$, i.e., $\big(g, ek^{\mathsf{PKE}}, vk^{\mathsf{Sig}}, c_1, c_2, c_\sigma\big) \notin L$, but where π is an accepting proof, i.e., $\mathsf{NIZK.Ver}\big(crs, x := (g, ek^{\mathsf{PKE}}, vk^{\mathsf{Sig}}, c_1, c_2, c_\sigma), \pi\big) = 1$. As in the proof of Lemma 6.5.10, one can show that there exists an adversary $\mathcal{A}_{\mathsf{snd}}^{b_1,b_2}$ such that

$$\Pr^{H_{b_1,b_2}}[B_1] \le q_{SD} \cdot \mathsf{Adv}^{\mathsf{NIZK\text{-}sim\text{-}snd}}_{\mathsf{NIZK},\mathcal{A}_{\mathsf{snd}}^{b_1,b_2}},$$

except that we here need (one-time) simulation soundness since the proof in the challenge ciphertext is simulated.

Let B_2 be the event that $\mathcal{A}$ queries its decryption oracle with a valid and proper ciphertext $(c_1, c_2, c_\sigma, \pi)$, i.e., $\big(g, ek^{\mathsf{PKE}}, vk^{\mathsf{Sig}}, c_1, c_2, c_\sigma\big) \in L$ 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 $\mathcal{O}_G$. Again as in the proof of Lemma 6.5.10, it can be shown that there exists an adversary $\mathcal{A}_{\mathsf{Sig}}^{b_1,b_2}$ such that

$$\Pr^{H_{b_1,b_2}}[B_2] \le \mathsf{Adv}^{\mathsf{Sig\text{-}EUF\text{-}CMA}}_{\mathsf{Sig},\mathcal{A}_{\mathsf{Sig}}^{b_1,b_2}}.$$

Finally, let B_3 be the event that H_{b_1,b_2} generates two different encryption keys $ek^{\mathsf{sPKE}} = \big(g, p, crs, ek^{\mathsf{PKE}}, vk^{\mathsf{Sig}}, ek_1, ek_2, \sigma\big)$ and $\big(ek^{\mathsf{sPKE}}\big)' = \big(g, p, crs, ek^{\mathsf{PKE}}, vk^{\mathsf{Sig}}, ek_1', ek_2', \sigma'\big)$ such that $ek_1 = ek_1'$ or $ek_2 = ek_2'$. Then,

$$\Pr^{H_{b_1,b_2}}[B_3] \le \frac{2(q_G+1)^2}{2^\kappa - 1},$$

which can be shown as in the proof of Lemma 6.5.10. For $B := B_1 \cup B_2 \cup B_3$, we therefore have

$$\Pr^{H_{b_1,b_2}}[B] \le q_{SD} \cdot \mathsf{Adv}^{\mathsf{NIZK\text{-}sim\text{-}snd}}_{\mathsf{NIZK},\mathcal{A}_{\mathsf{snd}}^{b_1,b_2}} + \mathsf{Adv}^{\mathsf{Sig\text{-}EUF\text{-}CMA}}_{\mathsf{Sig},\mathcal{A}_{\mathsf{Sig}}^{b_1,b_2}} + \frac{2(q_G+1)^2}{2^\kappa - 1}. \quad (6.8)$$

Claim 2. *There exist adversaries $\mathcal{A}'_{\mathsf{DDH}}$ and $\mathcal{A}''_{\mathsf{DDH}}$ such that*

$$\begin{aligned}\Pr^{H_{1,0}}[b'=1] - \Pr^{H_{0,0}}[b'=1] &\le 2 \cdot \mathsf{Adv}^{\mathsf{DDH}}_{g,\mathcal{A}'_{\mathsf{DDH}}} + \Pr^{H_{0,0}}[B] \\ &\qquad + \Pr^{H_{1,0}}[B] + 2^{2-\kappa}, \\ \Pr^{H_{1,1}}[b'=1] - \Pr^{H_{1,0}}[b'=1] &\le 2 \cdot \mathsf{Adv}^{\mathsf{DDH}}_{g,\mathcal{A}''_{\mathsf{DDH}}} + 2^{2-\kappa}.\end{aligned}$$

Proof of claim. We define the adversary $\mathcal{A}'_{\mathsf{DDH}}$ as follows. On input a triple (X, Y, T), it sets $ek_1 \leftarrow X$. It further generates all the remaining keys of the experiment (and thus lacks only the decryption key dk_1), samples $b \twoheadleftarrow \{0, 1\}$, and emulates $H_{b,0}$ toward $\mathcal{A}$. The oracle $\mathcal{O}_{SD}$ is emulated by decrypting the second ciphertext component instead of the first one using dk_2. When $\mathcal{A}_1$ returns (m_0, m_1, st), $\mathcal{A}'_{\mathsf{DDH}}$ samples $r \twoheadleftarrow \mathbb{Z}_p^*$ and sets

$$c_1 \leftarrow (g^r, X^r, Y, T \cdot m_b).$$

It further computes c_2 as an ElGamal encryption of m_0, encrypts both keys and their signature to obtain c_σ, and simulates the NIZK proof π using S_2. It continues the emulation by giving $c^* := (c_1, c_2, c_\sigma, \pi)$ to $\mathcal{A}_2$. When $\mathcal{A}_2$ outputs its decision bit b', $\mathcal{A}'_{\mathsf{DDH}}$ outputs $d = 1$ if $b' = b$, and $d = 0$ otherwise.

First note that if (X, Y, T) are three uniform group elements, c_1 is independent of the bit b, and thus

$$\Pr^{\mathsf{DDH}^{\mathrm{rand}}_{g,\mathcal{A}'_{\mathsf{DDH}}}}[d = 1] = \frac{1}{2}.$$

On the other hand, if (X, Y, T) is a DDH triple, we have for a uniform $s \in \mathbb{Z}_p$,

$$c_1 = (g^r, X^r, Y, T \cdot m_b) = (g^r, ek_1^r, g^s, ek_1^s \cdot m_b),$$

which corresponds to a proper ElGamal encryption of m_b if $X \neq 1$ and $Y \neq 1$. Further note that, as in the proof of Lemma 6.5.10, $\mathcal{O}_{SD}$ is emulated perfectly if B does not occur. We therefore have

$$\Pr^{\mathsf{DDH}^{\mathrm{real}}_{g,\mathcal{A}'_{\mathsf{DDH}}}}[d = 1 \cap \neg B \mid X \neq 1 \neq Y] = \Pr^{H_{b,0}}[b' = b \cap \neg B].$$

This implies

$$\begin{aligned}
&\mathsf{Adv}^{\mathsf{DDH}}_{g,\mathcal{A}'_{\mathsf{DDH}}} \\
&= \Pr^{\mathsf{DDH}^{\mathrm{real}}_{g,\mathcal{A}'_{\mathsf{DDH}}}}[d = 1] - \Pr^{\mathsf{DDH}^{\mathrm{rand}}_{g,\mathcal{A}'_{\mathsf{DDH}}}}[d = 1] \\
&\geq \Pr^{\mathsf{DDH}^{\mathrm{real}}_{g,\mathcal{A}'_{\mathsf{DDH}}}}[d = 1 \cap \neg B \mid X \neq 1 \neq Y] \cdot \Pr^{\mathsf{DDH}^{\mathrm{real}}_{g,\mathcal{A}'_{\mathsf{DDH}}}}[X \neq 1 \neq Y] - \frac{1}{2} \\
&= \Pr^{H_{b,0}}[b' = b \cap \neg B] \cdot \Pr^{\mathsf{DDH}^{\mathrm{real}}_{g,\mathcal{A}'_{\mathsf{DDH}}}}[X \neq 1 \neq Y] - \frac{1}{2}.
\end{aligned}$$

Using the union bound and $|G| = p \geq 2^\kappa$, we further have

$$\begin{aligned}
&\Pr^{\mathsf{DDH}^{\mathrm{real}}_{g,\mathcal{A}'_{\mathsf{DDH}}}}[X \neq 1 \neq Y] \\
&= 1 - \Pr^{\mathsf{DDH}^{\mathrm{real}}_{g,\mathcal{A}'_{\mathsf{DDH}}}}[X = 1 \vee Y = 1] \\
&\geq 1 - \Pr^{\mathsf{DDH}^{\mathrm{real}}_{g,\mathcal{A}'_{\mathsf{DDH}}}}[X = 1] - \Pr^{\mathsf{DDH}^{\mathrm{real}}_{g,\mathcal{A}'_{\mathsf{DDH}}}}[Y = 1] \\
&\geq 1 - 2 \cdot 2^{-\kappa}.
\end{aligned}$$

Hence,

$$\mathsf{Adv}^{\mathsf{DDH}}_{g,\mathcal{A}'_{\mathsf{DDH}}} \geq \Pr^{H_{b,0}}[b' = b \cap \neg B] - 2^{1-\kappa} - \frac{1}{2}.$$

Since

$$\begin{aligned}
\Pr^{H_{b,0}}[b' = b] &\leq \Pr^{H_{b,0}}[(b' = b \cap \neg B) \cup B] \\
&\leq \Pr^{H_{b,0}}[b' = b \cap \neg B] + \Pr^{H_{b,0}}[B],
\end{aligned}$$

we obtain

$$\begin{aligned}
\mathsf{Adv}^{\mathsf{DDH}}_{g,\mathcal{A}'_{\mathsf{DDH}}} &\geq \Pr^{H_{b,0}}[b' = b] - \Pr^{H_{b,0}}[B] - 2^{1-\kappa} - \frac{1}{2} \\
&= \frac{1}{2}\Pr^{H_{0,0}}[b' = 0] + \frac{1}{2}\Pr^{H_{1,0}}[b' = 1] \\
&\quad - \frac{1}{2}\Pr^{H_{0,0}}[B] - \frac{1}{2}\Pr^{H_{1,0}}[B] - 2^{1-\kappa} - \frac{1}{2} \\
&= \frac{1}{2}\Pr^{H_{1,0}}[b' = 1] - \frac{1}{2}\Pr^{H_{0,0}}[b' = 1] \\
&\quad - \frac{1}{2}\Pr^{H_{0,0}}[B] - \frac{1}{2}\Pr^{H_{1,0}}[B] - 2^{1-\kappa}.
\end{aligned}$$

Rearranging the inequality concludes the proof of the first part of the claim.

The second part of the claim can be proven analogously, where $\mathcal{A}''_{\mathsf{DDH}}$ sets $ek_2 \leftarrow X$ instead of $ek_1 \leftarrow X$. Since it therefore has dk_1, which is the key used by the decryption algorithm, the decryption oracle can be emulated perfectly, even if B occurs. $\Diamond$

Using Claims 1 and 2, we get

$$
\begin{aligned}
&\mathsf{Adv}^{\mathsf{sPKE\text{-}IND\text{-}CCA}}_{\mathsf{sPKE},\mathcal{A}} \\
&\leq 2 \cdot \Pr^{\mathsf{Exp}^{\mathsf{sPKE\text{-}IND\text{-}CCA}}_{\mathsf{sPKE},\mathcal{A}}}[b' = b] - 1 \\
&= 2 \cdot \left(\frac{1}{2} \cdot \Pr^{\mathsf{Exp}^{\mathsf{sPKE\text{-}IND\text{-}CCA}}_{\mathsf{sPKE},\mathcal{A}}}[b' = 0 \mid b = 0] \right. \\
&\qquad \left. + \frac{1}{2} \cdot \Pr^{\mathsf{Exp}^{\mathsf{sPKE\text{-}IND\text{-}CCA}}_{\mathsf{sPKE},\mathcal{A}}}[b' = 1 \mid b = 1] \right) - 1 \\
&= \Pr^{\mathsf{Exp}^{\mathsf{sPKE\text{-}IND\text{-}CCA}}_{\mathsf{sPKE},\mathcal{A}}}[b' = 1 \mid b = 1] - \Pr^{\mathsf{Exp}^{\mathsf{sPKE\text{-}IND\text{-}CCA}}_{\mathsf{sPKE},\mathcal{A}}}[b' = 1 \mid b = 0] \\
&= \Pr^{\mathsf{Exp}^{\mathsf{sPKE\text{-}IND\text{-}CCA}}_{\mathsf{sPKE},\mathcal{A}}}[b' = 1 \mid b = 1] - \Pr^{H_{1,1}}[b' = 1] \\
&\qquad + \Pr^{H_{1,1}}[b' = 1] - \Pr^{H_{1,0}}[b' = 1] + \Pr^{H_{1,0}}[b' = 1] - \Pr^{H_{0,0}}[b' = 1] \\
&\qquad + \Pr^{H_{0,0}}[b' = 1] - \Pr^{\mathsf{Exp}^{\mathsf{sPKE\text{-}IND\text{-}CCA}}_{\mathsf{sPKE},\mathcal{A}}}[b' = 1 \mid b = 0] \\
&\leq \mathsf{Adv}^{\mathsf{NIZK\text{-}ZK}}_{\mathsf{NIZK},\mathcal{A}'_{\mathsf{ZK}}} + \mathsf{Adv}^{\mathsf{NIZK\text{-}ZK}}_{\mathsf{NIZK},\mathcal{A}''_{\mathsf{ZK}}} + 2 \cdot \mathsf{Adv}^{\mathsf{DDH}}_{g,\mathcal{A}'_{\mathsf{DDH}}} + 2 \cdot \mathsf{Adv}^{\mathsf{DDH}}_{g,\mathcal{A}''_{\mathsf{DDH}}} \\
&\qquad + \Pr^{H_{0,0}}[B] + \Pr^{H_{1,0}}[B] + 2^{3-\kappa}.
\end{aligned}
$$

Let $\mathcal{A}_{\mathsf{ZK}}$ be the adversary that runs $\mathcal{A}'_{\mathsf{ZK}}$ and $\mathcal{A}''_{\mathsf{ZK}}$ with probability 1/2 each, let $\mathcal{A}_{\mathsf{DDH}}$ run $\mathcal{A}'_{\mathsf{DDH}}$ and $\mathcal{A}''_{\mathsf{DDH}}$ with probability 1/2 each, let $\mathcal{A}_{\mathsf{snd}}$ run $\mathcal{A}^{0,0}_{\mathsf{snd}}$ and $\mathcal{A}^{1,0}_{\mathsf{snd}}$ with probability 1/2 each, and let $\mathcal{A}_{\mathsf{Sig}}$ run $\mathcal{A}^{0,0}_{\mathsf{Sig}}$ and $\mathcal{A}^{1,0}_{\mathsf{Sig}}$ with probability 1/2 each. Combing the result above with equation (6.8), we obtain

$$
\begin{aligned}
\mathsf{Adv}^{\mathsf{sPKE\text{-}IND\text{-}CCA}}_{\mathsf{sPKE},\mathcal{A}} \leq 4 \cdot \mathsf{Adv}^{\mathsf{DDH}}_{g,\mathcal{A}_{\mathsf{DDH}}} &+ 2 \cdot \mathsf{Adv}^{\mathsf{NIZK\text{-}ZK}}_{\mathsf{NIZK},\mathcal{A}_{\mathsf{ZK}}} + 2q_{SD} \cdot \mathsf{Adv}^{\mathsf{NIZK\text{-}sim\text{-}snd}}_{\mathsf{NIZK},\mathcal{A}_{\mathsf{snd}}} \\
&+ 2 \cdot \mathsf{Adv}^{\mathsf{Sig\text{-}EUF\text{-}CMA}}_{\mathsf{Sig},\mathcal{A}_{\mathsf{Sig}}} + \frac{4(q_G + 1)^2 + 8}{2^\kappa - 1},
\end{aligned}
$$

which concludes the proof. □

We finally show that sPKE is IK-CCA secure.

Lemma 6.5.12. *Let* sPKE *be the scheme from Section 6.5.2 and let* $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ *be a pair of probabilistic algorithms such that* $\mathcal{A}_1$ *and* $\mathcal{A}_2$ *together make at most* q_G *queries to* $\mathcal{O}_G$ *and at most* q_{SD} *queries to* $\mathcal{O}_{SD_0}$ *and* $\mathcal{O}_{SD_1}$ *combined. Then, there exist adversaries* $\mathcal{A}_{\mathsf{DDH}}$*,* $\mathcal{A}_{\mathsf{ZK}}$*,* $\mathcal{A}_{\mathsf{snd}}$*,* $\mathcal{A}_{\mathsf{PKE}}$*, and* $\mathcal{A}_{\mathsf{Sig}}$ *(which are all roughly as efficient as emulating an* *execution of* $\mathsf{Exp}^{\mathsf{sPKE\text{-}IK\text{-}CCA}}_{\mathsf{sPKE},\mathcal{A}}$*) such that*

$$\mathsf{Adv}^{\mathsf{sPKE\text{-}IK\text{-}CCA}}_{\mathsf{sPKE},\mathcal{A}} \leq 8 \cdot \mathsf{Adv}^{\mathsf{DDH}}_{g,\mathcal{A}_{\mathsf{DDH}}} + 2 \cdot \mathsf{Adv}^{\mathsf{NIZK\text{-}ZK}}_{\mathsf{NIZK},\mathcal{A}_{\mathsf{ZK}}} + 8q_{SD} \cdot \mathsf{Adv}^{\mathsf{NIZK\text{-}sim\text{-}snd}}_{\mathsf{NIZK},\mathcal{A}_{\mathsf{snd}}}$$
$$+ 2 \cdot \mathsf{Adv}^{\mathsf{PKE\text{-}IND\text{-}CPA}}_{\mathsf{PKE},\mathcal{A}_{\mathsf{PKE}}} + 8 \cdot \mathsf{Adv}^{\mathsf{Sig\text{-}EUF\text{-}CMA}}_{\mathsf{Sig},\mathcal{A}_{\mathsf{Sig}}} + \frac{16(q_G+2)^2+32}{2^\kappa - 1}.$$

Proof. We assume without loss of generality that $\mathcal{A}_2$ does not query the challenge ciphertext c^* to any of its decryption oracles $\mathcal{O}_{SD_0}$ or $\mathcal{O}_{SD_1}$, since doing so can only decrease the advantage. We define hybrid experiments H_0 to H_5 as follows:

- H_0 is identical to $\mathsf{Exp}^{\mathsf{sPKE\text{-}IK\text{-}CCA}}_{\mathsf{sPKE},\mathcal{A}}$, except that the common reference string *crs* is obtained via $(crs, \tau) \leftarrow S_1(1^\kappa)$ (instead of an invocation of $\mathsf{NIZK.Gen}$), and the proof π in the challenge ciphertext c^* is simulated using S_2.

- H_1 is identical to H_0, but when $\mathcal{A}_1$ outputs (m, st), the hybrid computes c_σ not as an encryption of $(ek_{b,1}, ek_{b,2}, \sigma_b)$, but as the encryption of 0^ℓ, where ℓ is the length of the encoding of $(ek_{b,1}, ek_{b,2}, \sigma_b)$ (where the encoding needs to be chosen such that this length is equal for all keys).

- H_2 is identical to H_1, except that for the generation of the challenge ciphertext c^*, the key $ek_{0,1}$ is replaced by $g^{d_{0,1}}$ for a freshly sampled $d_{0,1} \twoheadleftarrow \mathbb{Z}_p^*$.

- H_3 is identical to H_2, except that for the generation of the challenge ciphertext c^*, the key $ek_{0,2}$ is replaced by $g^{d_{0,2}}$ for a freshly sampled $d_{0,2} \twoheadleftarrow \mathbb{Z}_p^*$.

- H_4 is identical to H_3, except that for the generation of the challenge ciphertext c^*, the key $ek_{1,1}$ is replaced by $g^{d_{1,1}}$ for a freshly sampled $d_{1,1} \twoheadleftarrow \mathbb{Z}_p^*$.

- H_5 is identical to H_4, except that for the generation of the challenge ciphertext c^*, the key $ek_{1,2}$ is replaced by $g^{d_{1,2}}$ for a freshly sampled $d_{1,2} \twoheadleftarrow \mathbb{Z}_p^*$.

Note that the view of $\mathcal{A}$ in H_5 is independent from the bit b, which implies

$$\Pr^{H_5}[b' = b] = \frac{1}{2}. \tag{6.9}$$

It can be shown as in the proof of Lemma 6.5.11 that there exist an adversary $\mathcal{A}_{\mathsf{ZK}}$ such that

$$\Pr^{\mathsf{Exp}^{\mathsf{sPKE\text{-}IK\text{-}CCA}}_{\mathsf{sPKE},\mathcal{A}}}[b' = b] - \Pr^{H_0}[b' = b] = \mathsf{Adv}^{\mathsf{NIZK\text{-}ZK}}_{\mathsf{NIZK},\mathcal{A}_{\mathsf{ZK}}}. \tag{6.10}$$

Claim 1. *There exists an adversary $\mathcal{A}_{\mathsf{PKE}}$ such that*

$$\Pr^{H_0}[b' = b] - \Pr^{H_1}[b' = b] = \mathsf{Adv}^{\mathsf{PKE\text{-}IND\text{-}CPA}}_{\mathsf{PKE},\mathcal{A}_{\mathsf{PKE}}}.$$

Proof of claim. When $\mathcal{A}_{\mathsf{PKE}}$ obtains a public key ek from the CPA challenger, it generates all remaining keys itself and emulates H_0 (or H_1) toward $\mathcal{A}$. Note that $\mathcal{A}_{\mathsf{PKE}}$ never needs to decrypt any of the ciphertexts c_σ in the experiment and thus, the missing decryption key is not needed for the emulation. When $\mathcal{A}_1$ outputs (m, st), $\mathcal{A}_{\mathsf{PKE}}$ gives $\big(0^\ell, (ek_{b,1}, ek_{b,2}, \sigma_b)\big)$ to its CPA challenger to obtain a ciphertext c_σ, where ℓ is the length of the encoding of $(ek_{b,1}, ek_{b,2}, \sigma_b)$. The rest is done as in H_0. When $\mathcal{A}_2$ returns a bit b', $\mathcal{A}_{\mathsf{PKE}}$ returns $b'' = 1$ if $b' = b$, and $b'' = 0$ if $b' \neq b$.

Note that if the CPA challenger chooses the bit $b_{\mathsf{CPA}} = 0$, c_σ is an encryption of 0^ℓ, as in H_1, and if $b_{\mathsf{CPA}} = 1$, c_σ is as in H_0. Hence,

$$\begin{aligned}
\Pr^{\mathsf{Exp}^{\mathsf{PKE\text{-}IND\text{-}CPA}}_{\mathsf{PKE},\mathcal{A}_{\mathsf{PKE}}}}[b'' = 1 \mid b_{\mathsf{CPA}} = 0] &= \Pr^{H_1}[b' = b],\\
\Pr^{\mathsf{Exp}^{\mathsf{PKE\text{-}IND\text{-}CPA}}_{\mathsf{PKE},\mathcal{A}_{\mathsf{PKE}}}}[b'' = 1 \mid b_{\mathsf{CPA}} = 1] &= \Pr^{H_0}[b' = b].
\end{aligned}$$

We can therefore conclude

$$\begin{aligned}
\mathsf{Adv}^{\mathsf{PKE\text{-}IND\text{-}CPA}}_{\mathsf{PKE},\mathcal{A}_{\mathsf{PKE}}} &= 2 \cdot \Pr^{\mathsf{Exp}^{\mathsf{PKE\text{-}IND\text{-}CPA}}_{\mathsf{PKE},\mathcal{A}_{\mathsf{PKE}}}}[b'' = b_{\mathsf{CPA}}] - 1\\
&= \Pr^{\mathsf{Exp}^{\mathsf{PKE\text{-}IND\text{-}CPA}}_{\mathsf{PKE},\mathcal{A}_{\mathsf{PKE}}}}[b'' = 0 \mid b_{\mathsf{CPA}} = 0]\\
&\qquad + \Pr^{\mathsf{Exp}^{\mathsf{PKE\text{-}IND\text{-}CPA}}_{\mathsf{PKE},\mathcal{A}_{\mathsf{PKE}}}}[b'' = 1 \mid b_{\mathsf{CPA}} = 1] - 1\\
&= \Pr^{\mathsf{Exp}^{\mathsf{PKE\text{-}IND\text{-}CPA}}_{\mathsf{PKE},\mathcal{A}_{\mathsf{PKE}}}}[b'' = 1 \mid b_{\mathsf{CPA}} = 1]\\
&\qquad - \Pr^{\mathsf{Exp}^{\mathsf{PKE\text{-}IND\text{-}CPA}}_{\mathsf{PKE},\mathcal{A}_{\mathsf{PKE}}}}[b'' = 1 \mid b_{\mathsf{CPA}} = 0]\\
&= \Pr^{H_0}[b' = b] - \Pr^{H_1}[b' = b]. \qquad \diamondsuit
\end{aligned}$$

We define the event B analogous to the events in the proofs of Lemmata 6.5.10 and 6.5.11. As there, one can show for $i \in \{0, \dots, 5\}$ that there exist adversaries $\mathcal{A}^i_{\mathsf{snd}}$ and $\mathcal{A}^i_{\mathsf{Sig}}$ such that

$$\Pr^{H_i}[B] \le q_{SD} \cdot \mathsf{Adv}^{\mathsf{NIZK\text{-}sim\text{-}snd}}_{\mathsf{NIZK},\mathcal{A}^i_{\mathsf{snd}}} + \mathsf{Adv}^{\mathsf{Sig\text{-}EUF\text{-}CMA}}_{\mathsf{Sig},\mathcal{A}^i_{\mathsf{Sig}}} + \frac{2(q_G + 2)^2}{2^\kappa - 1}. \tag{6.11}$$

Claim 2. *There exist adversaries $\mathcal{A}^1_{\mathsf{DDH}}, \ldots, \mathcal{A}^4_{\mathsf{DDH}}$ such that for all $i \in \{1, 3\}$,*

$$\Pr^{H_i}[b' = b] - \Pr^{H_{i+1}}[b' = b] \leq \mathsf{Adv}^{\mathsf{DDH}}_{g, \mathcal{A}^i_{\mathsf{DDH}}} + \Pr^{H_i}[B] + \Pr^{H_{i+1}}[B] + 2^{2-\kappa},$$

and for $i \in \{2, 4\}$,

$$\Pr^{H_i}[b' = b] - \Pr^{H_{i+1}}[b' = b] \leq \mathsf{Adv}^{\mathsf{DDH}}_{g, \mathcal{A}^i_{\mathsf{DDH}}} + 2^{2-\kappa}.$$

Proof of claim. On input (X, Y, T), $\mathcal{A}^1_{\mathsf{DDH}}$ sets $ek_{0,1} \leftarrow X$, $\mathcal{A}^2_{\mathsf{DDH}}$ sets $ek_{0,2} \leftarrow X$, $\mathcal{A}^3_{\mathsf{DDH}}$ sets $ek_{1,1} \leftarrow X$, and $\mathcal{A}^4_{\mathsf{DDH}}$ sets $ek_{1,2} \leftarrow X$. All adversaries generate the remaining keys themselves and emulate H_i (or H_{i+1}) toward $\mathcal{A}$. To emulate the decryption oracles, $\mathcal{A}^1_{\mathsf{DDH}}$ and $\mathcal{A}^3_{\mathsf{DDH}}$ decrypt the second ciphertext component instead of the first one; $\mathcal{A}^2_{\mathsf{DDH}}$ and $\mathcal{A}^4_{\mathsf{DDH}}$ can emulate all oracles perfectly. As in the proof of Lemma 6.5.10, $\mathcal{O}_{SD}$ is also emulated perfectly by $\mathcal{A}^1_{\mathsf{DDH}}$ and $\mathcal{A}^3_{\mathsf{DDH}}$ if the event B does not occur. When $\mathcal{A}_1$ returns (m, st) and if $b = 0$, then $\mathcal{A}^1_{\mathsf{DDH}}$ samples $r \twoheadleftarrow \mathbb{Z}_p^*$, sets

$$c_1 \leftarrow \big(Y^r, T^r, Y, T \cdot m\big),$$

and generates the remaining ciphertext components as in the real experiment. The other adversaries generate the ciphertext components analogously. When $\mathcal{A}_2$ returns a bit b', then $\mathcal{A}^i_{\mathsf{DDH}}$ returns $d = 1$ if $b' = b$ and $d = 0$ if $b' \neq b$ for all $i \in \{1, \ldots, 4\}$.

Consider the case $i = 1$ and note that if (X, Y, T) is a DDH triple, we have $Y = g^s$ and $T = X^s$ for a uniform $s \in \mathbb{Z}_p$, and thus, if $b = 0$,

$$c_1 = \big(Y^r, X^{s \cdot r}, Y, X^s \cdot m\big) = \big(g^{s \cdot r}, (ek_{0,1})^{s \cdot r}, g^s, (ek_{0,1})^s \cdot m\big).$$

If $X \neq 1 \neq Y$, this corresponds to an encryption of m under $ek_{0,1}$, as in H_1. On the other hand, if (X, Y, T) are three uniform group elements and $X \neq 1 \neq Y$, then there are (uniformly distributed) $s, d_{0,1} \in \mathbb{Z}_p^*$ such that $Y = g^s$ and $T = g^{s \cdot d_{0,1}}$. Hence, we have in this case

$$c_1 = \big(g^{s \cdot r}, g^{s \cdot d_{0,1} \cdot r}, g^s, g^{s \cdot d_{0,1}} \cdot m\big) = \big(g^{s \cdot r}, \big(g^{d_{0,1}}\big)^{s \cdot r}, g^s, \big(g^{d_{0,1}}\big)^s \cdot m\big),$$

which corresponds to an encryption under the fresh key $g^{d_{0,1}}$, as in H_2. Further note that if $b = 1$, the emulation, H_1, and H_2 are all equivalent. We therefore have

$$\Pr^{\mathsf{DDH}^{\text{real}}_{g, \mathcal{A}^1_{\mathsf{DDH}}}}[d = 1 \cap \neg B \mid X \neq 1 \neq Y] = \Pr^{H_1}[b' = b \cap \neg B],$$

$$\Pr^{\mathsf{DDH}^{\text{rand}}_{g, \mathcal{A}^1_{\mathsf{DDH}}}}[d = 1 \cap \neg B \mid X \neq 1 \neq Y] = \Pr^{H_2}[b' = b \cap \neg B].$$

As in the proof of Lemma 6.5.11, we obtain

$$\Pr^{\mathsf{DDH}^{\mathrm{real}}_{g,\mathcal{A}^1_{\mathsf{DDH}}}}[d=1] \geq \Pr^{H_1}[b'=b] - \Pr^{H_1}[B] - 2^{1-\kappa}.$$

Moreover,

$$\begin{aligned}
&\Pr^{\mathsf{DDH}^{\mathrm{rand}}_{g,\mathcal{A}^1_{\mathsf{DDH}}}}[d=1]\\
&= \Pr^{\mathsf{DDH}^{\mathrm{rand}}_{g,\mathcal{A}^1_{\mathsf{DDH}}}}[d=1 \cap \neg B \mid X \neq 1 \neq Y] \cdot \Pr^{\mathsf{DDH}^{\mathrm{rand}}_{g,\mathcal{A}^1_{\mathsf{DDH}}}}[X \neq 1 \neq Y]\\
&\qquad + \Pr^{\mathsf{DDH}^{\mathrm{rand}}_{g,\mathcal{A}^1_{\mathsf{DDH}}}}[d=1 \cap (B \cup [X=1 \vee Y=1])]\\
&\leq \Pr^{H_2}[b'=b \cap \neg B] + \Pr^{\mathsf{DDH}^{\mathrm{rand}}_{g,\mathcal{A}^1_{\mathsf{DDH}}}}[B] + \Pr^{\mathsf{DDH}^{\mathrm{rand}}_{g,\mathcal{A}^1_{\mathsf{DDH}}}}[X=1 \vee Y=1]\\
&\leq \Pr^{H_2}[b'=b] + \Pr^{\mathsf{DDH}^{\mathrm{rand}}_{g,\mathcal{A}^1_{\mathsf{DDH}}}}[B] + 2^{1-\kappa}.
\end{aligned}$$

Since the probability of B in $\mathsf{DDH}^{\mathrm{rand}}_{g,\mathcal{A}^1_{\mathsf{DDH}}}$ is equal to its probability in H_2, we conclude

$$\begin{aligned}
\mathsf{Adv}^{\mathsf{DDH}}_{g,\mathcal{A}^1_{\mathsf{DDH}}} &= \Pr^{\mathsf{DDH}^{\mathrm{real}}_{g,\mathcal{A}^1_{\mathsf{DDH}}}}[d=1] - \Pr^{\mathsf{DDH}^{\mathrm{rand}}_{g,\mathcal{A}^1_{\mathsf{DDH}}}}[d=1]\\
&\geq \Pr^{H_1}[b'=b] - \Pr^{H_2}[b'=b] - \Pr^{H_1}[B] - \Pr^{H_2}[B] - 2^{2-\kappa}.
\end{aligned}$$

The proofs for $i \in \{2,3,4\}$ are analogous, where for $i \in \{2,4\}$, the occurrence of B does not matter since the decryption oracle can always be emulated perfectly. $\Diamond$

Using equation (6.10), Claims 1 and 2, and equation (6.9), we obtain

$$\begin{aligned}
&\Pr^{\mathsf{Exp}^{\mathsf{sPKE\text{-}IK\text{-}CCA}}_{\mathsf{sPKE},\mathcal{A}}}[b'=b]\\
&= \Pr^{\mathsf{Exp}^{\mathsf{sPKE\text{-}IK\text{-}CCA}}_{\mathsf{sPKE},\mathcal{A}}}[b'=b] - \Pr^{H_0}[b'=b] + \Pr^{H_0}[b'=b] - \Pr^{H_1}[b'=b]\\
&\qquad + \sum_{i=1}^{4}\Big(\Pr^{H_i}[b'=b] - \Pr^{H_{i+1}}[b'=b]\Big) + \Pr^{H_5}[b'=b]\\
&\leq \mathsf{Adv}^{\mathsf{NIZK\text{-}ZK}}_{\mathsf{NIZK},\mathcal{A}_{\mathsf{ZK}}} + \mathsf{Adv}^{\mathsf{PKE\text{-}IND\text{-}CPA}}_{\mathsf{PKE},\mathcal{A}_{\mathsf{PKE}}} + \sum_{i=1}^{4}\Big(\mathsf{Adv}^{\mathsf{DDH}}_{g,\mathcal{A}^i_{\mathsf{DDH}}} + \Pr^{H_i}[B]\Big)\\
&\qquad + 2^{4-\kappa} + \frac{1}{2}.
\end{aligned}$$

For the adversary $\mathcal{A}_{\mathsf{snd}}$ that runs $\mathcal{A}^1_{\mathsf{snd}}, \ldots, \mathcal{A}^4_{\mathsf{snd}}$ with probability $1/4$ each, and the adversary $\mathcal{A}_{\mathsf{Sig}}$ that runs $\mathcal{A}^1_{\mathsf{Sig}}, \ldots, \mathcal{A}^4_{\mathsf{Sig}}$ with probability $1/4$ each, we obtain using equation (6.11),

$$\sum_{i=1}^{4} \Pr^{H_i}[B] \leq 4q_{SD} \cdot \mathsf{Adv}^{\mathsf{NIZK\text{-}sim\text{-}snd}}_{\mathsf{NIZK},\mathcal{A}_{\mathsf{snd}}} + 4 \cdot \mathsf{Adv}^{\mathsf{Sig\text{-}EUF\text{-}CMA}}_{\mathsf{Sig},\mathcal{A}_{\mathsf{Sig}}} + \frac{8(q_G+2)^2}{2^\kappa - 1}.$$

Further defining $\mathcal{A}_{\mathsf{DDH}}$ as running $\mathcal{A}^1_{\mathsf{DDH}}, \ldots, \mathcal{A}^4_{\mathsf{DDH}}$ with probability $1/4$ each yields

$$\begin{aligned}
\mathsf{Adv}^{\mathsf{sPKE\text{-}IK\text{-}CCA}}_{\mathsf{sPKE},\mathcal{A}} &\leq 2 \cdot \Pr^{\mathsf{Exp}^{\mathsf{sPKE\text{-}IK\text{-}CCA}}_{\mathsf{sPKE},\mathcal{A}}}[b' = b] - 1 \\
&\leq 2 \cdot \mathsf{Adv}^{\mathsf{NIZK\text{-}ZK}}_{\mathsf{NIZK},\mathcal{A}_{\mathsf{ZK}}} + 2 \cdot \mathsf{Adv}^{\mathsf{PKE\text{-}IND\text{-}CPA}}_{\mathsf{PKE},\mathcal{A}_{\mathsf{PKE}}} + 8 \cdot \mathsf{Adv}^{\mathsf{DDH}}_{g,\mathcal{A}_{\mathsf{DDH}}} \\
&\quad + 8q_{SD} \cdot \mathsf{Adv}^{\mathsf{NIZK\text{-}sim\text{-}snd}}_{\mathsf{NIZK},\mathcal{A}_{\mathsf{snd}}} + 8 \cdot \mathsf{Adv}^{\mathsf{Sig\text{-}EUF\text{-}CMA}}_{\mathsf{Sig},\mathcal{A}_{\mathsf{Sig}}} \\
&\quad + \frac{16(q_G+2)^2}{2^\kappa - 1} + 2^{5-\kappa}.
\end{aligned}$$

Observing that $2^{5-\kappa} \leq \frac{32}{2^\kappa - 1}$ concludes the proof. □

6.6 Construction of an ACE Scheme

6.6.1 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. We base our construction on an sPKE scheme, which already has many important properties needed for a secure ACE scheme. A syntactical difference between sPKE and ACE schemes is that the key generation of the former on every invocation produces a fresh key pair, while the latter schemes allow the generation of keys for a given role. To bind key pairs to some role $i \in [n]$, we use the output of a pseudorandom function on input i as the randomness for the sPKE key generation. For role-respecting security, we have to ensure that an adversary can only produce ciphertexts for keys obtained from the key generation oracle. This is achieved by signing all keys with a signing key generated at setup. To prevent malleability attacks as the ones described in Section 6.3, the encryption algorithm additionally signs all ciphertexts with a separate signing key that is tied to the encryption key. To maintain anonymity, the signatures are not part of the ciphertext but the encrypters prove in zero-knowledge that they know such signatures. Finally, the modification detection simply checks whether the ciphertexts (without the NIZK proofs) are equal. Intuitively, this is sufficient since we assume the underlying sPKE scheme to be CCA secure, which implies that it is not possible to meaningfully modify a given ciphertext. Hence, a ciphertext is either equal to an existing one (and thus detected by the algorithm) or a fresh encryption.

Our construction. 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

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

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 $t = \texttt{sen}$, 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 $t = \texttt{sen}$, 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 $t = \texttt{rec}$, it outputs the decryption key $dk_i^{\mathsf{ACE}} := dk_i^{\mathsf{sPKE}}$.

Encryption: The algorithm $\mathsf{ACE.Enc}$ on input 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 message $m \in \mathcal{M}^{\mathsf{ACE}}$, samples randomness r and computes

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

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

Sanitization: On input sanitizer parameters $sp^{\mathsf{ACE}} = \big(sp^{\mathsf{sPKE}}, vk^{\mathsf{Sig}}, crs^{\mathsf{NIZK}}\big)$ and a ciphertext $c = \big(\tilde{c}, \pi^{\mathsf{NIZK}}\big)$, $\mathsf{ACE.San}$ 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 $\perp$.

Decryption: On input a decryption key dk_j^{ACE} and a sanitized ciphertext c', $\mathsf{ACE.Dec}$ 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.

We first show that our scheme is correct and strongly detectable.

Proposition 6.6.1. *Let* ACE *be the scheme from above. Then,* ACE *is perfectly correct, i.e.,* $\mathsf{Adv}_{\mathsf{ACE},\mathcal{A}}^{\mathsf{ACE\text{-}CORR}} = 0$ *for all* $\mathcal{A}$. *Moreover, if* F *is pseudorandom and* sPKE *is unrestricted strongly robust, then* ACE *is strongly detectable.*

Proof. Perfect correctness follows from the perfect correctness of the sPKE and signature schemes and the perfect completeness of the NIZK proof system.

To prove strong detectability, let $\mathcal{A}$ be a probabilistic algorithm. We assume without loss of generality that $\mathcal{A}$ returns (m, r, i, j) with $P(i, j) = 0$ since doing otherwise can only reduce the advantage. Let $H_0 := \mathsf{Exp}^{\mathsf{ACE\text{-}sDTCT}}_{\mathsf{ACE},\mathcal{A}}$, 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 strong detectability game, i.e.,

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

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* $\mathcal{A}^{\mathcal{O}(\cdot)}_{\mathsf{PRF}}$ *such that*

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

Proof of claim. Consider $\mathcal{A}^{\mathcal{O}(\cdot)}_{\mathsf{PRF}}$ 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,

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

We now construct a winner $\mathcal{A}_{\mathsf{rob}}$ 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, r, 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. Let $\tilde{r}$ be the randomness used by $\mathsf{ACE.Enc}(ek_i^{\mathsf{ACE}}, m; r)$ for the algorithm $\mathsf{sPKE.Enc}$. Then, $\mathcal{A}_{\mathsf{rob}}$ computes

$c \leftarrow \mathsf{sPKE.Enc}\big(ek_i^{\mathsf{sPKE}}, m; \tilde{r}\big)$, 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 $\mathsf{sPKE.Dec}\big(dk_i^{\mathsf{sPKE}}, \mathsf{sPKE.San}\big(sp^{\mathsf{sPKE}}, c\big)\big) \neq \bot$. Hence, $\mathcal{A}_{\mathsf{rob}}$ wins the robustness game if and only if $\mathcal{A}$ wins the strong detectability game in H_1. Using Claim 1, we can therefore conclude

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

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

Theorem 6.6.2. *If F is pseudorandom,* NIZK *is zero-knowledge and extractable,* Sig *is EUF-CMA secure, and the scheme* sPKE *is IND-CCA, IK-CCA, SAN-CCA, USROB, and UPD-CTXT secure and has negligible encryption-key collision probability, then the scheme* ACE *from above is PRV-CCA, sANON-CCA, SAN-CCA, UDEC, and RR secure, and has NDTCT-FENC.*

We first show that our scheme satisfies the privacy definition from Definition 6.4.2 if the underlying sanitizable public-key encryption scheme is IND-CCA secure, the PRF is secure, and the NIZK is zero-knowledge.

Lemma 6.6.3. *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 $\mathsf{Exp}^{\mathsf{ACE\text{-}PRV\text{-}ANON\text{-}CCA}}_{\mathsf{ACE},\mathcal{A}}$) such that*

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

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 $H_0 := \mathsf{Exp}^{\mathsf{ACE\text{-}PRV\text{-}ANON\text{-}CCA}}_{\mathsf{ACE},\mathcal{A}}$ 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 Proposition 6.6.1.

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

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

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* $\mathcal{A}_{\mathsf{ZK}}^{\mathcal{O}(\cdot,\cdot)}$ *such that*

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

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. ◇

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 $i_{q_0} \leftarrow \bot$, $k_q \leftarrow 1$, 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 $(j, \texttt{rec})$, obtain a decryption key via $\mathcal{O}_G$.

$\mathcal{O}_{SD}(\cdot,\cdot)$**:** On query $\big(j, c = (\tilde{c}, \pi^{\mathsf{NIZK}})\big)$, 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 $\mathsf{NIZK.Ver}\big(crs^{\mathsf{NIZK}}, x := (vk^{\mathsf{Sig}}, \tilde{c},), \pi^{\mathsf{NIZK}}\big) = 1$, 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 $\pi^{\mathsf{NIZK}} \leftarrow S_2^{\mathsf{NIZK}}\big(crs^{\mathsf{NIZK}}, \tau^{\mathsf{NIZK}}, x := (vk^{\mathsf{Sig}}, \tilde{c}^*)\big)$, and give st and the ciphertext $c^* := (\tilde{c}^*, \pi^{\mathsf{NIZK}})$ to $\mathcal{A}_2$. Emulate the oracles for $\mathcal{A}_2$ as follows.

$\mathcal{O}_G(\cdot,\cdot)$**:** On query $(i, \texttt{sen})$, 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 $(j, \texttt{rec})$, obtain a decryption key from $\mathcal{O}_G$.

$\mathcal{O}_{SD^*}(\cdot,\cdot)$**:** On query $\big(j, c = (\tilde{c}, \pi^{\mathsf{NIZK}})\big)$, run $\mathsf{ACE.DMod}(sp^{\mathsf{ACE}}, c^*, c)$. If the output is 1, return $\texttt{test}$. 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 $\mathsf{NIZK.Ver}\big(crs^{\mathsf{NIZK}}, x := (vk^{\mathsf{Sig}}, \tilde{c},), \pi^{\mathsf{NIZK}}\big) = 1$, return m, otherwise, return $\perp$.

Note that we never query the decryption oracle of the IND-CCA experiment on $\tilde{c}^*$ because we return $\texttt{test}$ 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 $(i_0, \texttt{sen})$ 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 $(i_0, \texttt{rec})$ 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

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

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

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

Combining this with Claims 1 and 2, we can conclude

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

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.

Lemma 6.6.4. *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* $(\cdot, \texttt{sen})$ *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* $\mathsf{Exp}_{\mathsf{ACE},\mathcal{A}}^{\mathsf{ACE\text{-}PRV\text{-}ANON\text{-}CCA}}$*) such that*

$$\begin{aligned}\mathsf{Adv}_{\mathsf{ACE},\mathcal{A}}^{\mathsf{ACE\text{-}sANON\text{-}CCA}} &\leq 2 \cdot \mathsf{Adv}_{F,\mathcal{A}_{\mathsf{PRF}}}^{\mathsf{PRF}} + 2 \cdot \mathsf{Adv}_{\mathsf{NIZK},\mathcal{A}_{\mathsf{ZK}}}^{\mathsf{NIZK\text{-}ZK}} \\ &\qquad + (q_S + q_D + 1)^2 \cdot \mathsf{Adv}_{\mathsf{sPKE},\mathcal{A}_{\mathsf{sPKE}}}^{\mathsf{sPKE\text{-}IK\text{-}CCA}}.\end{aligned}$$

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 Lemma 6.6.3, let $H_0 := \mathsf{Exp}_{\mathsf{ACE},\mathcal{A}}^{\mathsf{ACE\text{-}PRV\text{-}ANON\text{-}CCA}}$, 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 Lemma 6.6.3 shows that there exist $\mathcal{A}_{\mathsf{PRF}}$ and $\mathcal{A}_{\mathsf{ZK}}$ such that

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

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 Lemma 6.6.3, 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 $(sp^{\mathsf{sPKE}}, ek_0^{\mathsf{sPKE}}, ek_1^{\mathsf{sPKE}})$, the adversary $\mathcal{A}_{\mathsf{sPKE}}$ initializes $i_{q_0}, i_{q_1} \leftarrow \bot$, $k_q \leftarrow 1$, chooses $q_0, q_1 \twoheadleftarrow \{0, \dots, q_S + q_D\}$ uniformly at random, runs $\big(vk^{\mathsf{Sig}}, sk^{\mathsf{Sig}}\big) \leftarrow \mathsf{Sig.Gen}(1^\kappa)$,

$\big(crs^{\mathsf{NIZK}}, \tau^{\mathsf{NIZK}}\big) \leftarrow S_1^{\mathsf{NIZK}}(1^\kappa)$, 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 $ek_i^{\mathsf{ACE}} := \big(vk^{\mathsf{Sig}}, ek_i^{\mathsf{sPKE}}, vk_i^{\mathsf{Sig}}, sk_i^{\mathsf{Sig}}, \sigma_i^{\mathsf{Sig}}, crs^{\mathsf{NIZK}}\big)$ as done in H_2, 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 $\perp$. 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), $\mathcal{A}_{\mathsf{sPKE}}$ 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 $\perp$.

Note that $\mathcal{A}_{\mathsf{sPKE}}$ never queries any of the decryption oracles of the IK-CCA experiment on $\tilde{c}^*$ because we return `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 Lemma 6.6.3 that

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

We next prove sanitization security of our scheme.

Lemma 6.6.5. *Let* ACE *be the scheme from above, and 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*

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

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.,

$$W_{\mathsf{ACE}} := \big[b' = b \ \wedge\ c'_0 \neq \bot \neq c'_1 \ \wedge\ \forall j \in J\ m_{0,j} = m_{1,j} = \bot\big].$$

Similarly as in the proof of Lemma 6.6.3, it can be shown that there exist $\mathcal{A}_{\mathsf{PRF}}$ and $\mathcal{A}_{\mathsf{ZK}_1}$ such that

$$\Pr^{H_0}[W_{\mathsf{ACE}}] - \Pr^{H_2}[W_{\mathsf{ACE}}] = \mathsf{Adv}^{\mathsf{PRF}}_{F,\mathcal{A}_{\mathsf{PRF}}} + \mathsf{Adv}^{\mathsf{NIZK\text{-}ext}_1}_{\mathsf{NIZK},\mathcal{A}_{\mathsf{ZK}_1}}. \tag{6.12}$$

Let H_3 be as 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

$$\Pr^{H_3}[W_{\mathsf{ACE}}] = \Pr^{H_2}[W_{\mathsf{ACE}}]. \tag{6.13}$$

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 $\big(x_{\tilde{b}}, \pi^{\mathsf{NIZK}}_{\tilde{b}}\big)$. If the $\tilde{b}$'s extraction fails, $\mathcal{A}_{\mathsf{ZK}_2}$ wins the extraction game. Hence,

$$\Pr^{H_3}[B_E] \leq 2 \cdot \mathsf{Adv}^{\mathsf{NIZK\text{-}ext}_2}_{\mathsf{NIZK},\mathcal{A}_{\mathsf{ZK}_2}}. \tag{6.14}$$

For $\tilde{b} \in \{0,1\}$, let $B_{S,\tilde{b}}$ be the event that $(x_{\tilde{b}}, w_{\tilde{b}}) \in R$ and $ek^{\mathsf{sPKE}}_{i_{\tilde{b}}}$ 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*

$$\Pr^{H_3}[B_S] \leq 2 \cdot \mathsf{Adv}^{\mathsf{Sig\text{-}EUF\text{-}CMA}}_{\mathsf{Sig},\mathcal{A}_{\mathsf{Sig}}}. \tag{6.15}$$

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 $(i, \texttt{sen})$ 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

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

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

$$m_{k,j} \leftarrow \begin{cases} m_k, & ek_j^{\mathsf{sPKE}} = ek_{i_k}^{\mathsf{sPKE}} \text{ for } \big(ek_j^{\mathsf{sPKE}}, dk_j^{\mathsf{sPKE}}\big) = \\ & \qquad \mathsf{sPKE.Gen}\big(msk^{\mathsf{sPKE}}; U([j,0])\big), \\ \bot, & \text{else,} \end{cases} \tag{6.16}$$

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 $\mathsf{ACE.Dec}\big(\mathsf{ACE.Gen}(msk, j, \mathtt{rec}), c'_k\big) = m_k$ by the correctness of the sPKE scheme, 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 $V := V_0 \cap V_1$,

$$\begin{aligned} \Pr^{H_3}[W_{\mathsf{ACE}} \mid V \cap \neg B_E \cap \neg B_S] - \Pr^{H_4}[W_{\mathsf{ACE}} \mid V \cap \neg B_E \cap \neg B_S] \\ \leq 2(q_{R_1} + q_{R_2})\mathsf{Adv}_{\mathsf{sPKE},\mathcal{A}_{\mathsf{rob}}}^{\mathsf{sPKE\text{-}USROB}}, \end{aligned} \tag{6.17}$$

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.[5]

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 $q_0, q_1 \twoheadleftarrow \{1, \ldots, q_{S_1} + q_{R_1} + q_{D_1}\}$

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

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 $sp^{\mathsf{ACE}} := \big(sp^{\mathsf{sPKE}}, vk^{\mathsf{Sig}}, crs^{\mathsf{NIZK}}\big)$ 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 $i \notin \{i_{q_0}, i_{q_1}\}$, generate an encryption key $\big(vk^{\mathsf{Sig}}, ek_i^{\mathsf{sPKE}}, sk_i^{\mathsf{Sig}}, \sigma_i^{\mathsf{Sig}}, crs^{\mathsf{NIZK}}\big)$ as H_4, 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} and set $i_{q_l} \leftarrow i$. In both cases, set $k_q \leftarrow k_q + 1$ at the end.

On query $(j, \mathtt{rec})$, 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 $\big(j, c = \big(\tilde{c}, \pi^{\mathsf{NIZK}}\big)\big)$, 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 $i_{q_l} \leftarrow j$, if $\mathsf{NIZK.Ver}\big(crs^{\mathsf{NIZK}}, x := \big(vk^{\mathsf{Sig}}, \tilde{c}, \big), \pi^{\mathsf{NIZK}}\big) = 0$, 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 $\big(c_0 = \big(\tilde{c}_0, \pi_0^{\mathsf{NIZK}}\big), c_1 = \big(\tilde{c}_1, \pi_1^{\mathsf{NIZK}}\big), st\big)$, $\mathcal{A}_{\mathsf{sPKE}}$ 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

$$\Pr^{\mathsf{Exp}^{\mathsf{sPKE\text{-}SAN\text{-}CCA}}_{\mathsf{sPKE},\mathcal{A}_{\mathsf{sPKE}}}}[Q \mid V \cap \neg B_E \cap \neg B_S] \geq 1/(q_{S_1} + q_{R_1} + q_{D_1})^2. \tag{6.18}$$

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 $\big(st, c'_{\tilde{b}}\big)$ 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.,

$$W_{\mathsf{sPKE}} := \big[b'' = \tilde{b} \ \wedge \ \exists j, j' \in \{0,1\} \ m^{\mathsf{sPKE}}_{0,j} \neq \bot \neq m^{\mathsf{sPKE}}_{1,j'}))\big],$$

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 $m^{\mathsf{sPKE}}_{0,0} = m^{\mathsf{sPKE}}_{1,0} = \bar{m} \neq \bot$ by the correctness of sPKE, and thus

$$\Pr^{\mathsf{Exp}^{\mathsf{sPKE\text{-}SAN\text{-}CCA}}_{\mathsf{sPKE},\mathcal{A}_{\mathsf{sPKE}}}}[W_{\mathsf{sPKE}} \mid \neg(Q \cap V \cap \neg B_E \cap \neg B_S)] = \frac{1}{2}. \tag{6.19}$$

Next consider the case that the event $Q \cap V \cap \neg B_E \cap \neg B_S$ 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 $\bot$. Moreover, if $\mathcal{A}$ wins, we have $m^{H_4}_{0,j} = m^{H_4}_{1,j} = \bot$ for all $j \in J^{H_4}$, where the messages here refer to the ones in H_4, generated according to (6.16), and J^{H_4} is the set of all j such that $\mathcal{A}_1$ or $\mathcal{A}_2$ issued the query $(j, \mathtt{rec})$ to the oracle $\mathcal{O}_G$. Therefore, $\mathcal{O}_G$ is never gets a query for which it returns $\bot$ in this case. The event $Q \cap V \cap \neg B_E$ implies that the ciphertexts are encryptions of some message under ek^{sPKE}_0 or ek^{sPKE}_1. Correctness of sPKE now implies that $m^{\mathsf{sPKE}}_{0,0} \neq \bot \neq m^{\mathsf{sPKE}}_{1,0}$, i.e., the winning condition for $\mathcal{A}_{\mathsf{sPKE}}$ is satisfied. We can conclude that

$$\begin{aligned}\Pr^{\mathsf{Exp}^{\mathsf{sPKE\text{-}SAN\text{-}CCA}}_{\mathsf{sPKE},\mathcal{A}_{\mathsf{sPKE}}}}&[W_{\mathsf{sPKE}} \mid Q \cap V \cap \neg B_E \cap \neg B_S] \\ &\geq \Pr^{H_4}[W_{\mathsf{ACE}} \mid V \cap \neg B_E \cap \neg B_S].\end{aligned} \tag{6.20}$$

For brevity, we introduce

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

Putting our results together, we obtain

$$\begin{aligned}&\Pr^{\mathsf{Exp}^{\mathsf{sPKE\text{-}SAN\text{-}CCA}}_{\mathsf{sPKE},\mathcal{A}_{\mathsf{sPKE}}}}[W_{\mathsf{sPKE}}] \\ &= \Pr^{\mathsf{Exp}^{\mathsf{sPKE\text{-}SAN\text{-}CCA}}_{\mathsf{sPKE},\mathcal{A}_{\mathsf{sPKE}}}}[W_{\mathsf{sPKE}} \mid Q \cap V \cap \neg B_E \cap \neg B_S] \cdot p_G \\ &\qquad + \Pr^{\mathsf{Exp}^{\mathsf{sPKE\text{-}SAN\text{-}CCA}}_{\mathsf{sPKE},\mathcal{A}_{\mathsf{sPKE}}}}[W_{\mathsf{sPKE}} \mid \neg(Q \cap V \cap \neg B_E \cap \neg B_S)] \cdot (1 - p_G) \\ &\overset{(6.19)}{=} \Pr^{\mathsf{Exp}^{\mathsf{sPKE\text{-}SAN\text{-}CCA}}_{\mathsf{sPKE},\mathcal{A}_{\mathsf{sPKE}}}}[W_{\mathsf{sPKE}} \mid Q \cap V \cap \neg B_E \cap \neg B_S] \cdot p_G + \frac{1}{2}(1 - p_G).\end{aligned}$$

This implies

$$\begin{aligned}
&\Pr^{\mathsf{Exp}^{\mathsf{sPKE\text{-}SAN\text{-}CCA}}_{\mathsf{sPKE},\mathcal{A}_{\mathsf{sPKE}}}}[W_{\mathsf{sPKE}} \mid Q \cap V \cap \neg B_E \cap \neg B_S] \\
&= \frac{1}{p_G}\left[\Pr^{\mathsf{Exp}^{\mathsf{sPKE\text{-}SAN\text{-}CCA}}_{\mathsf{sPKE},\mathcal{A}_{\mathsf{sPKE}}}}[W_{\mathsf{sPKE}}] - \frac{1}{2}(1-p_G)\right] \\
&= \frac{1}{2\,p_G} \cdot \mathsf{Adv}^{\mathsf{sPKE\text{-}SAN\text{-}CCA}}_{\mathsf{sPKE},\mathcal{A}_{\mathsf{sPKE}}} + \frac{1}{2}.
\end{aligned} \tag{6.21}$$

Furthermore, using that we defined $H_0 = \mathsf{Exp}^{\mathsf{ACE\text{-}SAN\text{-}CCA}}_{\mathsf{ACE},\mathcal{A}}$, the definitions of W_{ACE} and the sanitization advantage, and equations (6.12) and (6.13), we obtain

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

Since the events 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 together with our results from above implies

$$\begin{aligned}
\Pr^{H_3}[W_{\mathsf{ACE}}] &= \Pr^{H_3}[W_{\mathsf{ACE}} \cap B_E] + \Pr^{H_3}[W_{\mathsf{ACE}} \cap \neg B_E \cap B_S] \\
&\qquad + \Pr^{H_3}[W_{\mathsf{ACE}} \cap \neg B_E \cap \neg B_S] \\
&\le \Pr^{H_3}[B_E] + \Pr^{H_3}[B_S] + \Pr^{H_3}[W_{\mathsf{ACE}} \cap \neg B_E \cap \neg B_S] \\
&\overset{(6.14),(6.15)}{\le} 2 \cdot \mathsf{Adv}^{\mathsf{NIZK\text{-}ext_2}}_{\mathsf{NIZK},\mathcal{A}_{\mathsf{ZK}_2}} + 2 \cdot \mathsf{Adv}^{\mathsf{Sig\text{-}EUF\text{-}CMA}}_{\mathsf{Sig},\mathcal{A}_{\mathsf{Sig}}} \\
&\qquad + \Pr^{H_3}[W_{\mathsf{ACE}} \cap \neg B_E \cap \neg B_S].
\end{aligned}$$

Note that the event W_{ACE} implies $c_0' \neq \bot$ and $c_1' \neq \bot$ and thus also the event V because if the verification fails, the algorithm $\mathsf{ACE.San}$ always returns $\bot$. Therefore, we can conclude that $W_{\mathsf{ACE}} = W_{\mathsf{ACE}} \cap V$. We thus have

$$\begin{aligned}
&\Pr^{H_3}[W_{\mathsf{ACE}} \cap \neg B_E \cap \neg B_S] \\
&= \Pr^{H_3}[W_{\mathsf{ACE}} \cap V \cap \neg B_E \cap \neg B_S] \\
&= \Pr^{H_3}[W_{\mathsf{ACE}} \mid V \cap \neg B_E \cap \neg B_S] \cdot \Pr^{H_3}[V \cap \neg B_E \cap \neg B_S].
\end{aligned}$$

Using results from above, we further have

$$\begin{aligned}
&\Pr^{H_3}[W_{\mathsf{ACE}} \mid V \cap \neg B_E \cap \neg B_S] \\
&\overset{(6.17)}{\leq} \underbrace{\Pr^{H_4}[W_{\mathsf{ACE}} \mid V \cap \neg B_E \cap \neg B_S]}_{\overset{(6.20)}{\leq} \Pr^{\mathsf{Exp}^{\mathsf{sPKE\text{-}SAN\text{-}CCA}}_{\mathsf{sPKE},\mathcal{A}_{\mathsf{sPKE}}}}[W_{\mathsf{sPKE}} \mid Q \cap V \cap \neg B_E \cap \neg B_S]} + 2(q_{R_1} + q_{R_2}) \cdot \mathsf{Adv}^{\mathsf{sPKE\text{-}USROB}}_{\mathsf{sPKE},\mathcal{A}_{\mathsf{rob}}} \\
&\overset{(6.21)}{\leq} \frac{1}{2\,p_G} \cdot \mathsf{Adv}^{\mathsf{sPKE\text{-}SAN\text{-}CCA}}_{\mathsf{sPKE},\mathcal{A}_{\mathsf{sPKE}}} + \frac{1}{2} + 2(q_{R_1} + q_{R_2}) \cdot \mathsf{Adv}^{\mathsf{sPKE\text{-}USROB}}_{\mathsf{sPKE},\mathcal{A}_{\mathsf{rob}}}.
\end{aligned}$$

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

$$\begin{aligned}
\frac{\Pr^{H_3}[V \cap \neg B_E \cap \neg B_S]}{p_G} &= \frac{\Pr^{\mathsf{Exp}^{\mathsf{sPKE\text{-}SAN\text{-}CCA}}_{\mathsf{sPKE},\mathcal{A}_{\mathsf{sPKE}}}}[V \cap \neg B_E \cap \neg B_S]}{\Pr^{\mathsf{Exp}^{\mathsf{sPKE\text{-}SAN\text{-}CCA}}_{\mathsf{sPKE},\mathcal{A}_{\mathsf{sPKE}}}}[Q \cap V \cap \neg B_E \cap \neg B_S]} \\
&= \left(\Pr^{\mathsf{Exp}^{\mathsf{sPKE\text{-}SAN\text{-}CCA}}_{\mathsf{sPKE},\mathcal{A}_{\mathsf{sPKE}}}}[Q \mid V \cap \neg B_E \cap \neg B_S]\right)^{-1} \\
&\overset{(6.18)}{\leq} (q_{S_1} + q_{R_1} + q_{D_1})^2.
\end{aligned}$$

Therefore,

$$\begin{aligned}
\Pr^{H_3}[W_{\mathsf{ACE}} \cap \neg B_E \cap \neg B_S] &\leq \frac{\Pr^{H_3}[V \cap \neg B_E \cap \neg B_S]}{2\,p_G} \cdot \mathsf{Adv}^{\mathsf{sPKE\text{-}SAN\text{-}CCA}}_{\mathsf{sPKE},\mathcal{A}_{\mathsf{sPKE}}} \\
&\quad + \frac{1}{2} + 2(q_{R_1} + q_{R_2}) \cdot \mathsf{Adv}^{\mathsf{sPKE\text{-}USROB}}_{\mathsf{sPKE},\mathcal{A}_{\mathsf{rob}}} \\
&\leq \frac{1}{2} \cdot (q_{S_1} + q_{R_1} + q_{D_1})^2 \cdot \mathsf{Adv}^{\mathsf{sPKE\text{-}SAN\text{-}CCA}}_{\mathsf{sPKE},\mathcal{A}_{\mathsf{sPKE}}} \\
&\quad + \frac{1}{2} + 2(q_{R_1} + q_{R_2}) \cdot \mathsf{Adv}^{\mathsf{sPKE\text{-}USROB}}_{\mathsf{sPKE},\mathcal{A}_{\mathsf{rob}}}.
\end{aligned}$$

This implies

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

and concludes the proof. □

We next prove non-detection of fresh encryptions, which follows from ciphertext unpredictability of the underlying sPKE scheme and the security of the PRF.

Lemma 6.6.6. *Let* ACE *be the scheme from above and let* $\mathcal{A}$ *be an attacker on the non-detection of fresh encryptions 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* $\mathsf{Exp}^{\mathsf{ACE\text{-}NDTCT\text{-}FENC}}_{\mathsf{ACE},\mathcal{A}}$*) such that*

$$\mathsf{Adv}^{\mathsf{ACE\text{-}NDTCT\text{-}FENC}}_{\mathsf{ACE},\mathcal{A}} \leq \mathsf{Adv}^{\mathsf{PRF}}_{F,\mathcal{A}_{\mathsf{PRF}}} + (q+1) \cdot \mathsf{Adv}^{\mathsf{sPKE\text{-}UPD\text{-}CTXT}}_{\mathsf{sPKE},\mathcal{A}_{\mathsf{sPKE}}}.$$

Proof. Let $H_0 := \mathsf{Exp}^{\mathsf{ACE\text{-}NDTCT\text{-}FENC}}_{\mathsf{ACE},\mathcal{A}}$ and H_1 be as H_0 where F_K is replaced by a truly uniform random function U. As in the proof of Lemma 6.6.3, 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 $\left(sp^{\mathsf{sPKE}}, ek^{\mathsf{sPKE}}, dk^{\mathsf{sPKE}}\right)$, sets $i_{q_0} \leftarrow \perp$, $k_q \leftarrow 1$, samples $q_0 \twoheadleftarrow \{0,\dots,q\}$, runs $\left(vk^{\mathsf{Sig}}, sk^{\mathsf{Sig}}\right) \leftarrow \mathsf{Sig.Gen}(1^\kappa)$, $crs^{\mathsf{NIZK}} \leftarrow \mathsf{NIZK.Gen}(1^\kappa)$, and gives $sp^{\mathsf{ACE}} := \left(sp^{\mathsf{sPKE}}, vk^{\mathsf{Sig}}, crs^{\mathsf{NIZK}}\right)$ to $\mathcal{A}$. It emulates the oracle $\mathcal{O}_G$ for $\mathcal{A}_1$ as follows. On query (i,t), if $k_q \neq q_0$ and $i \neq i_{q_0}$, then generate an encryption key $ek_i^{\mathsf{ACE}} := \big(vk^{\mathsf{Sig}}, ek_i^{\mathsf{sPKE}}, vk_i^{\mathsf{Sig}}, sk_i^{\mathsf{Sig}}, \sigma_i^{\mathsf{Sig}}, crs^{\mathsf{NIZK}}\big)$ and a decryption key $dk_i^{\mathsf{ACE}} := dk_i^{\mathsf{sPKE}}$ as H_1 does, where $\left(ek_i^{\mathsf{sPKE}}, dk_i^{\mathsf{sPKE}}\right)$ is obtained via $\mathcal{O}_G$ and remembered for future queries. Return ek_i^{ACE} if $t = \texttt{sen}$, and dk_i^{ACE} if $t = \texttt{rec}$. 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 $\left(m, i, c = \left(\tilde{c}, \pi^{\mathsf{NIZK}}\right)\right)$, $\mathcal{A}_{\mathsf{sPKE}}$ returns $(m, \tilde{c})$.

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

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

Hence, we can conclude

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

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

Lemma 6.6.7. *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*

$$\begin{aligned}\mathsf{Adv}^{\mathsf{ACE\text{-}UDEC}}_{\mathsf{ACE},\mathcal{A}} \le \mathsf{Adv}^{\mathsf{PRF}}_{F,\mathcal{A}_{\mathsf{PRF}}} &+ \mathsf{Adv}^{\mathsf{NIZK\text{-}ext}_1}_{\mathsf{NIZK},\mathcal{A}_{\mathsf{ZK}_1}} + \mathsf{Adv}^{\mathsf{NIZK\text{-}ext}_2}_{\mathsf{NIZK},\mathcal{A}_{\mathsf{ZK}_2}} \\ &+ \mathsf{Adv}^{\mathsf{Sig\text{-}EUF\text{-}CMA}}_{\mathsf{Sig},\mathcal{A}_{\mathsf{Sig}}} + q_R\cdot\mathsf{Adv}^{\mathsf{sPKE\text{-}USROB}}_{\mathsf{sPKE},\mathcal{A}_{\mathsf{rob}}}.\end{aligned}$$

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:

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

As in the proof of Lemma 6.6.5, 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 Lemma 6.6.5 that there exist $\mathcal{A}_{\mathsf{PRF}}$, $\mathcal{A}_{\mathsf{ZK}_1}$, $\mathcal{A}_{\mathsf{ZK}_2}$, and $\mathcal{A}_{\mathsf{Sig}}$ such that

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

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 $m_j \leftarrow \mathsf{ACE.Dec}\big(\mathsf{ACE.Gen}(msk, j, \mathtt{rec}), c'\big)$ is for $j \in J$ replaced by

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

One can show as in the proof of Lemma 6.6.5 that there exists a probabilistic algorithm $\mathcal{A}_{\mathsf{rob}}$ such that

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

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

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

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

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

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. □

Lemma 6.6.8. *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* $(\cdot, \texttt{sen})$ *and at most* q_R *queries of the form* $(\cdot, \texttt{rec})$ *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*

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

Proof. Let $H_0, \ldots, H_4$, $V := \big[\mathsf{NIZK.Ver}\big(crs^{\mathsf{NIZK}}, x, \pi^{\mathsf{NIZK}}\big) = 1\big]$, 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 Lemma 6.6.7, and let W_{RR} be the event that $\mathcal{A}$ wins the role-respecting game:

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

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

$$\Pr^{H_0}[W_{\mathsf{RR}}] - \Pr^{H_3}[W_{\mathsf{RR}}] = \mathsf{Adv}^{\mathsf{PRF}}_{F,\mathcal{A}_{\mathsf{PRF}}} + \mathsf{Adv}^{\mathsf{NIZK\text{-}ext_1}}_{\mathsf{NIZK},\mathcal{A}_{\mathsf{ZK}_1}}, \tag{6.22}$$

and

$$\Pr^{H_3}[B_E] \leq \mathsf{Adv}^{\mathsf{NIZK\text{-}ext_2}}_{\mathsf{NIZK},\mathcal{A}_{\mathsf{ZK}_2}}. \tag{6.23}$$

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 Lemma 6.6.5 that there exists an algorithm $\mathcal{A}_{\mathsf{rob}}$ such that

$$\begin{aligned}\Pr^{H_3}[W_{\mathsf{RR}} \cap V \cap \neg B_E \cap E_G] - \Pr^{H_4}[W_{\mathsf{RR}} \cap V \cap \neg B_E \cap E_G] \\ \leq q_R \cdot \mathsf{Adv}^{\mathsf{sPKE\text{-}USROB}}_{\mathsf{sPKE},\mathcal{A}_{\mathsf{rob}}}.\end{aligned} \tag{6.24}$$

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 $\mathsf{sPKE.Gen}$ are equal, which happens only with negligible 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, 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 $\mathsf{sPKE.Gen}$ 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. $\Diamond$

Now let E_E be the event that $\mathcal{A}$ made a query $(i, \cdot)$ to $\mathcal{O}_E$ such that $\big(vk^{\mathsf{Sig}}_i, ek^{\mathsf{sPKE}}_i\big) = \big(vk^{\mathsf{Sig}}_{i_w}, ek^{\mathsf{sPKE}}_{i_w}\big)$ for $\big(vk^{\mathsf{Sig}}_i, sk^{\mathsf{Sig}}_i\big) = \mathsf{Sig.Gen}\big(1^\kappa; U([i, 1])\big)$ and $\big(ek^{\mathsf{sPKE}}_i, dk^{\mathsf{sPKE}}_i\big) = \mathsf{sPKE.Gen}\big(msk^{\mathsf{sPKE}}; U([i, 0])\big)$. 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^{Sig^*}, the adversary $\mathcal{A}_{\mathsf{Sig}_1}$ initializes the values $i_{q_0} \leftarrow \bot$, $k_q \leftarrow 1$, 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 $\big(crs^{\mathsf{NIZK}}, \xi^{\mathsf{NIZK}}\big) \leftarrow E^{\mathsf{NIZK}}_1(1^\kappa)$ 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^{ACE}_i as H_3, encrypt m using ek^{ACE}_i, and return the resulting ciphertext. If $k_q = q_0$ or $i = i_{q_0}$, then 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 $vk_i^{\mathsf{Sig}} := vk^{\mathsf{Sig}*}$. Afterwards, sample randomness r, run $\tilde{c} \leftarrow \mathsf{sPKE.Enc}\big(ek_i^{\mathsf{sPKE}}, m; r\big)$, query the signing oracle on $\tilde{c}$ to obtain a signature σ_c^{Sig}, and execute

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

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, then we have for all $\hat{c}$ that $\mathcal{O}_E$ has returned that $\mathsf{ACE.DMod}\big(sp^{\mathsf{ACE}}, \hat{c}, c\big) = 0$. 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

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

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

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

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

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

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*

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

Proof of claim. The algorithm $\mathcal{A}_{\mathsf{Sig}_2}$ on input vk^{Sig^*} executes $\left(sp^{\mathsf{sPKE}}, msk^{\mathsf{sPKE}}\right) \leftarrow \mathsf{sPKE.Setup}(1^\kappa)$ 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 $c = (\tilde{c}, \pi^{\mathsf{NIZK}})$, $\mathcal{A}_{\mathsf{Sig}_1}$ extracts a witness

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

It finally returns the forgery attempt $\left(\left[ek^{\mathsf{sPKE}}_{i_w}, vk^{\mathsf{Sig}}_{i_w}\right], \sigma^{\mathsf{Sig}}_{i_w}\right)$. 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

$$\begin{aligned}\mathsf{Adv}^{\mathsf{Sig\text{-}EUF\text{-}CMA}}_{\mathsf{Sig},\mathcal{A}_{\mathsf{Sig}}} &= \frac{q_E}{q_E+1} \cdot \mathsf{Adv}^{\mathsf{Sig\text{-}EUF\text{-}CMA}}_{\mathsf{Sig},\mathcal{A}_{\mathsf{Sig}_1}} + \frac{1}{q_E+1} \cdot \mathsf{Adv}^{\mathsf{Sig\text{-}EUF\text{-}CMA}}_{\mathsf{Sig},\mathcal{A}_{\mathsf{Sig}_2}} \\ &\ge \frac{1}{q_E+1} \cdot \Pr^{H_3}[W_{\mathsf{RR}} \cap V \cap \neg B_E \cap \neg E_G]. \qquad (6.25)\end{aligned}$$

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

$$\begin{aligned}\Pr^{H_3}[W_{\mathsf{RR}}] &= \Pr^{H_3}[W_{\mathsf{RR}} \cap B_E] + \Pr^{H_3}[W_{\mathsf{RR}} \cap V \cap \neg B_E \cap E_G] \\ &\quad + \Pr^{H_3}[W_{\mathsf{RR}} \cap V \cap \neg B_E \cap \neg E_G] \\ &\overset{(6.23),(6.24),(6.25)}{\le} \mathsf{Adv}^{\mathsf{NIZK\text{-}ext}_2}_{\mathsf{NIZK},\mathcal{A}_{\mathsf{ZK}_2}} + q_R \cdot \mathsf{Adv}^{\mathsf{sPKE\text{-}USROB}}_{\mathsf{sPKE},\mathcal{A}_{\mathsf{rob}}} \\ &\quad + \Pr^{H_4}[W_{\mathsf{RR}} \cap V \cap \neg B_E \cap E_G] \\ &\quad + (q_E+1) \cdot \mathsf{Adv}^{\mathsf{Sig\text{-}EUF\text{-}CMA}}_{\mathsf{Sig},\mathcal{A}_{\mathsf{Sig}}}.\end{aligned}$$

Combined with Claim 1 and equation (6.22), this concludes the proof. $\square$

6.6.2 Lifting Equality to Disjunction of Equalities

We finally show how an ACE scheme for equality, as the one from Section 6.6.1, 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

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

where $\mathcal{D}$ is some finite set and $\ell \in \mathbb{N}$.[6] 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 $\mathbf{x} = (x_1, \ldots, x_\ell)$ contains one key of the ACE scheme for equality for each component x_i of the role vector. To encrypt a message, it 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 of knowledge that a valid key combination was used to encrypt the same message for each component and that all signatures are 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

[6] 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.

the language $L := \{x \mid \exists w \ (x, w) \in R\}$, where 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

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

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

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

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 `sen`, generates

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

and outputs the encryption key

$$ek^{\mathsf{ACE}_{\mathsf{DEq}}}_{\mathbf{x}} := \big(vk^{\mathsf{Sig}}, ek^{\mathsf{ACE}_=}_{(x_1,1)}, \dots, ek^{\mathsf{ACE}_=}_{(x_\ell,\ell)}, vk^{\mathsf{Sig}}_{\mathbf{x}}, sk^{\mathsf{Sig}}_{\mathbf{x}}, \sigma^{\mathsf{Sig}}_{\mathbf{x}}, crs^{\mathsf{NIZK}}\big);$$

on input $msk^{\mathsf{ACE}_{\mathsf{DEq}}}$, a role $\mathbf{y} \in \mathcal{D}^\ell$, and `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 $dk^{\mathsf{ACE}_{\mathsf{DEq}}}_{\mathbf{y}} := \big(dk^{\mathsf{ACE}_=}_{(y_1,1)}, \dots, dk^{\mathsf{ACE}_=}_{(y_\ell,\ell)}\big)$.

Encryption: On input an encryption key $ek_{\mathbf{x}}^{\mathsf{ACE_{DEq}}} = \big(vk^{\mathsf{Sig}}, ek_{(x_1,1)}^{\mathsf{ACE}_=}, \dots, ek_{(x_\ell,\ell)}^{\mathsf{ACE}_=}, vk_{\mathbf{x}}^{\mathsf{Sig}}, sk_{\mathbf{x}}^{\mathsf{Sig}}, \sigma_{\mathbf{x}}^{\mathsf{Sig}}, crs^{\mathsf{NIZK}}\big)$ and a message $m \in \mathcal{M}^{\mathsf{ACE_{DEq}}}$, the algorithm $\mathsf{ACE_{DEq}.Enc}$ samples randomness $r_1, \dots, r_\ell$ and computes

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

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

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

Decryption: The algorithm $\mathsf{ACE_{DEq}.Dec}$ on input a decryption key $dk_{\mathbf{y}}^{\mathsf{ACE_{DEq}}} = \big(dk_{(y_1,1)}^{\mathsf{ACE}_=}, \dots, dk_{(y_\ell,\ell)}^{\mathsf{ACE}_=}\big)$ and a sanitized ciphertext $c' := \big(c_1', \dots, c_\ell'\big)$, computes the message $m_i \leftarrow \mathsf{ACE}_=.\mathsf{Dec}\big(dk_{(y_i,i)}^{\mathsf{ACE}_=}, c_i'\big)$ for $i \in [\ell]$. If $m_i \neq \bot$ for some $i \in [\ell]$, $\mathsf{ACE_{DEq}.Dec}$ outputs the first such m_i; otherwise it outputs $\bot$.

Modification detection: On input sanitizer parameters $sp^{\mathsf{ACE_{DEq}}} := \big(sp^{\mathsf{ACE}_=}, vk^{\mathsf{Sig}}, crs^{\mathsf{NIZK}}\big)$ 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 $\mathsf{ACE_{DEq}.DMod}$ 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 $(2,2)$. For similar reasons, we do not achieve strong sanitization security.

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.

Proposition 6.6.9. *If* $\mathsf{ACE}_=$ *is correct and detectable, then* $\mathsf{ACE}_{\mathsf{DEq}}$ *from above is correct and detectable. If* $\mathsf{ACE}_=$ *is strongly detectable, then* $\mathsf{ACE}_{\mathsf{DEq}}$ *is also strongly detectable. More precisely, for all probabilistic algorithms* $\mathcal{A}$*, there exist probabilistic algorithms* $\mathcal{A}_{\mathsf{corr}}$*,* $\mathcal{A}_{\mathsf{dtct}}$*,* $\mathcal{A}'_{\mathsf{dtct}}$*, and* $\mathcal{A}_{\mathsf{sdtct}}$ *such that*

$$\begin{aligned}
\mathsf{Adv}^{\mathsf{ACE\text{-}CORR}}_{\mathsf{ACE}_{\mathsf{DEq}},\mathcal{A}} &\le \mathsf{Adv}^{\mathsf{ACE\text{-}CORR}}_{\mathsf{ACE}_=,\mathcal{A}_{\mathsf{corr}}} + (\ell-1)\cdot \mathsf{Adv}^{\mathsf{ACE\text{-}DTCT}}_{\mathsf{ACE}_=,\mathcal{A}_{\mathsf{dtct}}},\\
\mathsf{Adv}^{\mathsf{ACE\text{-}DTCT}}_{\mathsf{ACE}_{\mathsf{DEq}},\mathcal{A}} &\le \ell\cdot \mathsf{Adv}^{\mathsf{ACE\text{-}DTCT}}_{\mathsf{ACE}_=,\mathcal{A}'_{\mathsf{dtct}}},\\
\mathsf{Adv}^{\mathsf{ACE\text{-}sDTCT}}_{\mathsf{ACE}_{\mathsf{DEq}},\mathcal{A}} &\le \ell\cdot \mathsf{Adv}^{\mathsf{ACE\text{-}sDTCT}}_{\mathsf{ACE}_=,\mathcal{A}_{\mathsf{sdtct}}}.
\end{aligned}$$

Proof. We first prove correctness. Let $\mathcal{A}$ be a probabilistic algorithm and 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 $i < i_0$.

To prove detectability, consider the adversary $\mathcal{A}'_{\mathsf{dtct}}$ that 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 proof for strong detectability is similar, while $\mathcal{A}_{\mathsf{sdtct}}$ additionally outputs the randomness used for encrypting the chosen component when the randomness output by $\mathcal{A}$ is used to generate the whole ciphertext. $\square$

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

Theorem 6.6.10. *If F is pseudorandom,* NIZK *is zero-knowledge and extractable,* Sig *is EUF-CMA secure, and* $\mathsf{ACE}_=$ *is perfectly correct, strongly detectable, has NDTCT-FENC, and is PRV-CCA, wANON-CCA, SAN-CCA, RR, and UDEC secure, then the scheme* $\mathsf{ACE}_{\mathsf{DEq}}$ *from above has NDTCT-FENC and is PRV-CCA, wANON-CCA, SAN-CCA, RR, and UDEC secure.*

We prove this theorem in a sequence of lemmata, which each prove some of 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 the PRF is only used for the signatures, which are irrelevant for privacy and anonymity.

Lemma 6.6.11. *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 $\mathcal{A}_{\mathsf{ZK}}$, $\mathcal{A}_{\mathsf{ACE}}$, $\mathcal{A}'_{\mathsf{ZK}}$, and $\mathcal{A}'_{\mathsf{ACE}}$ (which are all roughly as efficient as emulating an execution of $\mathsf{Exp}^{\mathsf{ACE\text{-}PRV\text{-}ANON\text{-}CCA}}_{\mathsf{ACE}_{\mathsf{DEq}},\mathcal{A}}$) such that*

$$\begin{aligned}\mathsf{Adv}^{\mathsf{ACE\text{-}PRV\text{-}CCA}}_{\mathsf{ACE}_{\mathsf{DEq}},\mathcal{A}} &\leq 2 \cdot \mathsf{Adv}^{\mathsf{NIZK\text{-}ZK}}_{\mathsf{NIZK},\mathcal{A}_{\mathsf{ZK}}} + \ell \cdot \mathsf{Adv}^{\mathsf{ACE\text{-}PRV\text{-}CCA}}_{\mathsf{ACE}_=,\mathcal{A}_{\mathsf{ACE}}},\\ \mathsf{Adv}^{\mathsf{ACE\text{-}wANON\text{-}CCA}}_{\mathsf{ACE}_{\mathsf{DEq}},\mathcal{A}} &\leq 2 \cdot \mathsf{Adv}^{\mathsf{NIZK\text{-}ZK}}_{\mathsf{NIZK},\mathcal{A}'_{\mathsf{ZK}}} + \ell \cdot \mathsf{Adv}^{\mathsf{ACE\text{-}wANON\text{-}CCA}}_{\mathsf{ACE}_=,\mathcal{A}'_{\mathsf{ACE}}}.\end{aligned}$$

Proof. We only prove the statement about the privacy advantage. The proof for weak anonymity is completely analogous. We can assume without loss of generality that the adversary $\mathcal{A}$ ensures that $\mathbf{x}^0 = \mathbf{x}^1$ and $P(\mathbf{x}^0, \mathbf{y}) = 0$ for all $\mathbf{y} \in J$, since doing otherwise can only decrease the privacy advantage of $\mathcal{A}$. Let $H_0 := \mathsf{Exp}^{\mathsf{ACE\text{-}PRV\text{-}ANON\text{-}CCA}}_{\mathsf{ACE}_{\mathsf{DEq}},\mathcal{A}}$

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 $\mathsf{ACE_{DEq}.Setup}$, 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 $\mathsf{ACE_{DEq}.Enc}$ 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 Lemma 6.6.3 that there exists a probabilistic algorithm $\mathcal{A}_{\mathsf{ZK}}$ such that

$$\Pr^{H_0}[b' = b] - \Pr^{H_1}[b' = b] = \mathsf{Adv}^{\mathsf{NIZK\text{-}ZK}}_{\mathsf{NIZK},\mathcal{A}_{\mathsf{ZK}}}. \tag{6.26}$$

For $k \in \{0, \dots, \ell\}$, we define $H_{2,k}$ as follows. It is identical to H_1 except that after the adversary $\mathcal{A}_1$ returns $(m_0, m_1, \mathbf{x}^0, \mathbf{x}^1, st)$, we replace the ciphertext components in c^* by $c_i \leftarrow \mathsf{ACE_=.Enc}\big(ek^{\mathsf{ACE}_=}_{(x_i^0, i)}, m_0; r_i\big)$ for all $1 \le i \le k$, and by $c_i \leftarrow \mathsf{ACE_=.Enc}\big(ek^{\mathsf{ACE}_=}_{(x_i^1, i)}, m_1; r_i\big)$ 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 $\mathsf{ACE_{DEq}.San}$ does and if the verification succeeds, it uses its oracle $\mathcal{O}_{SD}$ to sanitize and decrypt all ciphertext components. As $\mathsf{ACE_{DEq}.Dec}$, it outputs the first message different from $\perp$, or it outputs $\perp$ if the verification fails or if no such message exists. When $\mathcal{A}_1$ 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 be obtained 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 the bit b', $\mathcal{A}_{\mathsf{ACE}}$ returns the same bit $b'' \leftarrow b'$.

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 challenge ciphertext. Hence, $\mathcal{A}_{\mathsf{ACE}}$ wins if $b'' = b$ and we have

$$\begin{aligned}
\mathsf{Adv}^{\mathsf{ACE\text{-}PRV\text{-}CCA}}_{\mathsf{ACE}_=, \mathcal{A}_{\mathsf{ACE}}} &= 2 \cdot \Pr^{\mathsf{Exp}^{\mathsf{ACE\text{-}PRV\text{-}ANON\text{-}CCA}}_{\mathsf{ACE}_=, \mathcal{A}_{\mathsf{ACE}}}}[b'' = b] - 1 \\
&= \Pr^{\mathsf{Exp}^{\mathsf{ACE\text{-}PRV\text{-}ANON\text{-}CCA}}_{\mathsf{ACE}_=, \mathcal{A}_{\mathsf{ACE}}}}[b'' = 1 \mid b = 1] \\
&\qquad\qquad - \Pr^{\mathsf{Exp}^{\mathsf{ACE\text{-}PRV\text{-}ANON\text{-}CCA}}_{\mathsf{ACE}_=, \mathcal{A}_{\mathsf{ACE}}}}[b'' = 1 \mid b = 0] \\
&= \sum_{k=1}^{\ell} \frac{1}{\ell} \Pr^{H_{2,k-1}}[b' = 1] - \sum_{k=1}^{\ell} \frac{1}{\ell} \Pr^{H_{2,k}}[b' = 1].
\end{aligned}$$

Since $H_{2,0}$ corresponds to H_1 with $b = 1$ and $H_{2,\ell}$ corresponds to H_1 with $b = 0$, this yields

$$\begin{aligned}
\ell \cdot \mathsf{Adv}^{\mathsf{ACE\text{-}PRV\text{-}CCA}}_{\mathsf{ACE}_=, \mathcal{A}_{\mathsf{ACE}}} &= \Pr^{H_{2,0}}[b' = 1] - \Pr^{H_{2,\ell}}[b' = 1] \\
&= \Pr^{H_1}[b' = 1 \mid b = 1] - \Pr^{H_1}[b' = 1 \mid b = 0] \\
&= 2 \cdot \Pr^{H_1}[b' = b] - 1.
\end{aligned}$$

Combining this with equation (6.26) concludes the proof. □

Next, we sketch how to prove sanitization security.

Lemma 6.6.12. *If F is pseudorandom,* NIZK *is extractable,* Sig *is EUF-CMA secure, and* $\mathsf{ACE}_=$ *is perfectly correct, strongly detectable, and SAN-CCA secure, then the scheme* $\mathsf{ACE}_{\mathsf{DEq}}$ *from above is SAN-CCA secure.*

Proof sketch. The basic idea is to construct an adversary $\mathcal{A}_{\mathsf{SAN}}$ against the sanitization security of $\mathsf{ACE}_=$ that chooses $k_0 \twoheadleftarrow \{1, \dots \ell\}$ uniformly at random and emulates an execution of $\mathsf{Exp}^{\mathsf{ACE\text{-}SAN\text{-}CCA}}_{\mathsf{ACE}_{\mathsf{DEq}}, \mathcal{A}}$. When $\mathcal{A}_1$ returns two ciphertexts c_0, c_1, $\mathcal{A}_{\mathsf{SAN}}$ gives the sanitized ciphertext $(c'_1, \dots, c'_\ell)$ to $\mathcal{A}_2$ where $c'_i \leftarrow \mathsf{ACE}_=.\mathsf{San}(c_{0,i})$ for $1 \le i < k_0$, $c'_i \leftarrow \mathsf{ACE}_=.\mathsf{San}(c_{1,i})$ for $k_0 < i \le \ell$, and c'_{k_0} is obtained from the challenger by submitting (c_{0,k_0}, c_{1,k_0}). When $\mathcal{A}_2$ returns the bit b', $\mathcal{A}_{\mathsf{SAN}}$ returns the same bit b'. Note that $\mathcal{A}_{\mathsf{SAN}}$ wins if the bit is guessed correctly and if both returned ciphertexts sanitize properly and no decryption key has been obtained that decrypts any of the ciphertexts. If the last two conditions are not satisfied, then also $\mathcal{A}$ does not win. For the hybrid argument to go through, however, we need to ensure that $\mathcal{A}_{\mathsf{SAN}}$ still wins with probability $1/2$ when $\mathcal{A}$ violates one of these two conditions. To achieve this, $\mathcal{A}_{\mathsf{SAN}}$ needs to detect that this would next happen and in this case abort the emulation, return two valid ciphertexts (if not done already) and guess a uniform bit. To detect this event before it happens, extract witnesses from the ciphertexts returned by $\mathcal{A}_1$. If the ciphertexts are valid, the extractions are successful, the signature scheme is EUF-CMA secure, and the PRF is pseudorandom, then the ciphertexts have with overwhelming probability been obtained by encrypting messages with encryption keys that $\mathcal{A}_1$ has obtained from the oracle $\mathcal{O}_G$. Hence, $\mathcal{A}_{\mathsf{SAN}}$ knows in this case for which roles the messages have been encrypted. When $\mathcal{A}_2$ asks for a decryption key, $\mathcal{A}_{\mathsf{SAN}}$ checks whether the policy allows this key to decrypt any of the two ciphertexts. Given perfect correctness and strong detectability, the decryptions yield $\perp$ if and only if the policy does not allow decryption. Therefore, $\mathcal{A}_{\mathsf{SAN}}$ can detect when the bad event is about to happen and abort in this case. □

Non-detection of fresh encryptions directly follows from the same property of the underlying ACE scheme.

Lemma 6.6.13. *Let* $\mathsf{ACE}_{\mathsf{DEq}}$*, be the scheme from above and let* $\mathcal{A}$ *be an attacker on the non-detection of fresh encryptions. Then, there exists a probabilistic algorithm* $\mathcal{A}'$ *(which is roughly as efficient as emulating an execution of* $\mathsf{Exp}^{\mathsf{ACE\text{-}NDTCT\text{-}FENC}}_{\mathsf{ACE}_{\mathsf{DEq}},\mathcal{A}}$*) such that*

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

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

Lemma 6.6.14. *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 Lemma 6.6.8, we define hybrids $H_0 := \mathsf{Exp}^{\mathsf{ACE\text{-}URR}}_{\mathsf{ACE}_{\mathsf{DEq}},\mathcal{A}}$, 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 $w = \big(ek^{\mathsf{ACE}_=}_{(x_1,1)}, \dots, ek^{\mathsf{ACE}_=}_{(x_\ell,\ell)}, m, r_1, \dots, r_\ell, vk^{\mathsf{Sig}}_{\mathbf{x}}, \sigma^{\mathsf{Sig}}_{\mathbf{x}}, \sigma^{\mathsf{Sig}}_c\big)$ is extracted from π^{NIZK} by E_2^{NIZK} after $\mathcal{A}$ returned $c := \big(c_1, \dots, c_\ell, \pi^{\mathsf{NIZK}}\big)$. 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 Lemma 6.6.8. 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. □

6.7 ACE in Constructive Cryptography

6.7.1 The Natural Construction with ACE

We first briefly sketch the natural ideal resource one would like to construct with an ACE scheme. This resource corresponds to a repository with an interface for a trusted authority and n interfaces for the users of the system that provides the following functionalities: The trusted authority should be able to register parties for certain roles such that users having role i should be able to input a message m for role i into the repository, and users having role j should be able to access this message if and only if $P(i, j) = 1$.

The real resource from which ACE should construct this ideal resource contains a repository whose contents can be read by everyone and only the sanitizer can write to. For a user to input data into the repository, it must be encrypted and the resulting ciphertext has to be sent to the sanitizer, who then sanitizes it and inputs the result into the repository (if the sanitization is successful and the ciphertext is not detected to be modification of a previously received ciphertext). To allow this, the real resource also needs to contain (anonymous) secure channels from the users to the sanitizer. To access data, users simply read the sanitized ciphertexts from the repository and decrypt them with their decryption keys. To distribute keys, the real resource also needs to contain secure channels from the central authority to the users, and an authenticated channel from the central authority to the sanitizer.

6.7.2 Issues Preventing the Construction

If we allow the registrations of roles to be dynamic, the same commitment problem arises that we encountered for identity-based encryption in Section 4.4.1 and for functional encryption in Section 5.5.1.

But even for static registrations at the beginning or when allowing random oracles, there is a more fundamental problem that prevents us from constructing such a repository. Consider for example users A_1, A_2, and B such that A_1 has role i_1 that allows to send information to B, but A_2 has no role that allows this. In an ideal world, it should therefore be possible at the interface for A_1 to send messages to B, but not at the interface for A_2. In the real world, however, everyone who knows the encryption key for the role i_1 can produce ciphertexts that pass sanitization and can subsequently be decrypted by B. If this is modeled in constructive cryptography and A_1 is dishonest, this means that the distinguisher knows such a key and can produce such ciphertexts. If A_2 is also dishonest, then the distinguisher can input these ciphertexts at the interface of A_2, and B will receive it. Hence, the ideal resource must allow all dishonest users to send messages to B if at least one of them is allowed to do so. If the receiver B is also dishonest, B can decrypt if there exists a dishonest user who is allowed to decrypt. In general, a dishonest party A can send messages to another dishonest party B if (and only if) there exist dishonest parties A' and B' such that A' has some role i_0 and B' has some role j_0 with $P(i_0, j_0) = 1$.

At first, this might seem to be reasonable and to simply capture potential collusions among dishonest parties. Indeed, if such a system is implemented in practice, dishonest users can exchange their keys outside the system and use them to encrypt and decrypt as in the examples above. For many policies, however, the remaining guarantees are extremely weak. For example, assume that $P(i, i) = 1$ for all roles i, i.e., users are allowed to send to themselves and to other users with the same role. This is a quite natural assumption and for example the case for the Bell-LaPadula policy $P(i, j) = 1 \Leftrightarrow i \leq j$. In this setting, two dishonest users A and B can communicate if there exists *some* dishonest user who has *some* role. That is because this user's encryption key can be used by A to encrypt and the corresponding decryption key can be used by B to decrypt. Hence, dishonest users can essentially always communicate, which completely undermines the goal of access control encryption.

6.7.3 Conclusions

One possible conclusion is that ACE simply only provides very weak guarantees. This can be justified by the fact that dishonest parties can indeed share their keys and use them as described above to communicate.

Another conclusion is that constructive cryptography is too pessimistic. While it is true that users *can* exchange their keys, they actively have to do so in practice. There are many reasons why they might not do that, e.g., the keys could be very large or somehow protected by hardware, users might fear punishment if it is discovered that they have shared their keys, or they might simply not want to do so. Hence, a more useful and realistic guarantee one can expect from the construction described above is that two users A and B can communicate via the system if they have roles that allow them to do so, or if they have obtained keys from other users that are allowed to communicate. This means that the ideal resource needs to be quantified over the collusion of dishonest parties.

In constructive cryptography, it is not possible to quantify over the information dishonest parties have exchanged since the distinguisher obtains all information from all dishonest parties. In this sense, the framework captures the worst-case scenario in which all dishonest parties collude with each other. To formalize the intuition above, it is therefore necessary to develop a variant of constructive cryptography that allows such a quantification. This is, however, beyond the scope of this thesis and left for future research.

Chapter 7

Conclusion

We have analyzed several types of encryption schemes in the constructive cryptography framework and have taken a critical look at classical security definitions for these schemes. This has led to a better understanding of how those schemes can be used and which security notions are needed for natural applications of them. Due to our analysis, we could also identify issues with existing definitions and introduce new ones to fix these issues. Below, we discuss our results and open problems for the individual primitives in detail.

One-time pad. We have analyzed encryption with a dishonest receiver and have shown that it is impossible in this setting to construct a fully secure channel from a shared secret key and an authenticated channel. We have further shown that the one-time pad, in contrast to ordinary encryption schemes, can be used to construct a variant of a secure channel that allows a dishonest receiver to leak the message only before receiving it, not afterwards.

An open problem is the analogous treatment of the setting with a potentially dishonest sender for the one-time pad as well as for other types of encryption schemes. Furthermore, it would be interesting to analyze which constructions one can achieve with different types of deniable encryption schemes, which provide similar features as the one of the one-time pad we used in our construction.

Identity-based encryption. The standard application of identity-based encryption is non-interactive secure communication. We have modeled this in the constructive cryptography framework and analyzed the security against passive attackers. Our analysis revealed that the construction of an ideal resource that allows to adaptively register new users after sending messages is impossible in the standard model. We have further shown that this ideal resource can be constructed in the random oracle model and that weaker ideal resources can be constructed in the standard model. To achieve the latter construction, the IBE scheme has to be IND-ID1-CPA secure. This is a new security notion we introduce and which is weaker than IND-ID-CPA security, the standard notion for IBE schemes. Similarly, we have introduced IND-sID1-CPA security as a variant of selective security and shown for which construction schemes satisfying it can be used.

An open problem is to find IND-ID1-CPA or IND-sID1-CPA secure IBE schemes that are more efficient than existing ones. While our new notions are formally weaker than existing ones (but still achieve meaningful constructions), it is unclear how to exploit their weakness to obtain more efficient schemes. A further open problem is to transfer our analysis to a setting with active attackers and understand the precise connection to chosen-ciphertext attacks.

Functional encryption. We have formalized the security of functional encryption schemes via the construction of a repository with fine-grained access control. Similarly to our results for identity-based encryption, the construction of a repository that allows to adaptively grant access rights is impossible in the standard model and possible in the random oracle model. We then have compared existing security definitions to the construction of different variants of repositories. Due to our analysis, we were able to clarify a misconception about the adequacy of the security definition by Boneh, Sahai, and Waters [BSW11], and to improve the overall understanding of simulation-based security for functional encryption.

We have sketched how our results can be generalized to variants of functional encryption for randomized functions and functions of more than one variable. A possible direction for future research is to precisely formalize these more general repositories and to investigate whether existing security definitions are sufficient for their construction.

Access control encryption. In contrast to the other types of encryption discussed above, we have not formally carried out a constructive treatment of access control encryption. By considering how such schemes would be used, we could nevertheless identify issues of existing security notions, most importantly a lack of security against chosen-ciphertext attacks. We have introduced new security definitions to fix these issues and presented new schemes satisfying our strong notions.

Analyzing our new notions in the constructive cryptography framework is left as an open problem. As discussed in Section 6.7.3, meaningful results in this direction require an extension of the framework to allow the quantification of the ideal resource over the collusions of dishonest parties. A further open problem is to find more efficient schemes and schemes that support more general security policies for our new notions.

Bibliography

[ABN10] M. Abdalla, M. Bellare, and G. Neven, "Robust encryption," in *Theory of Cryptography Conference (TCC) 2010*, Springer Berlin Heidelberg, 2010, pp. 480–497. DOI: 10.1007/978-3-642-11799-2_28.

[AGVW13] S. Agrawal, S. Gorbunov, V. Vaikuntanathan, and H. Wee, "Functional encryption: New perspectives and lower bounds," in *Advances in Cryptology—CRYPTO 2013*, Springer Berlin Heidelberg, 2013, pp. 500–518. DOI: 10.1007/978-3-642-40084-1_28.

[ABF+13] J. Alwen, M. Barbosa, P. Farshim, R. Gennaro, S. D. Gordon, S. Tessaro, and D. A. Wilson, "On the relationship between functional encryption, obfuscation, and fully homomorphic encryption," in *IMA International Conference on Cryptography and Coding (IMACC) 2013*, Springer Berlin Heidelberg, 2013, pp. 65–84. DOI: 10.1007/978-3-642-45239-0_5.

[AOZZ15] J. Alwen, R. Ostrovsky, H.-S. Zhou, and V. Zikas, "Incoercible multi-party computation and universally composable receipt-free voting," in *Advances in Cryptology—CRYPTO 2015*, Springer Berlin Heidelberg, 2015, pp. 763–780. DOI: 10.1007/978-3-662-48000-7_37.

[BPW07] M. Backes, B. Pfitzmann, and M. Waidner, "The reactive simulatability (RSIM) framework for asynchronous systems," *Information and Computation*, vol. 205, no. 12, pp. 1685–1720, 2007. DOI: 10.1016/j.ic.2007.05.002.

[BMM17] C. Badertscher, C. Matt, and U. Maurer, "Strengthening access control encryption," in *Advances in Cryptology—ASIACRYPT 2017*, to appear, Springer Berlin Heidelberg, 2017.

[BMM+15a] C. Badertscher, C. Matt, U. Maurer, P. Rogaway, and B. Tackmann, "Augmented secure channels and the goal of the TLS 1.3 record layer," in *International Conference on Provable Security (ProvSec) 2015*, Springer International Publishing, 2015, pp. 85–104. DOI: `10.1007/978-3-319-26059-4_5`.

[BMM+15b] ——, "Robust authenticated encryption and the limits of symmetric cryptography," in *IMA International Conference on Cryptography and Coding (IMACC) 2015*, Springer International Publishing, 2015, pp. 112–129. DOI: `10.1007/978-3-319-27239-9_7`.

[BF13] M. Barbosa and P. Farshim, "On the semantic security of functional encryption schemes," in *IACR International Conference on Practice and Theory in Public-Key Cryptography (PKC) 2013*, Springer Berlin Heidelberg, 2013, pp. 143–161. DOI: `10.1007/978-3-642-36362-7_10`.

[Bea92] D. Beaver, "Foundations of secure interactive computing," in *Advances in Cryptology—CRYPTO 1991*, Springer Berlin Heidelberg, 1992, pp. 377–391. DOI: `10.1007/3-540-46766-1_31`.

[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*, Springer Berlin Heidelberg, 2001, pp. 566–582. DOI: `10.1007/3-540-45682-1_33`.

[BDPR98] M. Bellare, A. Desai, D. Pointcheval, and P. Rogaway, "Relations among notions of security for public-key encryption schemes," in *Advances in Cryptology—CRYPTO*

1998, Springer Berlin Heidelberg, 1998, pp. 26–45. DOI: 10.1007/BFb0055718.

[BO13] M. Bellare and A. O'Neill, "Semantically-secure functional encryption: Possibility results, impossibility results and the quest for a general definition," in *International Conference on Cryptology and Network Security (CANS) 2013*, Springer International Publishing, 2013, pp. 218–234. DOI: 10.1007/978-3-319-02937-5_12.

[BR93] M. Bellare and P. Rogaway, "Random oracles are practical: A paradigm for designing efficient protocols," in *ACM Conference on Computer and Communications Security (CCS) 1993*, ACM, 1993, pp. 62–73. DOI: 10.1145/168588.168596.

[BB04] D. Boneh and X. Boyen, "Efficient selective-ID secure identity-based encryption without random oracles," in *Advances in Cryptology—EUROCRYPT 2004*, Springer Berlin Heidelberg, 2004, pp. 223–238. DOI: 10.1007/978-3-540-24676-3_14.

[BCHK07] D. Boneh, R. Canetti, S. Halevi, and J. Katz, "Chosen-ciphertext security from identity-based encryption," *SIAM Journal on Computing*, vol. 36, no. 5, pp. 1301–1328, 2007. DOI: 10.1137/S009753970544713X.

[BF01] D. Boneh and M. Franklin, "Identity-based encryption from the Weil pairing," in *Advances in Cryptology—CRYPTO 2001*, Springer Berlin Heidelberg, 2001, pp. 213–229. DOI: 10.1007/3-540-44647-8_13.

[BSW11] D. Boneh, A. Sahai, and B. Waters, "Functional encryption: Definitions and challenges," in *Theory of Cryptography Conference (TCC) 2011*, Springer Berlin Heidelberg, 2011, pp. 253–273. DOI: 10.1007/978-3-642-19571-6_16.

[BSW12] ——, "Functional encryption: A new vision for public-key cryptography," *Commun. ACM*, vol. 55, no. 11, pp. 56–64, Nov. 2012. DOI: 10.1145/2366316.2366333.

[Can01] R. Canetti, "Universally composable security: A new paradigm for cryptographic protocols," in *IEEE Symposium on Foundations of Computer Science (FOCS) 2001*, Oct. 2001, pp. 136–145. DOI: 10.1109/SFCS.2001.959888.

[CDNO97] R. Canetti, C. Dwork, M. Naor, and R. Ostrovsky, "Deniable encryption," in *Advances in Cryptology—CRYPTO 1997*, Springer Berlin Heidelberg, 1997, pp. 90–104. DOI: 10.1007/BFb0052229.

[CFGN96] R. Canetti, U. Feige, O. Goldreich, and M. Naor, "Adaptively secure multi-party computation," in *ACM Symposium on Theory of Computing (STOC) 1996*, ACM, 1996, pp. 639–648. DOI: 10.1145/237814.238015.

[CHK03] R. Canetti, S. Halevi, and J. Katz, "A forward-secure public-key encryption scheme," in *Advances in Cryptology—EUROCRYPT 2003*, Springer Berlin Heidelberg, 2003, pp. 255–271. DOI: 10.1007/3-540-39200-9_16.

[CKN03] R. Canetti, H. Krawczyk, and J. B. Nielsen, "Relaxing chosen-ciphertext security," in *Advances in Cryptology—CRYPTO 2003*, Springer Berlin Heidelberg, 2003, pp. 565–582. DOI: 10.1007/978-3-540-45146-4_33.

[CV12] R. Canetti and M. Vald, "Universally composable security with local adversaries," in *International Conference on Security and Cryptography for Networks (SCN) 2012*, Springer Berlin Heidelberg, 2012, pp. 281–301. DOI: 10.1007/978-3-642-32928-9_16.

[Coc01] C. Cocks, "An identity based encryption scheme based on quadratic residues," in *IMA International Conference on Cryptography and Coding (IMACC) 2001*, Springer Berlin Heidelberg, 2001, pp. 360–363. DOI: 10.1007/3-540-45325-3_32.

[CMT13] S. Coretti, U. Maurer, and B. Tackmann, "Constructing confidential channels from authenticated channels—public-key encryption revisited," in *Advances in Cryptology—ASIACRYPT 2013*, Springer Berlin Heidelberg, 2013, pp. 134–153. DOI: 10.1007/978-3-642-42033-7_8.

[CS98] R. Cramer and V. Shoup, "A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack," in *Advances in Cryptology—CRYPTO 1998*, Springer Berlin Heidelberg, 1998, pp. 13–25. DOI: 10.1007/BFb0055717.

[DHO16] I. Damgård, H. Haagh, and C. Orlandi, "Access control encryption: Enforcing information flow with cryptography," in *Theory of Cryptography Conference (TCC) 2016-B*, Springer Berlin Heidelberg, 2016, pp. 547–576. DOI: 10.1007/978-3-662-53644-5_21.

[DIJ+13] A. De Caro, V. Iovino, A. Jain, A. O'Neill, O. Paneth, and G. Persiano, "On the achievability of simulation-based security for functional encryption," in *Advances in Cryptology—CRYPTO 2013*, Springer Berlin Heidelberg, 2013, pp. 519–535. DOI: 10.1007/978-3-642-40084-1_29.

[DDN00] D. Dolev, C. Dwork, and M. Naor, "Nonmalleable cryptography," *SIAM Journal on Computing*, vol. 30, no. 2, pp. 391–437, 2000. DOI: 10.1137/S0097539795291562.

[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 *IACR International Conference on Practice and Theory in Public-Key Cryptography (PKC) 2013*, 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 *IACR International Conference on Practice and Theory in Public-Key Cryptography (PKC) 2017*, Springer Berlin Heidelberg, 2017, pp. 88–118. DOI: 10.1007/978-3-662-54388-7_4.

[GGH+13] S. Garg, C. Gentry, S. Halevi, M. Raykova, A. Sahai, and B. Waters, "Candidate indistinguishability obfuscation and functional encryption for all circuits," in *IEEE Symposium on Foundations of Computer Science (FOCS) 2013*, Oct. 2013, pp. 40–49. DOI: 10.1109/FOCS.2013.13.

[GPV08] C. Gentry, C. Peikert, and V. Vaikuntanathan, "Trapdoors for hard lattices and new cryptographic constructions," in *ACM Symposium on Theory of Computing (STOC) 2008*, ACM, 2008, pp. 197–206. DOI: 10.1145/1374376.1374407.

[GMW87] O. Goldreich, S. Micali, and A. Wigderson, "How to play any mental game," in *ACM Symposium on Theory of Computing (STOC) 1987*, ACM, 1987, pp. 218–229. DOI: 10.1145/28395.28420.

[GGG+14] S. Goldwasser, S. D. Gordon, V. Goyal, A. Jain, J. Katz, F.-H. Liu, A. Sahai, E. Shi, and H.-S. Zhou, "Multi-input functional encryption," in *Advances in Cryptology—EUROCRYPT 2014*, Springer Berlin Heidelberg, 2014, pp. 578–602. DOI: 10.1007/978-3-642-55220-5_32.

[GGJS13] S. Goldwasser, V. Goyal, A. Jain, and A. Sahai, *Multi-input functional encryption*, Cryptology ePrint Archive, Report 2013/727, http://eprint.iacr.org/2013/727, 2013.

[GKP+13] S. Goldwasser, Y. Kalai, R. A. Popa, V. Vaikuntanathan, and N. Zeldovich, "Reusable garbled circuits and succinct functional encryption," in *ACM Symposium on Theory of Computing (STOC) 2013*, ACM, 2013, pp. 555–564. DOI: 10.1145/2488608.2488678.

[GM84] S. Goldwasser and S. Micali, "Probabilistic encryption," *Journal of Computer and System Sciences*, vol. 28, no. 2, pp. 270–299, 1984. DOI: 10.1016/0022-0000(84)90070-9.

[GJJS04] P. Golle, M. Jakobsson, A. Juels, and P. Syverson, "Universal re-encryption for mixnets," in *Topics in Cryptology—CT-RSA 2004*, Springer Berlin Heidelberg, 2004, pp. 163–178. DOI: 10.1007/978-3-540-24660-2_14.

[GVW12] S. Gorbunov, V. Vaikuntanathan, and H. Wee, “Functional encryption with bounded collusions via multi-party computation,” in *Advances in Cryptology—CRYPTO 2012*, Springer Berlin Heidelberg, 2012, pp. 162–179. DOI: `10.1007/978-3-642-32009-5_11`.

[GKL+13] S. D. Gordon, J. Katz, F.-H. Liu, E. Shi, and H.-S. Zhou, *Multi-input functional encryption*, Cryptology ePrint Archive, Report 2013/774, `http://eprint.iacr.org/2013/774`, 2013.

[GJKS15] V. Goyal, A. Jain, V. Koppula, and A. Sahai, “Functional encryption for randomized functionalities,” in *Theory of Cryptography Conference (TCC) 2015*, Springer Berlin Heidelberg, 2015, pp. 325–351. DOI: `10.1007/978-3-662-46497-7_13`.

[GPSW06] V. Goyal, O. Pandey, A. Sahai, and B. Waters, “Attribute-based encryption for fine-grained access control of encrypted data,” in *ACM Conference on Computer and Communications Security (CCS) 2006*, 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 Conference (TCC) 2004*, 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*, Springer Berlin Heidelberg, 2006, pp. 444–459. DOI: `10.1007/11935230_29`.

[HMM15] D. Hofheinz, C. Matt, and U. Maurer, “Idealizing identity-based encryption,” in *Advances in Cryptology—ASIACRYPT 2015*, Springer Berlin Heidelberg, 2015, pp. 495–520. DOI: `10.1007/978-3-662-48797-6_21`.

[HS15] D. Hofheinz and V. Shoup, “GNUC: A new universal composability framework,” *Journal of Cryptology*, vol. 28, no. 3, pp. 423–508, 2015. DOI: `10.1007/s00145-013-9160-y`.

[KW17] S. Kim and D. J. Wu, "Access control encryption for general policies from standard assumptions," in *Advances in Cryptology—ASIACRYPT 2017*, to appear, Springer Berlin Heidelberg, 2017.

[KMO+13] M. Kohlweiss, U. Maurer, C. Onete, B. Tackmann, and D. Venturi, "Anonymity-preserving public-key encryption: A constructive approach," in *International Symposium on Privacy Enhancing Technologies (PETS) 2013*, Springer Berlin Heidelberg, 2013, pp. 19–39. DOI: `10.1007/978-3-642-39077-7_2`.

[KT09] R. Küsters and M. Tuengerthal, "Universally composable symmetric encryption," in *IEEE Computer Security Foundations Symposium (CSF) 2009*, Jul. 2009, pp. 293–307. DOI: `10.1109/CSF.2009.18`.

[KT13] ——, *The IITM model: A simple and expressive model for universal composability*, Cryptology ePrint Archive, Report 2013/025, `http://eprint.iacr.org/2013/025`, 2013.

[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`.

[MM13] C. Matt and U. Maurer, "The one-time pad revisited," in *IEEE International Symposium on Information Theory (ISIT) 2013*, Jul. 2013, pp. 2706–2710. DOI: `10.1109/ISIT.2013.6620718`.

[MM15] ——, "A definitional framework for functional encryption," in *IEEE Computer Security Foundations Symposium (CSF) 2015*, Jul. 2015, pp. 217–231. DOI: `10.1109/CSF.2015.22`.

[Mau12] U. Maurer, "Constructive cryptography – a new paradigm for security definitions and proofs," in *Theory of Security and Applications: Joint Workshop, TOSCA 2011*, Springer Berlin Heidelberg, 2012, pp. 33–56. DOI: `10.1007/978-3-642-27375-9_3`.

[MY91] U. M. Maurer and Y. Yacobi, "Non-interactive public-key cryptography," in *Advances in Cryptology—EUROCRYPT 1991*, Springer Berlin Heidelberg, 1991, pp. 498–507. DOI: `10.1007/3-540-46416-6_43`.

[MR11] U. Maurer and R. Renner, "Abstract cryptography," in *Symposium on Innovations in Computer Science (ICS) 2011*, Tsinghua University Press, Jan. 2011, pp. 1–21.

[MRT12] U. Maurer, A. Rüedlinger, and B. Tackmann, "Confidentiality and integrity: A constructive perspective," in *Theory of Cryptography Conference (TCC) 2012*, Springer Berlin Heidelberg, 2012, pp. 209–229. DOI: `10.1007/978-3-642-28914-9_12`.

[MR92] S. Micali and P. Rogaway, "Secure computation," in *Advances in Cryptology—CRYPTO 1991*, Springer Berlin Heidelberg, 1992, pp. 392–404. DOI: `10.1007/3-540-46766-1_32`.

[NY90] M. Naor and M. Yung, "Public-key cryptosystems provably secure against chosen ciphertext attacks," in *ACM Symposium on Theory of Computing (STOC) 1990*, ACM, 1990, pp. 427–437. DOI: `10.1145/100216.100273`.

[Nie02] J. B. Nielsen, "Separating random oracle proofs from complexity theoretic proofs: The non-committing encryption case," in *Advances in Cryptology—CRYPTO 2002*, Springer Berlin Heidelberg, 2002, pp. 111–126. DOI: `10.1007/3-540-45708-9_8`.

[NMO06] R. Nishimaki, Y. Manabe, and T. Okamoto, "Universally composable identity-based encryption," in *Progress in Cryptology—VIETCRYPT 2006*, Springer Berlin Heidelberg, 2006, pp. 337–353. DOI: `10.1007/11958239_23`.

[ONe10] A. O'Neill, *Definitional issues in functional encryption*, Cryptology ePrint Archive, Report 2010/556, `http://eprint.iacr.org/2010/556`, 2010.

[PW01] B. Pfitzmann and M. Waidner, “A model for asynchronous reactive systems and its application to secure message transmission,” in *IEEE Symposium on Security and Privacy (S&P) 2001*, 2001, pp. 184–200. DOI: `10.1109/SECPRI.2001.924298`.

[PR07] M. Prabhakaran and M. Rosulek, “Rerandomizable RCCA encryption,” in *Advances in Cryptology—CRYPTO 2007*, Springer Berlin Heidelberg, 2007, pp. 517–534. DOI: `10.1007/978-3-540-74143-5_29`.

[RS92] C. Rackoff and D. Simon, “Non-interactive zero-knowledge proof of knowledge and chosen ciphertext attack,” in *Advances in Cryptology—CRYPTO 1991*, Springer Berlin Heidelberg, 1992, pp. 433–444. DOI: `10.1007/3-540-46766-1_35`.

[RSM05] D. Raub, R. Steinwandt, and J. Müller-Quade, “On the security and composability of the one time pad,” in *Conference on Current Trends in Theory and Practice of Computer Science (SOFSEM) 2005*, Springer Berlin Heidelberg, 2005, pp. 288–297. DOI: `10.1007/978-3-540-30577-4_32`.

[SPPM13] R. Sadikin, Y. Park, K. Park, and S. Moon, “Universal composability notion for functional encryption schemes,” *Journal of the Korea Industrial Information Systems Research*, vol. 18, no. 3, pp. 17–26, 2013. DOI: `10.9723/jksiis.2013.18.3.017`.

[Sah99] A. Sahai, “Non-malleable non-interactive zero knowledge and adaptive chosen-ciphertext security,” in *IEEE Symposium on Foundations of Computer Science (FOCS) 1999*, 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*, Springer Berlin Heidelberg, 2005, pp. 457–473. DOI: `10.1007/11426639_27`.

[Sha85] A. Shamir, “Identity-based cryptosystems and signature schemes,” in *Advances in Cryptology—CRYPTO 1984*, Springer Berlin Heidelberg, 1985, pp. 47–53. DOI: `10.1007/3-540-39568-7_5`.

[Sha49] C. E. Shannon, "Communication theory of secrecy systems," *The Bell System Technical Journal*, vol. 28, no. 4, pp. 656–715, Oct. 1949. DOI: 10.1002/j.1538-7305.1949.tb00928.x.

[TZMT17] G. Tan, R. Zhang, H. Ma, and Y. Tao, "Access control encryption based on LWE," in *ACM International Workshop on ASIA Public-Key Cryptography (APKC) 2017*, ACM, 2017, pp. 43–50. DOI: 10.1145/3055504.3055509.

[UM10] D. Unruh and J. Müller-Quade, "Universally composable incoercibility," in *Advances in Cryptology—CRYPTO 2010*, Springer Berlin Heidelberg, 2010, pp. 411–428. DOI: 10.1007/978-3-642-14623-7_22.

[Wat05] B. Waters, "Efficient identity-based encryption without random oracles," in *Advances in Cryptology—EUROCRYPT 2005*, Springer Berlin Heidelberg, 2005, pp. 114–127. DOI: 10.1007/11426639_7.

Curriculum Vitae

Christian Matt
Citizen of the Federal Republic of Germany.
Born on May 12, 1986, in Speyer, Germany.

High School Education

09/1997–03/2006	*Gymnasium,* Hannah-Arendt-Gymnasium, Haßloch, Germany.

Internships

04/2006–09/2006	*Intern, software development,* SAP AG, St. Leon-Rot, Germany.

University Studies

10/2006–10/2011	*Diplom in computer science,* Karlsruhe Institute of Technology, Germany.
10/2006–12/2011	*Diplom in mathematics,* Karlsruhe Institute of Technology, Germany.
10/2009–06/2010	*Exchange student,* Lancaster University, United Kingdom.
03/2012–present	*Dr. sc. ETH Zurich in computer science,* ETH Zurich, Switzerland.