Sender-binding Encryption

Towards Finding the Weakest Encryption Security to Realise Secure Communication

Zur Erlangung des akademischen Grades einer

Doktorin der Naturwissenschaften

von der KIT-Fakultät für Informatik
des Karlsruher Instituts für Technologie (KIT)

genehmigte

Dissertation

von

Rebecca Schwerdt

Tag der mündlichen Prüfung:	18. Juli 2024
Erster Referent:	Prof. Dr. Jörn Müller-Quade
Zweite:r Referent:in:	Prof. Dr. Dominique Unruh

Contents

Abstract

The oldest and arguably most important cryptographic application is secure communication. For modern day digital communication this task is commonly split into encryption and authentication of messages. Encryption which is strong enough for this purpose if authentication is already provided has long been known. However, it is also vital to know which type of encryption security is not only sufficient but necessary. Using encryption schemes which are unnecessarily strong does not only waste computational resources but also unnecessarily precludes communication being conducted in a secure manner when resources are too scarce.

For public-key encryption the problem was raised more than twenty years ago, but still remains open to this day. Furthermore, many applications nowadays use hybrid encryption rather than plain public-key encryption, which combines the benefits of public-key key encapsulation with those of symmetrically encrypted messages. In this hybrid encryption setting, however, minimal encryption requirements for secure communication have not received much attention at all, so far.

In this dissertation the question which level of encryption security is absolutely necessary to facilitate secure communication from given authenticated channels is considered specifically in the universal composability (UC) framework. In the UC framework secure communication as well as authenticated channels are modelled as ideal functionalities. Security is asserted via the real/ideal paradigm in such a way that it is guaranteed even under arbitrary and concurrent composition of protocols. All proofs in this work are conducted for static corruption and for a polynomial set of participating parties.

The dissertation introduces the new public-key concept of sender-binding encryption and develops a corresponding security notion termed IND-SB-CPA—which stands for indistinguishability under sender-binding chosen plaintext attack. IND-SB-CPA security is specifically tailored to be as weak as possible but still sufficient in conjunction with authentication to achieve secure communication. Furthermore, a related concept of sender-binding key encapsulation for hybrid encryption as well as the respective version of IND-SB-CPA security is provided. Both new concepts and security notions are thoroughly investigated regarding generic transformations from other types of encryption and key encapsulation schemes as well as their theoretic classification with respect to established game-based security notions.

In the setting of public-key encryption it is not only shown that IND-SB-CPA secure sender-binding encryption unifies all prior approaches and is in fact strong enough to UC-realise secure message transfer from authenticated channels. For the first time in this line of research, an optimality result is provided as well: Restricted to the de facto standard encrypt-then-authenticate principle to combine encryption and authentication, IND-SB-CPA secure sender-binding encryption is shown to be the weakest possible public-key encryption for the purpose of secure communication.

For the setting of hybrid encryption, IND-SB-CPA secure sender-binding key encapsulation is proven strong enough to realise secure communication both for single message communication—where a new symmetric key is used for every message—as well as session communication—where one symmetric key is persistently used throughout a whole session—if key encapsulation is paired with a suitably secure symmetric encryption.

Overall, restricted to static corruption, a polynomial set of participating parties and the encrypt-then-authenticate principle this dissertation is able to fully answer the above research question for the public-key encryption setting, while considerable headway is made for hybrid encryption as well.

Zusammenfassung

Die älteste und vermutlich wichtigste Anwendung von Kryptographie ist sichere Kommunikation. Für heutige digitale Kommunikation wird diese Aufgabe üblicherweise in die Aspekte Verschlüsselung und Authentifikation von Nachrichten unterteilt. Verschlüsselungsverfahren, die stark genug für diesen Zweck sind solange Authentifizierung bereits gewährleistet wird, sind schon lange Zeit bekannt. Allerdings ist es ebenso relevant zu wissen welches Sicherheitslevel für Verschlüsselungen nicht nur hinreichend, sondern auch notwendig ist. Werden Verschlüsselungsverfahren eingesetzt, die unnötig starke Sicherheit bieten, so wird nicht nur Rechenkapazität verschwendet. Es wird auch unnötigerweise verhindert, dass sichere Kommunikation durchgeführt werden kann, wenn die zur Verfügung stehenden Ressourcen zu begrenzt für solch aufwändige Verfahren sind.

Für asymmetrische Verschlüsselung wurde diese Frage bereits vor zwanzig Jahren aufgeworfen, ist aber dennoch bis heute unbeantwortet. Hinzu kommt, dass viele Anwendungen heutzutage hybride Verschlüsselung statt einfacher asymmetrischer Verschlüsselung einsetzen, um die Vorteile von asymmetrischer Key Encapsulation mit denen von symmetrisch verschlüsselten Nachrichten zu kombinieren. Im Kontext von hybrider Verschlüsselung hat das Thema der Minimalanforderungen für sichere Kommunikation bisher allerdings kaum Aufmerksamkeit erfahren.

In dieser Dissertation wird die Frage, welches Sicherheitslevel für Verschlüsselung tatsächlich notwendig ist um sichere Kommunikation auf authentifizierten Kanälen zu erreichen, konkret im Universal Composability (UC) Modell betrachtet. Im UC-Modell werden sichere Kommunikation sowie authentifizierte Kanäle als ideale Funktionalitäten modelliert. Sicherheit wird so über das real/ideal-Paradigma definiert, dass sie auch bei beliebiger und nebenläufiger Komposition von Protokollen gewährleistet bleibt. Alle Beweise in dieser Arbeit werden für statische Korruption und eine polynomiell beschränkte Menge beteiligter Parteien geführt.

Die Dissertation führt das neue Konzept von Sender-gebundener Verschlüsselung ein und entwickelt einen zugehörigen Sicherheitsbegriff mit Namen IND-SB-CPA, was als Abkürzung für „indistinguishability under sender-binding chosen plaintext attack“ steht. IND-SB-CPA Sicherheit ist speziell dafür entwickelt schwächst möglich und gleichzeitig ausreichend stark zu sein, um sichere Kommunikation mittels authentifizierter Kanäle zu erreichen. Zusätzlich wird das verwandte Konzept von Sender-gebundener Key Encapsulation für hybride Verschlüsselung inklusive einer zugehörigen Version von IND-SB-CPA Sicherheit vorgestellt. Beide neuen Konzepte und Sicherheitsbegriffe werden umfassend in Bezug auf generische Transformationen ausgehend von anderen Formen der Verschlüsselung und Key Encapsulation untersucht, sowie theoretisch in das Feld bereits etablierter spielebasierter Sicherheitsbegriffe eingeordnet.

Im Kontext der asymmetrischen Verschlüsselung wird nicht nur gezeigt, dass IND-SB-CPA sichere Sender-gebundene Verschlüsselung alle bisherigen Ansätze abdeckt und tatsächlich ausreichend Sicherheit bietet, um sichere Nachrichtenübertragung mittels authentifizierter Kanäle im UC-Modell zu realisieren. Zum ersten Mal auf diesem konkreten Forschungsgebiet wird außerdem eine Optimalitätsaussage getroffen: Eingeschränkt auf den faktischen Standard des encrypt-then-authenticate Prinzips zur Verbindung von Verschlüsselung und Authentifikation wird bewiesen, dass IND-SB-CPA sichere Sender-gebundene Verschlüsselung tatsächlich der schwächst mögliche asymmetrische Verschlüsselungsbegriff zur UC-Realisierung von sicherer Kommunikation ist.

Im Kontext hybrider Verschlüsselung wird gezeigt, dass IND-SB-CPA sichere Sender-gebundene Key Encapsulation stark genug ist, um sichere Kommunikation sowohl im single-message Modus zu erreichen, wo für jede Nachricht ein neuer symmetrischer Schlüssel verwendet wird, als auch im Session Modus, wo ein symmetrischer Schlüssel langlebig für eine ganze Kommunikationssitzung Bestand hat – solange die Key Encapsulation mit ausreichend sicherer symmetrischer Verschlüsselung kombiniert wird.

Insgesamt gelingt es in dieser Dissertation unter Beschränkung auf statische Korruption, eine polynomielle Menge beteiligter Parteien und das encrypt-then-authenticate Prinzip die obige Forschungsfrage im Kontext von asymmetrischer Verschlüsselung vollständig zu beantworten, während auch für hybride Verschlüsselung erhebliche Fortschritte erzielt werden können.

Acknowledgements

First and foremost, I would like to thank my supervisor Professor Jörn Müller-Quade for all his support, immense freedom to pursue my own ideas and research interests, as well as for all the trust that went with this freedom. I would furthermore like to thank him for creating an exceedingly flexible work environment, enabling me to put all my energy into my work as well as into raising a family without making it a difficult balancing act.

Additional thanks should go to Professor Dominique Unruh for co-refereeing my thesis and providing valuable feedback.

This endeavour would also not have been possible without many of my colleagues and friends. I am grateful to Bernhard Löwe for spiking my interest in cryptography, starting me on the path of this PhD and helping me pursue it. To Matthias Nagel, for taking me under his wing from my first day at the institute, always ready to answer any question. I also want to thank Jeremias Mechler, for valuable insights and feedback in my defense preparation, but more importantly for years of supportive friendship with the best tea and chocolate breaks. Words can not express my appreciation to Astrid Ottenhues for her years of support, interesting discussions, heaps of motivation and a million good mornings. I would also like to express deep gratitude to my friend Joachim Breitner for always being there, lending me his ear, and providing intellectual, emotional and technical support.

I am grateful to all my collaborators on various research projects, but in particular Wasilij Beskorovajnov, Roland Gröll, Sarai Eilebrecht and Laurin Benz, who worked with me on topics around sender-binding encryption and enhanced my very theoretic perspective with a more practical view.

Thanks should also go to my mother Claudia Schwerdt for providing me with a strong role model as well as always supporting me in any endeavour I took on. Last, but certainly not least, I would like to express my deepest gratitude to my husband Sven Schwerdt for his unconditional love and support as well as to our amazing children for their curiosity, patience, and the firm belief that Mommy can do anything.

I've been posting my public key for 15 years now, but no one has ever asked me for it or used it for anything as far as I can tell. Maybe I should try posting my *private* key instead.

Randall Munroe, xkcd #1553

CHAPTER 1
Introduction

Secure communication is the oldest and arguably most important cryptographic problem. The first documented use of "cryptography" to hide the content of messages can be dated back to roughly 1900 BCE [WM11].

To this day, we face a similar challenge, just with vastly more common and daily applications. The construction of secure digital communication is the problem which initially gave birth to the whole field of modern cryptography. But even though it is the mother of all cryptographic applications, there are still some things our field of study has not fully answered yet.

Secure communication is usually thought to be comprised of three (simplified) aspects: No external adversary can…

(1) …know what a message says.

(2) …change what a message says.

(3) …change sender or receiver of a message.

The common way to facilitate secure communication in our modern digital era is to combine encryption with authentication, a division which was firstly introduced in 1976 by Diffie and Hellman [DH76]. Hence a common practice is to not construct secure communication from scratch, but rather assume authenticated channels as a given building block. In this case the authenticated channel takes care of the second and (most of the) third requirement while the rest—mainly secrecy of the message content—is left to be solved via some form of encryption.

Development on the encryption side has come a long way already. Firstly, symmetric encryption schemes were proposed, where each pair of parties needs a mutual secret of sufficient size to facilitate encryption and decryption. In 1976, Diffie and Hellman proposed the first public-key encryption scheme [DH76], giving rise to the field of asymmetric cryptography. With public-key encryption, every party generates their own secret and public key pair (sk, pk) and publishes the public part pk which is sufficient for other parties to encrypt messages for the holder of the secret key. There is no need for key exchanges or individual keys for every communication partner with public-key encryption schemes anymore. Several decades later, hybrid encryption [Sho01] was introduced, where keys are exchanged via a public-key key encapsulation mechanism and subsequently used to symmetrically encrypt messages via a data encapsulation mechanism (DEM). For many applications this compromise can combine the best of both worlds.

Let us firstly consider public-key encryption, before we move on to hybrid encryption via the KEM-DEM framework.

The strongest known encryption security notion is indistinguishability under adaptive chosen ciphertext attack (IND-CCA2). In 2001, Canetti showed that IND-CCA2 secure public-key encryption is strong enough to facilitate secure communication given authenticated channels [Can01]. This was done by concatenating the sender ID S to the message m before encryption, resulting in a ciphertext which contains both in a secret and non-malleable way (cf. Figure 1). Two years later, Canetti, Krawczyk and Nielsen noted in [CKN03] that while IND-CCA2 is strong

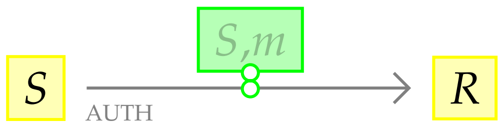


Figure 1: IND-CCA2 Paired with Authenticated Channel

enough for the purpose it is actually stronger than required. On an intuitive level this is because IND-CCA2 secure encryption also provides above property (2), which is already provided by the authenticated channel as well. Hence we arrive at the following research question:

> *What is the weakest public-key encryption security definition in order to establish secure communication from existing authenticated channels?*

This question is the first of two main research questions of this dissertation. Although I mainly value the question for its theoretic merit, it has real world implications. More and more communication is handled digitally and a growing portion of our digital communication is handled over secure channels. In the best case, using unnecessarily strong encryption to do so—just because we did not know that a weaker type of encryption yielded the same security levels while allowing for more efficient constructions—means wasting valuable time and resources. In the worst case, sensible communication is handled in an insecure way because the additional (actually unnecessary) resources can not be spared. But despite its importance, the above research question remains open for close to two decades now.

Canetti, Krawczyk and Nielsen started to answer it in 2003 by relaxing IND-CCA2 to indistinguishability under replayable chosen ciphertext attack (IND-RCCA) security [CKN03]. Another breakthrough was the use of tag-based encryption (TBE) for the purpose, where MacKenzie, Reiter and Yang used indistinguishability under adaptive-tag weakly chosen ciphertext attack (IND-atag-CCA) secure tag-based encryption to facilitate secure message transfer (SMT) from authenticated channels [MRY04]. They now added the sender ID S as the tag to the ciphertext because unlike the message content, S does not require any secrecy.

All this prior work is conducted in the universal composability (UC) framework, with secure communication defined as an ideal functionality which provides secure message transfer. Ideal authenticated channels are also defined in an ideal functionality while a protocol which fulfils this security notion is said to UC-realise secure message transfer.

In this dissertation I introduce the concept of sender-binding encryption (SBE) with the corresponding security notion of indistinguishability under sender-binding chosen plaintext attack (IND-SB-CPA) security. IND-SB-CPA intuitively provides three properties which complement the aspects provided by an authenticated channel. They are visualised in Figure 2.

1 The ciphertext contains sender ID S in a non-secret but non-malleable way. This binds the ciphertext to sender S.

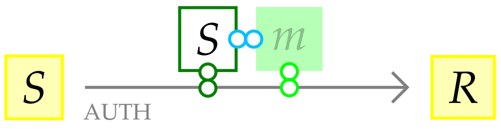


Figure 2: IND-SB-CPA Paired with Authenticated Channel

2 The ciphertext contains message m in a secret but malleable way. This binds the ciphertext to receiver R since it is only decryptable with R's secret key.

3 Both parts of the ciphertext are bound together.

Because this notion precisely complements authenticated channels, it unifies all prior approaches to the above research question: Every prior protocol realising secure message transfer from some form of public-key encryption and ideal authenticated channels can be construed as constructing an IND-SB-CPA secure sender-binding encryption scheme.

But I do not only prove that under static corruption and restriction to a polynomial set of participating parties, IND-SB-CPA secure sender-binding encryption is sufficiently strong to UC-realise secure message transfer if we assume ideal authenticated channels. For the first time with regards to this research question, I am also able to provide an optimality result: If we restrict ourselves to the encrypt-then-authenticate (ETA) principle, which is the de facto standard to combine authentication and encryption, *any* generalised public-key encryption scheme which UC-realises multiple secure message transfer from ideal authenticated channels is necessarily an IND-SB-CPA secure sender-binding encryption scheme. Hence under the mentioned restrictions, IND-SB-CPA security is the weakest possible security notion and fully answers our research question.

While public-key encryption is still widely used, many real world applications have moved on to using hybrid encryption instead. In the field of hybrid encryption, however, the question of how strong (or weak) encryption should be for secure communication has so far been largely ignored. Constructing an IND-CCA2 secure public-key encryption scheme is seen as the only significant goal. The fact that in practice, encryption schemes are usually paired with authentication to gain secure communication, and that authentication already provides some non-malleability properties which do not need to be provided by the encryption scheme anymore, has unfortunately not gotten much attention, yet.

Looking at the first research question above, however, a second corresponding one instantly presents itself:

What is the weakest hybrid encryption security definition in order to establish secure communication from existing authenticated channels?

For the second part of this dissertation, I apply the sender-binding concept to hybrid encryption by developing the notion of a sender-binding key encapsulation mechanism for which I also define IND-SB-CPA security. I prove that only the weakest known data encapsulation mechanism is necessary to construct and IND-SB-CPA secure sender-binding encryption from an IND-SB-CPA secure key encapsulation mechanism. This in turn provides a way to UC-realise multiple secure message transfer from ideal authenticated channels by the first part of the dissertation. Previously, this has only been achieved from hybrid encryption via IND-CCA2 secure public-key encryption which also requires the respective key encapsulation mechanism and data encapsulation mechanism to be IND-CCA2 secure [CS02; HHK10].

Furthermore I look at secure session communication, where each symmetric data encapsulation mechanism key is persistently used for several messages. For this version of hybrid encryption I prove that under static corruption and restriction to a polynomial set of participating parties, an IND-SB-CPA secure sender-binding key encapsulation mechanism is sufficiently strong to UC-realise secure channels if we employ an indistinguishability under chosen plaintext attack (IND-CPA) secure data encapsulation mechanism and assume ideal authenticated channels. Again, the only previous realisation proof for secure channels requires both the key encapsulation mechanism and data encapsulation mechanism to be IND-CCA2 secure [NMO06]. The question about optimality of IND-SB-CPA security regarding hybrid encryption had to be left open at this point.

In the following section I explain the full contribution of this dissertation in more detail. But before I do so, I want to shortly explain my choices regarding writing conventions on personal pronouns. As I would most like to invite you, the reader, to join a conversation with me about my work, I will refrain from the schools of writing which promote the strict use of "the author" or passive voice. I will mostly use the pronoun "I" instead. Similarly, I will not talk passively about "the reader" but directly address you in an invitation to follow my explanations. Also for this purpose, large parts of this dissertation use the pronoun "we"—not to indicate several authors or follow some writing convention, but rather to denote "you and me, just as if we were standing in front of a blackboard where I walk you through my proof" [Bre16].

1.1 Contribution and Overview

In this section I give a more detailed summary of all contributions provided in this dissertation. The contributions are separately visualised regarding public-key encryption and hybrid encryption in Figures 3 and 4, respectively.[1]Novel definitions are indicated by light green colour, UC-related proofs via dark green, generic game-based transformations are coloured in light blue, theoretic classifications highlighted yellow and efficient protocols—which are not my own work, but which I briefly discuss for completeness—are greyed out purple.

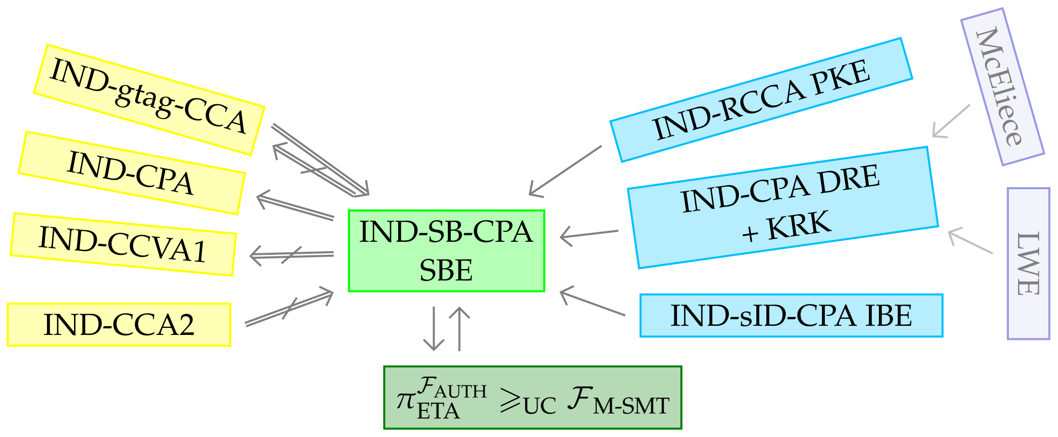


Figure 3: Overview of Public-key Encryption Contribution

[1] With a lot of acronyms which are not vital to understand at this point if you are not already familiar with them.

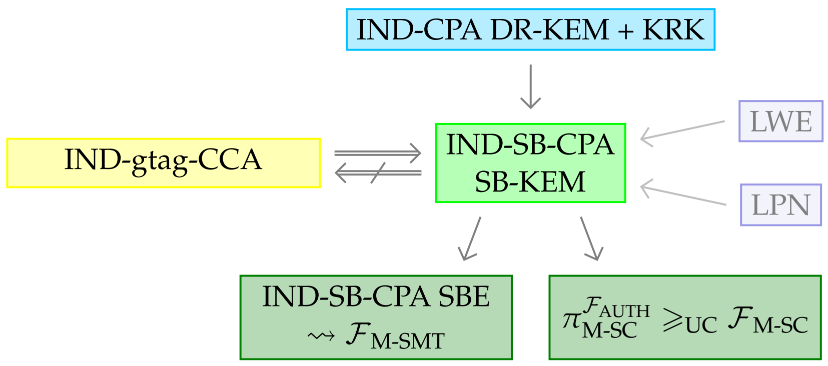

IND-CPA DR-KEM + KRK
IND-gtag-CCA
IND-SB-CPA SB-KEM
LWE
LPN
IND-SB-CPA SBE $\rightsquigarrow \mathcal{F}_{\text{M-SMT}}$
$\pi_{\text{M-SC}}^{\mathcal{F}_{\text{AUTH}}} \geq_{\text{UC}} \mathcal{F}_{\text{M-SC}}$

Figure 4: Overview of Hybrid Encryption Contribution

In addition to indicating all the different contributions, the following list provides an overview of the structure of this dissertation and its individual chapters:

- Chapter 3 is fully dedicated to the first research question concerning public-key encryption. After looking at related work in Section 3.1, I develop the central new concepts of sender-binding encryption and IND-SB-CPA security (🟩) in Section 3.2. I then go on to provide a comprehensive proof in the UC framework that under static corruption the protocol π_{ETA}, which combines sender-binding encryption with authenticated channels, UC-realises the ideal secure message transfer functionality $\mathcal{F}_{\text{M-SMT}}$ in the $\mathcal{F}_{\text{AUTH}}$-hybrid model if the sender-binding encryption scheme satisfies IND-SB-CPA security (🟩). This is done in Section 3.3. The last part of this chapter, Section 3.4, considers the reverse direction. I show that using the encrypt-then-authenticate principle to combine encryption with ideal authentication, every generalised public-key encryption scheme which is strong enough to allow UC realisation of $\mathcal{F}_{\text{M-SMT}}$ in the $\mathcal{F}_{\text{AUTH}}$-hybrid model necessarily is some form of sender-binding encryption which satisfies IND-SB-CPA security (🟩).

- Chapter 4 is structured similar to Chapter 3 but concerning hybrid encryption. I firstly look at related work in Section 4.1 before introducing the notion of a sender-binding key encapsulation mechanism with its respective IND-SB-CPA security notion (🟩) in Section 4.2. In Section 4.3 I show how to generically construct IND-SB-CPA secure sender-binding encryption from an IND-SB-CPA secure sender-binding key encapsulation mechanism combined with an indistinguishability under one-time attack (IND-OT) secure data encapsulation mechanism (🟩)[2]. For the last part of this chapter, I move from single-message to session communication, where each symmetric key is used to encrypt multiple messages: In Section 3.3, I prove in the UC framework that under static corruption the protocol $\pi_{\text{M-SC}}$ combining an IND-SB-CPA secure sender-binding key encapsulation mechanism and IND-CPA secure data encapsulation mechanism with authenticated channels UC-realises the ideal secure channel functionality $\mathcal{F}_{\text{M-SC}}$ in the $\mathcal{F}_{\text{AUTH}}$-hybrid model (🟩).

- Chapter 5 details how to generically transform other types of public-key encryption and key encapsulation mechanisms into IND-SB-CPA secure sender-binding schemes. In Section 5.1 I provide various generic transformations from classic public-key encryption, dual receiver encryption (DRE) as well as identity-based encryption (IBE) schemes to

[2] Note that although this can be thought of as a generic construction (🟦), in this case I view it as a detour on the way to UC-realizing secure message transfer (🟩), where the remaining step is taken care of by Section 3.3.

IND-SB-CPA secure sender-binding encryption (🟦). Section 5.2 gives a similar generic transformation from a dual receiver key encapsulation mechanism (DR-KEM) to sender-binding key encapsulation (🟦).

- In Chapter 6 we analyse how IND-SB-CPA security can be contextualised with regards to other game-based security notions. Again, we begin with the sender-binding encryption notion in Section 6.1 (🟨) before moving on to the sender-binding key encapsulation notion in Section 6.1 (🟨). In both cases we see that comparisons are non-trivial with respect to game-based notions for classic public-key encryption schemes and key encapsulation mechanisms while IND-SB-CPA can easily shown to be weaker than the weakest known respective tag-based security notion.
- Chapter 7 contains a short discussion of several McEliece-, learning parity with noise (LPN)- and learning with errors (LWE)-based protocols for IND-SB-CPA secure sender-binding encryption and sender-binding key encapsulation mechanisms (🟪). All of the constructions were proven secure in the standard model and are at least competitively efficient with current encryption and key encapsulation protocols in the standard model which are strong enough to facilitate secure communication—sometimes even promising slight efficiency gains or providing additional properties like post-quantum security. Note that none of the protocols I discuss in Chapter 7 are my own work (which is why they are greyed out in Figures 3 and 4) and I merely present some of the results to show the potential real-world impact of my exclusively theoretic contributions.

Math tip: If one of your equations ever looks too simple, try adding some purely decorative constants.

Randall Munroe, xkcd #2566

CHAPTER 2
Preliminaries

Before we proceed with the main parts of this dissertation, I will provide you with all the definitional basics needed to understand the upcoming chapters. The two main areas we encounter throughout Chapters 3 to 7 are game-based and simulation-based security notions. For game-based security, I recap the formal definitions of different types of public-key encryption schemes, the KEM-DEM framework for hybrid encryption and various corresponding security notions in Section 2.1. Afterwards I briefly explain the principles of simulation-based security and the UC framework as well as formal definitions of relevant ideal functionalities in Section 2.2.

This chapter can be used as an introduction to the basics but—especially for game-based definitions—is mainly designed as a place to look up details of definitions while reading later chapters. Hence you should feel free to skip sections about security concepts you are already familiar with. For additional look-ups there are lists of acronyms as well as variables provided in Appendices A and B, respectively. Note that if you are reading this as a digital copy, the respective entries in these lists can also be reached by clicking on acronyms or variables in the text.

Some preliminary parts are taken verbatim or with slight changes from [Sch24; SBB+23b; BGMOS21].

One central definition we encounter constantly throughout this chapter with both game-based and simulation-based security is the notion of negligibility:

Definition (Negligible Function):
A function $\varepsilon : \mathbb{N} \to \mathbb{R}$ is called *negligible* if and only if for any polynomial $\mathsf{p} : \mathbb{N} \to \mathbb{R}^+$ there exists a $\lambda_0 \in \mathbb{N}$ such that for all $\lambda \geqslant \lambda_0$ we have

$$\varepsilon(\lambda) \leqslant \frac{1}{\mathsf{p}(\lambda)}. \qquad \circ$$

Note in particular, that by this definition the function

$$\mathsf{q} \cdot \varepsilon = \Big(\lambda \mapsto \mathsf{q}(\lambda) \cdot \varepsilon(\lambda)\Big)$$

is negligible as well for any negligible ε and polynomial $\mathsf{q} : \mathbb{N} \to \mathbb{R}^+$.

2.1 Game-based Security

Game-based security comes in the form of a formally defined game between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$. A primitive is deemed secure with respect to a certain security notion if there is no adversary who can win the corresponding security game with non-negligible success probability. For distinguishing security notions this is taken to mean "non-negligibly better than guessing", i.e. the adversary's respective advantage, instead of plain non-negligibility. The adversary $\mathcal{A}$ usually also has access to one or multiple oracles $\mathcal{O}$ which give them a better chance to win the security game by providing additional information.

For the sake of brevity I will not explicitly write out the security games as interactions between challengers and adversaries in this section but rather directly note down resulting advantages where the code of the challenger is inserted and only adversaries and oracles are named as probabilistic polynomial time (PPT) participants.

The formal definitions of different types of public-key encryption schemes are given alongside corresponding security notions in Section 2.1.1. In Section 2.1.2 hybrid encryption via the KEM-DEM framework is explained with the different KEM/DEM security notions provided in Section 2.1.3.

2.1.1 Different Public-key Encryption Schemes and Security Notions

In this section we look at definitions and security notions for different types of public-key encryption schemes. They include classic public-key encryption (PKE), tag-based encryption (TBE), dual receiver encryption (DRE) and identity-based encryption (IBE).

All of the below security games are concerned with indistinguishability of ciphertexts. This means they each follow the same basic structure:

- The challenge setting is determined.
- The adversary chooses two challenge messages m_0, m_1.
- A random bit $b \overset{\$}{\leftarrow} \{0,1\}$ is drawn and message m_b encrypted in the challenge setting to obtain the challenge ciphertext c^*.
- The adversary tries to determine/guess bit b from ciphertext c^*.

Although it might seem repetitive to give the identical parts of the security definitions again and again, I choose to present each one in a self-contained way to provide easy access to all necessary information when a specific definition is looked up.

Classic Public-key Encryption (PKE)

The historically first developed concept of symmetric encryption relies on two parties to share a mutual secret which enables them to securely communicate. This mutual secret is needed to both encrypt messages and decrypt ciphertexts. In contrast, for public-key encryption each party generates their own secret and public key pair (sk, pk). If published, the public part pk enables anyone to encrypt messages while decryption is only possible with the secret key sk.

Definition (Classic public-key encryption):
A *classic public-key encryption (PKE) scheme* $\Sigma = (\texttt{gen}, \texttt{enc}, \texttt{dec})$ (cf. [BDPR98]) consists of three PPT algorithms:

$$\begin{aligned} \texttt{gen}: &\quad 1^\lambda \mapsto (sk, pk) \\ \texttt{enc}: &\quad (pk, m) \mapsto c \\ \texttt{dec}: &\quad (sk, c) \mapsto m \end{aligned}$$

On input of the security parameter λ, the key generation algorithm $\texttt{gen}$ generates a pair of secret and public keys sk and pk. The encryption algorithm $\texttt{enc}$ produces a ciphertext c by input of a public key pk and a message m. The decryption algorithm $\texttt{dec}$ takes a secret key sk and ciphertext c as input, outputting a message m. A public-key encryption scheme is required to satisfy *correctness*, i.e. for all messages m in the message space $\mathbf{M}$ we have

$$m = \texttt{dec}\Big(sk, \texttt{enc}(pk, m)\Big)$$

whenever $(sk, pk) \leftarrow \texttt{gen}(1^\lambda)$. $\circ$

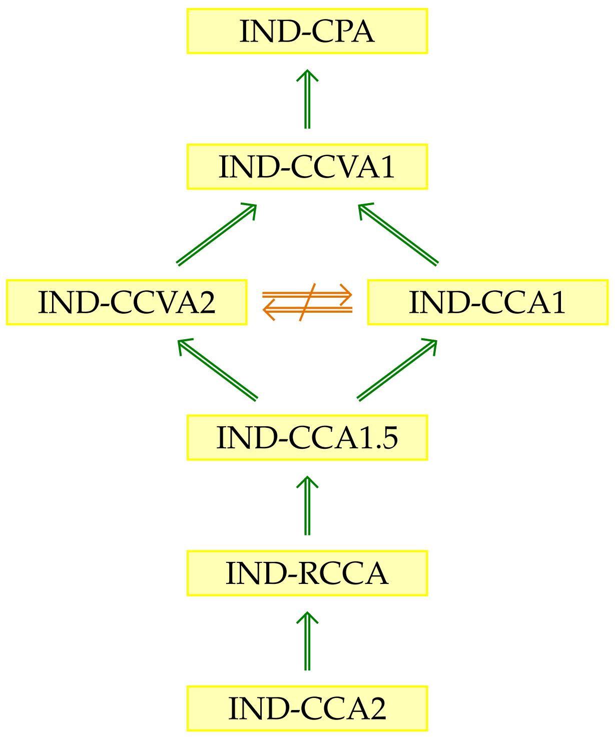


Figure 5: Classic Public-key Encryption Security Notions

Figure 5 gives an overview of the different indistinguishability security notions known for classic public-key encryption and their implicational hierarchy. The definitions of the two strongest (IND-CCA2 and -RCCA) notions as well as the two weakest ones (IND-CPA and -CCVA1) will later on be relevant to give context to the weakness of the new IND-SB-CPA security notion and are therefore included in this section. All the classic public-key encryption security notions only differ in the amount of oracle access provided to the adversary.

Definition (IND-CCA2):
A classic public-key encryption scheme $\Sigma = (\texttt{gen}, \texttt{enc}, \texttt{dec})$ satisfies *indistinguishability under adaptive chosen ciphertext attack (IND-CCA2)* (cf. [BDPR98]), if and only if for any PPT adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ the advantage

$$\begin{aligned}\mathsf{Adv}^{\mathrm{CCA2}}_{\mathcal{A},\Sigma}(\lambda) := \Big|\mathbb{P}\Big[b \leftarrow \mathcal{A}_2^{\mathcal{O}^*}(c^*, aux) \,\Big|\, &(sk, pk) \leftarrow \texttt{gen}(1^\lambda);\\ &(m_0, m_1, aux) \leftarrow \mathcal{A}_1^{\mathcal{O}}(pk); b \xleftarrow{\$} \{0,1\};\\ &c^* \leftarrow \texttt{enc}(pk, m_b)\Big] - \tfrac{1}{2}\Big|\end{aligned}$$

is negligible in λ, where $\mathcal{O}$ denotes the oracle $\texttt{dec}(sk, \cdot)$ and $\mathcal{O}^*$ denotes the oracle

$$\mathcal{O}^*(c) = \begin{cases} \perp, & c = c^* \\ \mathcal{O}(c), & \text{otherwise.} \end{cases}$$ ∘

The notion of IND-RCCA security grants slightly less oracle access to the adversary in the second oracle phase where no ciphertext is decrypted which contains one of the challenge messages m_0, m_1. Because this obviously yields a strictly weaker security notion it is called a relaxation of IND-CCA2. More formally:

Definition (IND-RCCA):
A classic public-key encryption scheme $\Sigma = (\texttt{gen}, \texttt{enc}, \texttt{dec})$ satisfies *indistinguishability under replayable chosen ciphertext attack (IND-RCCA)* (cf. [CKN03]), if and only if for any PPT adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ the advantage

$$\begin{aligned} \mathsf{Adv}^{\mathrm{RCCA}}_{\mathcal{A},\Sigma}(\lambda) := \Big| \mathbb{P}\Big[b \leftarrow \mathcal{A}_2^{\mathcal{O}'}(c^*, aux) \;\Big|\; & (sk, pk) \leftarrow \texttt{gen}(1^\lambda); \\ & (m_0, m_1, aux) \leftarrow \mathcal{A}_1^{\mathcal{O}}(pk); b \xleftarrow{\$} \{0,1\}; \\ & c^* \leftarrow \texttt{enc}(pk, m_b) \Big] - \tfrac{1}{2} \Big| \end{aligned}$$

is negligible in λ, where $\mathcal{O}$ denotes the oracle $\texttt{dec}(sk, \cdot)$ and $\mathcal{O}'$ denotes the oracle

$$\mathcal{O}'(c) = \begin{cases} \perp, & \mathcal{O}(c) \in \{m_0, m_1\} \\ \mathcal{O}(c), & \text{otherwise.} \end{cases}$$ ∘

On the other end of the security spectrum lie the security notions of IND-CCVA1 and IND-CPA. Now, instead of any decryption possibilities, at most some verification of whether a ciphertext is well-formed is provided to the adversary.

Definition (IND-CCVA1):
A classic public-key encryption scheme $\Sigma = (\texttt{gen}, \texttt{enc}, \texttt{dec})$ satisfies *indistinguishability under non-adaptive chosen ciphertext verification attack (IND-CCVA1)* (cf. [PSJ12; DDA13]), if and only if for any PPT adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ the advantage

$$\begin{aligned} \mathsf{Adv}^{\mathrm{CCVA1}}_{\mathcal{A},\Sigma}(\lambda) := \Big| \mathbb{P}\Big[b \leftarrow \mathcal{A}_2(c^*, aux) \;\Big|\; & (sk, pk) \leftarrow \texttt{gen}(1^\lambda); \\ & (m_0, m_1, aux) \leftarrow \mathcal{A}_1^{\mathcal{O}^\perp}(pk); b \xleftarrow{\$} \{0,1\}; \\ & c^* \leftarrow \texttt{enc}(pk, m_b) \Big] - \tfrac{1}{2} \Big| \end{aligned}$$

is negligible in λ, where $\mathcal{O}^\perp$ denotes the verification oracle

$$\mathcal{O}^\perp(c) = \begin{cases} \mathsf{false}, & \texttt{dec}(sk, c) = \perp \\ \mathsf{true}, & \text{otherwise.} \end{cases}$$ ∘

IND-CPA security denotes the weakest possible indistinguishability notion where the adversary has no oracle access at all:

Definition (IND-CPA):
A classic public-key encryption scheme $\Sigma = (\mathtt{gen}, \mathtt{enc}, \mathtt{dec})$ satisfies *indistinguishability under chosen plaintext attack (IND-CPA)* (cf. [GM84]), if and only if for any PPT adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ the advantage

$$\begin{aligned}\mathsf{Adv}^{\mathrm{CPA}}_{\mathcal{A},\Sigma}(\lambda) := \Big|\mathbb{P}\Big[b \leftarrow \mathcal{A}_2(c^*, aux) \;\Big|\; &(sk, pk) \leftarrow \mathtt{gen}(1^\lambda);\\ &(m_0, m_1, aux) \leftarrow \mathcal{A}_1(pk); b \xleftarrow{\$} \{0,1\};\\ &c^* \leftarrow \mathtt{enc}(pk, m_b)\Big] - \tfrac{1}{2}\Big|\end{aligned}$$

is negligible in λ. ∘

Tag-based Encryption (TBE)

Tag-based encryption (TBE) corresponds to a classic public-key encryption scheme that is enhanced by the use of tags. Both encryption and decryption take a tag from a predefined tag space as additional arguments. We can see from the correctness property that tag-based encryption is intended to be employed with identical tags for encryption and subsequent decryption.

Definition (Tag-based Encryption):
For a *tag-based encryption (TBE) scheme* Σ (cf. [MRY04]) with message space $\mathbf{M}$ and tag space $\mathbf{T}$ the three PPT algorithms $(\mathtt{gen}, \mathtt{enc}, \mathtt{dec})$ are enhanced by tags $\tau \in \mathbf{T}$:

$$\begin{aligned}\mathtt{gen}:&\quad 1^\lambda \mapsto (sk, pk)\\ \mathtt{enc}:&\quad (pk, \tau, m) \mapsto c\\ \mathtt{dec}:&\quad (sk, \tau, c) \mapsto m.\end{aligned}$$

Other than the additional tags for enc and dec these algorithms correspond to the classical public-key encryption interface. Correspondingly, a tag-based encryption scheme is expected to fulfil the notion of *correctness*, i.e. that whenever $(sk, pk) \leftarrow \mathtt{gen}(1^\lambda)$, then for all $\tau \in \mathbf{T}$ and $m \in \mathbf{M}$

$$m = \mathtt{dec}\Big(sk, \tau, \mathtt{enc}(pk, \tau, m)\Big).$$

∘

Let us now move on to tag-based encryption security notions. The two notions IND-atag-CCA, IND-stag-CCA I explain below again have the same structure and only slightly differ in the amount of oracle access given to the adversary. More importantly, however, they differ in when the challenge tag τ^* is chosen. This is highlighted in blue.

Definition (IND-atag-CCA):
A tag-based encryption scheme $\Sigma = (\mathtt{gen}, \mathtt{enc}, \mathtt{dec})$ satisfies *indistinguishability under adaptive-tag weakly chosen ciphertext attack (IND-atag-CCA)* (cf. [MRY04]), if and only if for any PPT adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ the advantage

$$\begin{aligned}\mathsf{Adv}^{\text{atag-CCA}}_{\mathcal{A},\Sigma}(\lambda) := \Big|\mathbb{P}\Big[b \leftarrow \mathcal{A}_2^{\mathcal{O}_{\tau^*}}(c^*, aux) \;\Big|\; &(sk, pk) \leftarrow \mathtt{gen}(1^\lambda);\\ &(m_0, m_1, \tau^*, aux) \leftarrow \mathcal{A}_1^{\mathcal{O}}(pk); b \xleftarrow{\$} \{0,1\};\\ &c^* \leftarrow \mathtt{enc}(pk, \tau^*, m_b)\Big] - \tfrac{1}{2}\Big|\end{aligned}$$

is negligible in λ, where $\mathcal{O}$ denotes the full oracle $\mathtt{dec}(sk,\cdot,\cdot)$ and $\mathcal{O}_{\tau^*}$ denotes the punctured oracle

$$\mathcal{O}_{\tau^*}(\tau,c) = \begin{cases} \bot, & \tau = \tau^* \\ \mathcal{O}(\tau,c), & \text{otherwise.} \end{cases}$$
○

Forcing the adversary instead to determine the challenge tag τ^* at the start of the game without any prior input results in the weaker security notion of IND-stag-CCA.

Definition (IND-stag-CCA):
A tag-based encryption scheme $\Sigma = (\mathtt{gen}, \mathtt{enc}, \mathtt{dec})$ satisfies *indistinguishability under selective-tag weakly chosen ciphertext attack (IND-stag-CCA)* (cf. [Kil06]), if and only if for any PPT adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3)$ the advantage

$$\begin{aligned} \mathsf{Adv}^{\text{stag-CCA}}_{\mathcal{A},\Sigma}(\lambda) := \Big|\mathbb{P}\Big[b \leftarrow \mathcal{A}_3^{\mathcal{O}_{\tau^*}}(c^*, aux_2) \,\Big|\, &(\tau^*, aux_1) \leftarrow \mathcal{A}_1(1^\lambda); \\ &(sk,pk) \leftarrow \mathtt{gen}(1^\lambda); (m_0, m_1, aux_2) \leftarrow \mathcal{A}_2^{\mathcal{O}_{\tau^*}}(pk, aux_1); \\ &b \xleftarrow{\$} \{0,1\}; c^* \leftarrow \mathtt{enc}(pk, \tau^*, m_b)\Big] - \tfrac{1}{2}\Big| \end{aligned}$$

is negligible in λ, where $\mathcal{O}$ denotes the full oracle $\mathtt{dec}(sk,\cdot,\cdot)$ and $\mathcal{O}_{\tau^*}$ denotes the punctured oracle

$$\mathcal{O}_{\tau^*}(\tau,c) = \begin{cases} \bot, & \tau = \tau^* \\ \mathcal{O}(\tau,c), & \text{otherwise.} \end{cases}$$
○

Dual Receiver Encryption (DRE)

The idea behind dual receiver encryption (DRE) is to encrypt each plaintext for two different recipients instead of just one.

Definition (Dual Receiver Encryption):
Formally a *dual receiver encryption (DRE)* scheme $\Sigma = (\mathtt{gen}, \mathtt{enc}, \mathtt{dec})$ (cf. [DLKY04]) with message space $\mathbf{M}$ consists of three PPT algorithms:

$$\begin{aligned} \mathtt{gen}:&\quad 1^\lambda \mapsto (sk,pk) \\ \mathtt{enc}:&\quad (pk_1, pk_2, m) \mapsto c \\ \mathtt{dec}:&\quad (sk_i, pk_1, pk_2, c) \mapsto m, \text{ where } i \in \{1,2\}. \end{aligned}$$

A dual receiver encryption scheme is also required to satisfy *correctness*, i.e. that for every $m \in \mathbf{M}$

$$m = \mathtt{dec}\Big(sk_i, pk_1, pk_2, \mathtt{enc}(pk_1, pk_2, m)\Big)$$

whenever $(sk_1, pk_1), (sk_2, pk_2) \leftarrow \mathtt{gen}(1^\lambda)$ and $i \in \{1,2\}$. ○

Of course, dual receiver encryption could be constructed from a classic public-key encryption scheme by encrypting a message twice under the two respective public keys and concatenating the ciphertexts. What makes dual receiver encryption schemes interesting, however, is not only accomplishing this in a more efficient manner than using two separate encryptions. More importantly, a dual receiver encryption scheme might provide additional properties. The most common such property—and the only one I use in this dissertation—is soundness. Soundness provides a guarantee to the two decrypting parties that they in fact both got the same message instead of two different ones.

Definition (Soundness):
A dual receiver encryption scheme $\Sigma = (\texttt{gen}, \texttt{enc}, \texttt{dec})$ satisfies *soundness* (cf. [CFZ14]), if and only if for any PPT adversary $\mathcal{A}$ the advantage

$$\begin{aligned}\mathsf{Adv}^{\text{sound}}_{\mathcal{A},\Sigma}(\lambda) := \mathbb{P}\Big[&\texttt{dec}(sk_1, pk_1, pk_2, c) \neq \texttt{dec}(sk_2, pk_1, pk_2, c) \mid \\ &(sk_1, pk_1), (sk_2, pk_2) \leftarrow \texttt{gen}(1^\lambda); \\ &c \leftarrow \mathcal{A}(sk_1, pk_1, sk_2, pk_2)\Big]\end{aligned}$$

is negligible in λ. ◦

The only dual receiver encryption security notion we encounter is plain IND-CPA security. As with classic public-key encryption this denotes an indistinguishability game which does not provide any oracle access to the adversary:

Definition (IND-CPA for DRE):
A dual receiver encryption scheme $\Sigma = (\texttt{gen}, \texttt{enc}, \texttt{dec})$ satisfies *indistinguishability under chosen plaintext attack (IND-CPA)* (cf. [DLKY04]), if and only if for any PPT adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ the advantage

$$\begin{aligned}\mathsf{Adv}^{\text{CPA}}_{\mathcal{A},\Sigma}(\lambda) := \Big|\mathbb{P}\Big[&b \leftarrow \mathcal{A}_2(c^*, aux) \mid (sk_1, pk_1), (sk_2, pk_2) \leftarrow \texttt{gen}(1^\lambda); \\ &(m_0, m_1, aux) \leftarrow \mathcal{A}_1(pk_1, pk_2); b \xleftarrow{\$} \{0, 1\}; \\ &c^* \leftarrow \texttt{enc}(pk_1, pk_2, m_b)\Big] - \tfrac{1}{2}\Big|\end{aligned}$$

is negligible in λ. ◦

Identity-based Encryption (IBE)

The first practical identity-based encryption (IBE) scheme was proposed by Boneh and Franklin in [BF01]. The idea behind identity-based encryption is that as soon as a scheme is initialised by generation of a master secret and public key pair (msk, mpk), individual user secret keys usk can be derived from the master secret key msk with the user's ID. Encryption now only requires knowledge of the master public key and a party's identity, alleviating the need to publish individual public keys.

Definition (Identity-based Encryption):
An *identity-based encryption (IBE)* scheme $\Sigma = (\texttt{gen}, \texttt{ext}, \texttt{enc}, \texttt{dec})$ (cf. [BF01]) with message space $\mathbf{M}$ and identity space $\mathbf{ID}$ consists of four PPT algorithms:

$$\begin{aligned}\texttt{gen}:&\quad (1^\lambda) \mapsto (mpk, msk)\\ \texttt{ext}:&\quad (msk, id) \mapsto usk_{id}\\ \texttt{enc}:&\quad (mpk, id, m) \mapsto c\\ \texttt{dec}:&\quad (usk_{id}, id, c) \mapsto m\end{aligned}$$

For correctness, we require for all IDs $id \in \mathbf{ID}$ and all messages $m \in \mathbf{M}$ that

$$m = \texttt{dec}\Big(usk_{id}, id, \texttt{enc}(mpk, id, m)\Big)$$

whenever $(mpk, msk) \leftarrow \texttt{gen}(1^\lambda)$, and $usk_{id} \leftarrow \texttt{ext}(msk, id)$. ◦

The identity-based encryption security notion we encounter in Section 5.1.3 is IND-sID-CPA. This is the weakest common identity-based encryption security notion and similar to the IND-stag-CCA notion for tag-based encryption. It is mainly characterised by the adversary having to choose the challenge ID id^* right at the start of the game.

Definition (IND-sID-CPA):
An identity-based encryption scheme $\Sigma = (\mathtt{gen}, \mathtt{ext}, \mathtt{enc}, \mathtt{dec})$ satisfies *indistinguishability under selective identity chosen plaintext attack (IND-sID-CPA)* (cf. [BCHK07]), if and only if for any PPT adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3)$ the advantage

$$\begin{aligned}\mathsf{Adv}^{\text{sID-CPA}}_{\mathcal{A},\Sigma}(\lambda) := \Big|\mathbb{P}\Big[b \leftarrow \mathcal{A}_3^{\mathcal{O}_{id^*}}(c^*, aux_2) \,\Big|\, & (id^*, aux_1) \leftarrow \mathcal{A}_1(1^\lambda); (msk, mpk) \leftarrow \mathtt{gen}(1^\lambda);\\ & (m_0, m_1, aux_2) \leftarrow \mathcal{A}_2^{\mathcal{O}_{id^*}}(mpk, aux_1); b \xleftarrow{\$} \{0,1\};\\ & c^* \leftarrow \mathtt{enc}(mpk, id^*, m_b)\Big] - \tfrac{1}{2}\Big|\end{aligned}$$

is negligible in λ, where $\mathcal{O}$ denotes the full extraction oracle $\mathtt{ext}(msk, \cdot)$ and $\mathcal{O}_{id^*}$ denotes the punctured oracle

$$\mathcal{O}_{id^*}(id) = \begin{cases} \perp, & id = id^* \\ \mathcal{O}(msk, id), & \text{otherwise.} \end{cases}$$

○

2.1.2 Hybrid Encryption via the KEM-DEM Framework

In Chapter 4 of this dissertation we switch from pure public-key encryption to hybrid encryption via the KEM-DEM framework. The KEM-DEM framework was introduced by Cramer and Shoup in [CS02] and subsequently included in the encryption ISO standard in 2006 [Sho01; ISO06]. With this form of hybrid encryption the advantages of both public-key and symmetric primitives are combined: Encrypting actual messages with symmetric encryption allows for more efficiency while a public key infrastructure alleviates the need for a cumbersome key exchange protocol.

Hence the KEM-DEM framework consists of two modular components. The first component is a public-key key encapsulation mechanism (KEM). With the use of a public key pk, the encapsulation algorithm of a key encapsulation mechanism simultaneously generates a symmetric key K and its encryption C. The encryption C can subsequently be decrypted to the key K with the corresponding secret key sk. More formally:

Definition (KEM):
A *key encapsulation mechanism (KEM)* is given by a set of three PPT algorithms $(\mathtt{gen}, \mathtt{enc}, \mathtt{dec})$:

$$\begin{aligned} \mathtt{gen}:\quad & 1^\lambda \mapsto (sk, pk)\\ \mathtt{enc}:\quad & pk \mapsto (K, C)\\ \mathtt{dec}:\quad & (sk, C) \mapsto K \end{aligned}$$

such that the correctness property holds, i.e. that $K = \mathtt{dec}(sk, C)$ whenever $(sk, pk) \leftarrow \mathtt{gen}(1^\lambda)$ and $(K, C) \leftarrow \mathtt{enc}(pk)$. ○

The second component is a symmetric data encapsulation mechanism (DEM) which uses a symmetric key K to encrypt and decrypt messages:

Definition (DEM):
A *data encapsulation mechanism (DEM)* with key space $\mathbf{K}$ and message space $\mathbf{M}$ is given by a set of two PPT algorithms $(\texttt{DEM.enc}, \texttt{DEM.dec})$:

$$\begin{aligned}\texttt{DEM.enc}:\ & (K, m) \mapsto c \\ \texttt{DEM.dec}:\ & (K, c) \mapsto m\end{aligned}$$

such that for all keys $K \in \mathbf{K}$ and messages $m \in \mathbf{M}$ the correctness property

$$m = \texttt{DEM.dec}\Big(K, \texttt{DEM.enc}(K, m)\Big)$$

holds. ∘

To combine both components into KEM-DEM hybrid encryption the key encapsulation mechanism and data encapsulation mechanism need to be compatible with each other, i.e. use the same key space/distribution $\mathbf{K}$ for the symmetric key K.

The KEM-DEM framework comes in two flavours which slightly differ in how the key encapsulation and data encapsulation primitives are combined. One construction—which I call *single-message communication*—generates a fresh symmetric key for each encryption of a message. This is the original definition of the KEM-DEM framework as introduced in [CS02] and intuitively yields a public-key encryption scheme $(\texttt{Gen}, \texttt{Enc}, \texttt{Dec})$ via $\texttt{Gen} \equiv \texttt{gen}$ and

```
Enc(pk, m):
  • $(K, C) \leftarrow \texttt{enc}(pk)$.
  • $c \leftarrow \texttt{DEM.enc}(K, m)$.
  ↪ Return $(C, c)$.

Dec(sk, (C, c)):
  • $K \leftarrow \texttt{dec}(sk, C)$.
  • $m \leftarrow \texttt{DEM.dec}(K, c)$.
  ↪ Return $m$.
```

For *session communication* on the other hand [NMO06], one party generates a persistent symmetric key via the key encapsulation mechanism and sends it to the communication partner. The generating party does not need a key encapsulation key pair themselves. The symmetric session key is only sent once at the start of the session and subsequently used for many messages between the two involved parties. Session communication KEM-DEM can be viewed as a classic symmetric encryption scheme where the key exchange is handled via key encapsulation mechanism.

Tag-based Key Encapsulation

A slight variation of classical key encapsulation mechanisms are tag-based key encapsulation mechanisms (tag-KEMs) [AGKS05] which additionally use a tag τ to encapsulate and decapsulate the symmetric key K. The encapsulation phase of the tag-based key encapsulation mechanism is split into two separate phases: A first phase that generates the symmetric key via an algorithm `key` and a second phase that encapsulates the given symmetric key using the tag τ and algorithm `enc`. The split is made to allow for the tag τ to depend on the symmetric key K.

Definition (Tag-KEM):
A *tag-based key encapsulation mechanism* with tag space $\mathbf{T}$ is given by a set of four PPT algorithms $(\texttt{gen}, \texttt{key}, \texttt{enc}, \texttt{dec})$ with

$$\begin{aligned} \texttt{gen}: &\quad 1^\lambda \mapsto (sk, pk) \\ \texttt{key}: &\quad pk \mapsto (K, aux) \\ \texttt{enc}: &\quad (\tau, aux) \mapsto C \\ \texttt{dec}: &\quad (sk, \tau, C) \mapsto K \end{aligned}$$

such that the correctness property holds, i.e. for every tag $\tau \in \mathbf{T}$ we have $K = \texttt{dec}(sk, \tau, C)$ whenever

$$\begin{aligned} (sk, pk) &\leftarrow \texttt{gen}(1^\lambda) \\ (K, aux) &\leftarrow \texttt{key}(pk) \\ C &\leftarrow \texttt{enc}(\tau, aux). \end{aligned}$$ ◦

When introducing tag-based key encapsulation mechanisms, Abe et al. [AGKS05] employ them in a slightly modified tag-KEM-DEM framework where the symmetrically encrypted message is used as the tag for encapsulation. This allows for a weaker data encapsulation mechanism to be used.

Dual Receiver Key Encapsulation

Just as with public-key encryption, there exists a dual receiver version of key encapsulation mechanisms which is intended to encrypt the same symmetric key for two receiving parties at once. The following definition of a dual receiver key encapsulation mechanism is based on [CFZ14]. Note that [CFZ14] presents the definition in the common reference string (CRS) model while we always assume group parameters to be fixed out of scope.

Definition (DR-KEM):
A *dual receiver key encapsulation mechanism (DR-KEM)* is given by a set of three PPT algorithms $(\texttt{gen}, \texttt{enc}, \texttt{dec})$ with

$$\begin{aligned} \texttt{gen}: &\quad 1^\lambda \mapsto (sk, pk) \\ \texttt{enc}: &\quad (pk_1, pk_2) \mapsto (K, C) \\ \texttt{dec}: &\quad (sk_i, pk_1, pk_2, C) \mapsto K, \text{ where } i \in \{1, 2\} \end{aligned}$$

such that the following correctness property holds:

$$K = \texttt{dec}(sk_i, pk_1, pk_2, C)$$

whenever $(sk_1, pk_1), (sk_2, pk_2) \leftarrow \texttt{gen}(1^\lambda)$, $i \in \{1, 2\}$ and

$$(K, C) \leftarrow \texttt{enc}(pk_1, pk_2).$$ ◦

2.1.3 KEM/DEM Security Notions

In this section we look at the definitions of game-based security notions for various types of key encapsulation mechanisms and data encapsulation mechanisms. Again, I limit myself to those security notions whose details become relevant later on in this dissertation. We start with definitions for key encapsulation mechanisms and move on to data encapsulation notions afterwards.

IND-CCA2 for Tag-based Key Encapsulation

As always, the strongest security notion is denoted by IND-CCA2 where the adversary has access to oracles which decrypt anything other than the challenge itself. The following tag-based key encapsulation specific definition was taken from [AGKS05]. Note that in [AGKS05], there is a first oracle phase where the adversary only has pk as prior input. Since the adversary has equal oracle access for the second phase and only gains additional input in between (rather than making any output themselves), the first oracle phase is redundant and I choose to present the notion without it.

Definition (IND-CCA2 for tag-KEMs):
A tag-based key encapsulation mechanism $\Sigma = (\texttt{gen}, \texttt{key}, \texttt{enc}, \texttt{dec})$ with key space $\mathbf{K}$ satisfies *indistinguishability under adaptive chosen ciphertext attack (IND-CCA2)*, if and only if for any PPT adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ the advantage

$$\begin{aligned}\mathsf{Adv}^{\mathrm{CCA2}}_{\mathcal{A},\Sigma}(\lambda) := \Big|\mathbb{P}\Big[b \leftarrow \mathcal{A}_2^{\mathcal{O}^*}(C^*, aux_{\mathcal{A}}) \;\Big|\; &(sk, pk) \leftarrow \texttt{gen}(1^\lambda);\\ &(aux, K_0) \leftarrow \texttt{key}(pk); K_1 \xleftarrow{\$} \mathbf{K}; b \xleftarrow{\$} \{0,1\};\\ &(\tau^*, aux_{\mathcal{A}}) \leftarrow \mathcal{A}_1^{\mathcal{O}}(pk, K_b); C^* \leftarrow \texttt{enc}(aux, \tau^*)\Big] - \tfrac{1}{2}\Big|\end{aligned}$$

is negligible in λ, where $\mathcal{O}$ denotes $\texttt{dec}(sk, \cdot, \cdot)$ and $\mathcal{O}^*(\tau, C)$ returns $\perp$ for $(\tau, C) = (\tau^*, C^*)$ and $\texttt{dec}(sk, \tau, C)$ otherwise. ○

IND-CPA and Soundness for Dual Receiver Key Encapsulation

At the other end of the security spectrum from IND-CCA2 lies IND-CPA security, where no access to decryption oracles is provided at all. The IND-CPA security notion specifically corresponding to the dual receiver key encapsulation mechanism looks as follows:

Definition (IND-CPA for DR-KEMs):
A dual receiver key encapsulation mechanism $\Sigma = (\texttt{gen}, \texttt{enc}, \texttt{dec})$ with key space $\mathbf{K}$ satisfies *indistinguishability under chosen plaintext attack (IND-CPA)* (cf. [CFZ14]), if and only if for any PPT adversary $\mathcal{A}$ the advantage

$$\begin{aligned}\mathsf{Adv}^{\mathrm{CPA}}_{\mathcal{A},\Sigma}(\lambda) := \Big|\mathbb{P}\Big[b \leftarrow \mathcal{A}(pk_1, pk_2, K_b, C^*) \;\Big|\; &(sk_1, pk_1), (sk_2, pk_2) \leftarrow \texttt{gen}(1^\lambda);\\ &(K_0, C^*) \leftarrow \texttt{enc}(pk_1, pk_2); K_1 \xleftarrow{\$} \mathbf{K};\\ &b \xleftarrow{\$} \{0,1\}\Big] - \tfrac{1}{2}\Big|\end{aligned}$$

is negligible in the security parameter λ. ○

As with dual receiver encryption, the true benefits of dual receiver key encapsulation only unfold when additional properties are added. The most commonly used additional property is soundness, which guarantees both decrypting parties that they received the same encapsulated key K instead of two different ones.

Definition (Soundness for DR-KEMs):
A dual receiver key encapsulation mechanism $\Sigma = (\mathtt{gen}, \mathtt{enc}, \mathtt{dec})$ satisfies *soundness* (cf. [CFZ14]), if and only if for any PPT adversary $\mathcal{A}$ the advantage

$$\begin{aligned}\mathsf{Adv}^{\text{sound}}_{\mathcal{A},\Sigma}(\lambda) := \mathbb{P}\Big[&\mathtt{dec}(sk_1, pk_1, pk_2, C) \neq \mathtt{dec}(sk_1, pk_1, pk_2, C) \mid \\ &C \leftarrow \mathcal{A}(sk_1, pk_1, sk_2, pk_2); \\ &(sk_1, pk_1), (sk_2, pk_2) \leftarrow \mathtt{gen}(1^\lambda)\Big]\end{aligned}$$

is negligible in the security parameter λ. ◦

Moving from key encapsulation to data encapsulation security notions we notice that there are two fundamentally different ways they are denoted in prior works. Staying in line with the nomenclature of public-key encryption and key encapsulation security, I follow the more descriptive variant of [HHK10]. Note that with the also commonly used PX-CY notation of [KY06], IND-CPA corresponds to P2-C0, while IND-OT corresponds to P0-C0.

IND-CPA for Data Encapsulation

Session communication—where each symmetric key may be used for many different messages—requires relatively weak data encapsulation security when authenticated channels are used. Let me therefore recap symmetric key IND-CPA security.

Definition (IND-CPA for DEMs):
A data encapsulation mechanism $\Sigma = (\mathtt{DEM.enc}, \mathtt{DEM.dec})$ with key space $\mathbf{K}$ satisfies *indistinguishability under chosen plaintext attack (IND-CPA)*, if and only if for any PPT adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ the advantage

$$\begin{aligned}\mathsf{Adv}^{\text{CPA}}_{\mathcal{A},\Sigma}(\lambda) := \Big|\mathbb{P}\Big[&b \leftarrow \mathcal{A}_2^{\mathcal{O}_{\mathtt{enc}}}(c^*, aux) \mid K \xleftarrow{\$} \mathbf{K}; \\ &(m_0, m_1, aux) \leftarrow \mathcal{A}_1^{\mathcal{O}_{\mathtt{enc}}}(1^\lambda); \\ &b \xleftarrow{\$} \{0,1\}; c^* \leftarrow \mathtt{enc}(K, m_b)\Big] - \tfrac{1}{2}\Big|\end{aligned}$$

is negligible in λ, where the encryption oracle is defined as

$$\mathcal{O}_{\mathtt{enc}}(\cdot) := \mathtt{DEM.enc}(K, \cdot).$$ ◦

IND-OT for Data Encapsulation

For single message communication—where each symmetric key is used only once—the even weaker security notion of IND-OT suffices for the data encapsulation mechanism. It does not even provide the adversary with an encryption oracle—as IND-CPA does—and is specifically tailored to the situation of single-use keys.

Definition (IND-OT for DEMs):
A data encapsulation mechanism $\Sigma = (\texttt{DEM.enc}, \texttt{DEM.dec})$ with key space $\mathbf{K}$ satisfies *indistinguishability under one-time attack (IND-OT)*, if and only if for any PPT adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ the following advantage is negligible in λ:

$$\begin{aligned}\mathsf{Adv}^{\mathrm{OT}}_{\mathcal{A},\Sigma}(\lambda) := \Big|\mathbb{P}\Big[b \leftarrow \mathcal{A}_2(c^*, aux) \,\Big|\, & K \overset{\$}{\leftarrow} \mathbf{K};\\ & (m_0, m_1, aux) \leftarrow \mathcal{A}_1(1^\lambda); b \overset{\$}{\leftarrow} \{0,1\};\\ & c^* \leftarrow \texttt{enc}(K, m_b)\Big] - \tfrac{1}{2}\Big|.\end{aligned}$$ ◦

Now that we are familiar with all necessary game-based definitions, let us switch to the parallel world of simulation-based and in particular UC security.

2.2 Simulation-based Security and Universal Composability

In this section I briefly recap the real/ideal-paradigm as well as the universal composability (UC) framework. Readers who are already intimately familiar with these topics might want to skip to Section 2.2.2 or even to Chapter 3.

Traditional game-based security notions are usually centred around attacks, demanding that a very specific attack—like distinguishing which of two chosen messages is contained in a ciphertext—must not be successful. For protocols to be actually real-world secure via this method, however, every conceivable attack has to be foreseen or at least accidentally covered by the chosen game-based properties.

Simulation-based security (cf. [Can00]), on the other hand, works fundamentally differently. Security requirements are captured by describing the task in the form of an ideal functionality $\mathcal{F}$ which acts as a kind of trusted third party: The functionality $\mathcal{F}$ is handed the inputs of all participating parties, conducts the desired protocol (e.g. some function calculation on the inputs) and hands the various outputs back to the individual protocol participants.

In addition to ideal functionalities we need to understand the concept of computational indistinguishability before we can define security according to the real/ideal-paradigm.

Definition (Computational Indistinguishability):
Let $X = (X_\lambda)_{\lambda\in\mathbb{N}}$ and $Y = (Y_\lambda)_{\lambda\in\mathbb{N}}$ be two sequences of random variables. Then X and Y are *computationally indistinguishable*, denoted by $X \sim Y$, if and only if for all PPT algorithms $\mathcal{Z}$ the advantage

$$\mathsf{Adv}^{\mathrm{indist}}_{X,Y,\mathcal{Z}}(\lambda) = \Big|\mathbb{P}\Big[\mathcal{Z}(1^\lambda, x \leftarrow X_\lambda) = 1\Big] - \mathbb{P}\Big[\mathcal{Z}(1^\lambda, x \leftarrow Y_\lambda) = 1\Big]\Big|$$

is negligible. ◦

Note that any protocol execution can be viewed as a sequence of random variables in the security parameter λ.

The real/ideal-paradigm now defines security in relation to the ideal functionality which describes the desired task: If the execution of a real protocol π is at least as secure as the execution with the ideal functionality $\mathcal{F}$, the protocol is said to securely realise the functionality. For a protocol π to be "at least as secure" as functionality $\mathcal{F}$ means that for every real adversary

$\mathcal{A}$ attacking the real protocol π there is a simulator (an equally successful ideal adversary) $\mathcal{S}$ attacking the ideal functionality $\mathcal{F}$ such that the executions of π with $\mathcal{A}$ and of $\mathcal{F}$ with $\mathcal{S}$ are computationally indistinguishable.

While this core concept of comparison to an ideal functionality remains the same, simulation security in the UC framework (cf. [Can01]) goes one step further. The computational distinguisher $\mathcal{Z}$, called environment, does not only get the transcript

$$x \leftarrow X_\lambda \qquad \text{or} \qquad x \leftarrow Y_\lambda$$

of an execution to try to distinguish both cases. Instead the environment is allowed to adaptively control the execution by directing protocol parties. Hence the UC framework gives a stronger security notion because the need to simulate one step at a time in response to adaptive inputs from the environment rules out standard techniques like rewinding. As a result, protocols which UC-realise a functionality remain secure under arbitrary and concurrent composition, while protocols which just securely realise a functionality in the classic sense provide stand-alone security only.

In the following Section 2.2.1 I will explain various aspects of the UC framework which are helpful to comprehend definitions and security proofs in this dissertation. Due to the complexity of the full UC model I will necessarily omit most technical details which an interested reader can find in [Can01]. For all the different components, lemmata end theorems I will refrain from lengthy formal definitions for the benefit of intuitive explanations. In some cases I will also stray from original denominations and use other commonly preferred terms instead, because they are more descriptive in my opinion. I will, e.g. only talk about "adversaries" instead of "control functions".

At the very end of this section, in Section 2.2.2, I present and explain the concrete ideal functionalities we encounter in later chapters.

2.2.1 The Universal Composability Framework

Computations in the UC framework are based on Turing machines. To facilitate communication and joint computations, classic Turing machines are enhanced with several additional tapes to create *interactive* Turing machines. Note that [Can01] distinguishes between interactive Turing machines and interactive Turing machine instances (ITIs). Intuitively, an interactive Turing machine is an intangible, static object and defined by its program alone, while an ITI is the concrete run-time object associated with this program and as such has a well-defined current internal state. This distinction will not be relevant to any observations I make throughout this dissertation, which is why I will restrict myself to talking about ITIs—although in some rare cases, using the phrase interactive Turing machine to denote the intangible object might have been more precise.

Interactive Turing Machines. The differences between classic Turing machines and *interactive Turing machine instances (ITIs)* as employed in the UC framework include:

- An *identity tape* which contains the program code and ID of the ITI.
- An *incoming message tape*. This represents the ITI's inbox for messages from other ITIs.
- An *outgoing message tape*. This represents the ITI's outbox for messages which should be sent to other ITIs.

- An *input tape*. This is similar to the incoming message tape, but contains messages which are passed locally instead of over a network. Think for instance about messages passed between different processes on the same computer.
- An *output tape*. This is the analogon of the outgoing message tape with respect to local messages.
- A *random tape* to facilitate probabilistic computations.
- A *temporary halting state*. This state is reached every time the ITI writes anything on one of its outgoing tapes and indicates that the ITI has passed activation to another ITI. Correspondingly, receiving a message on one of its incoming tapes reactivates an ITI from the temporary halting state, lets it proceed to read the next incoming message and process the message according to its program.

To be able to execute protocols in the UC framework, different types of ITIs are needed.

"Normal" ITIs. We can think of "normal" ITIs as processes running on a physical computer. Processes running on the same machine—which are modelled by ITIs with the same party ID—can communicate via input/output tapes. This communication is instantaneous, secret and reliable. Processes running on different machines on the other hand—i.e. ITIs with different party IDs—need to communicate via the network which is modelled by the incoming and outgoing message tapes. For this dissertation, it is mostly sufficient to think of each party as running only one process. We will therefore usually identify a protocol party with one single "normal" ITI.

Environment $\mathcal{Z}$. The first special type of ITI we need to discuss is the *environment* $\mathcal{Z}$. As the name suggests, it is meant to model the external environment in which a computation is executed. This includes, e.g. concurrently running processes as well as user inputs/outputs to and from the system. As such, the environment only communicates via input and output tapes with other ITIs where communication is instantaneous, secret and reliable. The environment is not allowed to participate in the network itself. However, since it dynamically invokes other ITIs, directs their behaviour via inputs and receives their protocol outputs, the environment acts like the overall director of any execution. For the later concepts of protocol emulation and UC realisation, the environment furthermore takes the role of an interactive distinguisher.

Adversary $\mathcal{A}$. A second special type of ITI is the *adversary* $\mathcal{A}$. One purpose of the adversary is to control the message network. A message which an ITI writes on its outgoing message tape is not directly copied to the incoming message tape of the specified receiving ITI. Instead, the message with its intended recipient and sender information is handed to the adversary which can arbitrarily change, reroute, delay or block the message rather than just delivering it to the intended receiver. This corresponds to the common Dolev-Yao threat model [DY83].

The second adversarial power of $\mathcal{A}$ is to corrupt any "normal" ITI of its choice. Note that the adversary can not corrupt the environment $\mathcal{Z}$. Whenever we talk about corruption in this dissertation we mean byzantine—also called malicious—corruption. In case of corruption, the content of all the ITI's tapes is handed over to the adversary which subsequently completely impersonates the corrupted ITI. This means from the point of corruption onward, any inputs to the corrupted ITI are just passed to the adversary and the adversary is allowed to provide outputs to other ITIs in the name of the corrupted one.

System of ITIs. A *system of ITIs* $\Psi = (\mathcal{Z}, \mathcal{A})$ formally only consists of an environment $\mathcal{Z}$ and an adversary $\mathcal{A}$. Any "normal" ITIs participating in the system will be dynamically invoked by the environment during an execution of the system—at least for our simplified version where each party is represented by only one ITI. In general, "normal" ITIs also have the power to invoke subsidiary ITIs belonging to the same party.

The environment acts as initial ITI of the system, i.e. an execution of the system $\Psi = (\mathcal{Z}, \mathcal{A})$ on input x is defined as activating environment $\mathcal{Z}$ with x on its input tape and continuing the execution until $\mathcal{Z}$ permanently halts. In this case we say the system has halted with the output of the system being the output of $\mathcal{Z}$.

Remember that each ITI only remains active until it writes on one of its outgoing tapes. At this point, activation is passed to another ITI: Either the recipient ITI in case of outputs or the adversary $\mathcal{A}$ for outgoing messages. At each point in the execution of the system, exactly one ITI in the system will be active. If any ITI halts without writing on one of its outgoing tapes, activation is passed to the environment $\mathcal{Z}$.

Resource-bounded Computation. As with most cryptographic security analyses, the UC model is not concerned with adversaries or protocol parties who have unbounded computational power. One standard way to limit the computational power of individual ITIs is to only allow probabilistic polynomial time (PPT) ITIs, i.e. those for which a polynomial p exists that bounds their overall runtime with respect to some security parameter λ. Executing a whole system with a dynamically evolving number of participants, however, calls for a more intricate definition of bounded resources to ensure that the overall runtime of the system remains bounded while each component only requires local knowledge to restrict their own runtime.

Canetti solves this problem via so called "imports" which are added to every message. Intuitively, the initial import to the initial ITI, i.e. the environment, bounds the overall runtime of the whole system. Any import given to an ITI can be seen as a kind of allowance for computational runtime: The ITI can either use this allowance itself or (partly) hand it on to other ITIs by providing it with a message that contains the respective import. While we will not concern ourselves with the formal definition here, the important parts to note are that the overall runtime of a protocol execution as well as the runtime of any individual participating ITI is subject to a polynomial bound $\mathsf{p}(\lambda)$ in a universal security parameter λ. I.e. all ITIs as well as the overall protocol executions are in particular PPT. We will mainly use this fact with respect to the environment $\mathcal{Z}$ when conducting security proofs.

Protocol Executions. Let π be a PPT protocol, i.e. a piece of code such that an ITI running program code π is PPT. An execution of protocol π is denoted by

$$\mathsf{EXEC}(\pi, \mathcal{A})_{\mathcal{Z}}$$

and defined as the execution of system $\Psi = (\mathcal{Z}, \mathcal{A})$ on input 1^λ with the additional provision that any "normal" ITI invoked by $\mathcal{Z}$ runs protocol π.

An overview of the system setup for protocol executions can be found in Figure 6. Note that message communication over the network is depicted as dashed lines, while input/output communication is indicated by continuous lines. Uncorrupted "normal" ITIs running protocol π are presented as P^π in green, while corrupted ones are depicted as P in orange and incorporated into the adversary.

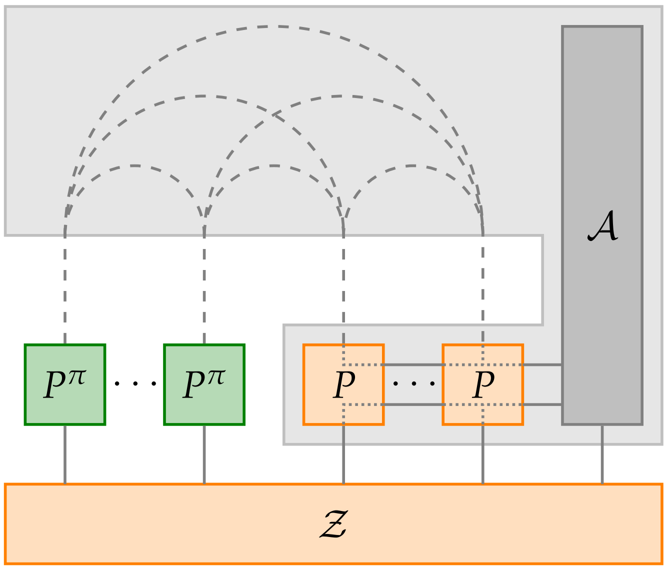


Figure 6: Overview of Protocol Execution

Ideal Functionalities and Dummy Parties. The third special kind of ITI we encounter in the UC framework are *ideal functionalities*, which are essential to formalise the ideal world which real protocols can be compared to. Ideal functionalities act as an imaginary trusted third party. Instead of conducting a joint protocol over the network, participating protocol parties hand all their inputs to the ideal functionality which performs any necessary computations and hands respective outputs back to the participants. All communication with the ideal functionality is handled via input and output tapes, so that it is neither disclosed to the adversary $\mathcal{A}$ nor subject to its manipulations.

Because ideal functionalities incorporate all the actual desired functionality of the situation, they only ever interact with *dummy parties*. Dummy parties do nothing other than handing through inputs and outputs between environment and ideal functionality. They do not run any actual code themselves and do not use the network to communicate. Note that dummy parties can still be corrupted by the adversary $\mathcal{A}$, though, in which case (as with corruption of any other party) the adversary may completely override their normal behaviour.

The intention behind ideal functionalities, also sometimes called ideal protocols, is to provide an optimal solution to compare actual protocols with. Note that "optimal" in this case will often only mean "realistically optimal" instead of "perfect utopia". For instance, functionalities for secure communication will still disclose to the adversary that a message was sent (just not its content) and wait for the adversary's OK to deliver it—because no cryptographic protocol can realistically preclude the adversary just cutting the network line. This information leakage and definition of adversarial power is handled via input/output tapes between the ideal functionality and the adversary, which makes it much more explicit than many other types of security notions.

The execution of an ideal functionality $\mathcal{F}$, denoted by

$$\mathsf{EXEC}(\mathcal{F},\mathcal{A})_{\mathcal{Z}},$$

is defined as the execution of the system $\Psi = (\mathcal{Z}, \mathcal{A})$ with the additional provision that any "normal" ITI invoked by environment $\mathcal{Z}$ is a dummy party which interacts with the same ITI running the code of $\mathcal{F}$. An overview of the system setup for ideal functionality executions can be found in Figure 7. Again, uncorrupted dummy parties are presented as P in green, while corrupted ones are depicted in orange and incorporated into the adversary.

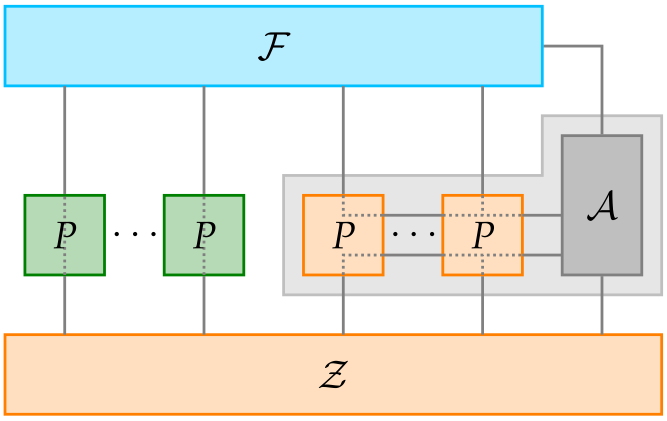


Figure 7: Overview of Ideal Functionality Execution

Security in the UC Framework

With all the above building blocks, we can finally talk about how security is defined in the UC framework. Just as with stand-alone simulation-based security, UC security is based on comparing protocols and ideal functionalities via computational indistinguishability. The environment $\mathcal{Z}$ acts as distinguisher in this setup while the simulator $\mathcal{S}$ is formally just an adversary ITI.

Definition (UC Emulation and Realization):
Let π_1 and π_2 be PPT protocols. We say that π_1 *UC-emulates* π_2 if and only if for any PPT adversary $\mathcal{A}$ there exists a PPT simulator $\mathcal{S}$ such that executions of protocol π_1 conducted with $\mathcal{A}$ and protocol π_2 conducted with $\mathcal{S}$ are computationally indistinguishable for any environment $\mathcal{Z}$. More formally:

$$\mathsf{EXEC}(\pi_1, \mathcal{A}) \sim \mathsf{EXEC}(\pi_2, \mathcal{S}),$$

i.e. for any PPT environment $\mathcal{Z}$ (acting as distinguisher) the advantage

$$\mathsf{Adv}^{\text{indist}}_{\mathsf{EXEC}(\pi_1,\mathcal{A}),\mathsf{EXEC}(\pi_2,\mathcal{S}),\mathcal{Z}}(\lambda)$$

is negligible in the security parameter λ.

If protocol π_2 is given by an ideal functionality $\mathcal{F}$, we usually say that π_1 *UC-realises* ideal functionality $\mathcal{F}$ and also write

$$\pi_1 \geqslant_{\text{UC}} \mathcal{F}. \qquad \circ$$

This definition nicely captures the intuition of UC security that a protocol π_1 is stronger/more secure than π_2 if the attacks of any adversary $\mathcal{A}$ can equally successfully be conducted (simulated) in an interaction with π_2. In [Can01], however, Canetti showed that restricting ourselves to a dummy adversary, which only acts as a mindless puppet of the environment $\mathcal{Z}$ actually yields an equivalent definition.

Dummy Adversary $\mathcal{D}$. The *dummy adversary* is a special case of the adversary $\mathcal{A}$, usually denoted as $\mathcal{D}$. The dummy adversary has no true program code of its own, other than to act however the environment $\mathcal{Z}$ directs. The environment $\mathcal{Z}$ in turn makes all adversarial decisions. For this purpose the dummy adversary instantly forwards everything it learns on its incoming tapes to the environment—unless, of course, the input already originated with the environment—in particular including all messages from the outgoing message tapes of normal ITIs which are given to the adversary as the director of the network. Any action like delivering, blocking or changing those network messages is then taken if and only if (and when) the environment asks the dummy adversary to do so. Following the environment's instructions furthermore includes corrupting other ITIs only on demand. Subsequently the dummy adversary forwards all tape contents, inputs and messages it learns through the corruption and impersonates the corrupted ITI in exactly the way the environment specifies.

Lemma 1 (Completeness of the Dummy Adversary):
Let π_1 and π_2 be PPT protocols. Then π_1 UC-emulates π_2 if and only if it UC-emulates π_2 with respect to the dummy adversary $\mathcal{D}$, i.e. if there exists a PPT simulator $\mathcal{S}$ such that

$$\mathsf{EXEC}(\pi_1, \mathcal{D}) \sim \mathsf{EXEC}(\pi_2, \mathcal{S})$$

are computationally indistinguishable for any PPT environment $\mathcal{Z}$.

For the proof of Lemma 1, see [Can01]. While it is harder to grasp the intuition from this equivalent definition of UC emulation, it is easier to work with in security proofs, as it contains one less quantifier to take care of.

Hybrid Executions. Another common practice to simplify security proofs is to introduce hybrid executions

$$\mathsf{EXEC}(\pi_1, \mathcal{D}) =: H_0, H_1 \ldots, H_n := \mathsf{EXEC}(\pi_2, \mathcal{S})$$

which provide a more gradual change from π_1 to π_2. Using the obvious finite transitivity of computational indistinguishability, the proof that π_1 UC-emulates π_2 can be exchanged for the set of individual proofs that each hybrid H_{i-1} is computationally indistinguishable from the next hybrid H_i for $i \in \{1, \ldots, n\}$. In many cases, if the hybrid protocols are appropriately chosen, those individual proofs are easier to conduct than proving UC emulation in one step.

For most UC security reductions, however, a stronger transitivity version for polynomially many hybrids is needed. Many different versions of such polynomial hybrid arguments exist. In this dissertation I will only use the hybrid argument with a universal distinguishing bound. The following lemma is a simplified version of this type of hybrid argument as formulated in [MF21], where the respective proof can also be found:

Lemma 2 (Hybrid Argument):
Let $\mathsf{p} : \mathbb{N} \to \mathbb{N}$ be a polynomial and $(H_i)_{i \in \mathbb{N}_0}$ be a sequence of random variables. If for each distinguisher $\mathcal{Z}$ there exists a function ε such that there exists an integer $\lambda_0 \in \mathbb{N}$ such that for all $\lambda \geqslant \lambda_0$ and all $i < \mathsf{p}(\lambda)$ we have

$$\mathsf{Adv}^{\mathit{indist}}_{H_{i-1}, H_i, \mathcal{Z}}(\lambda) \leqslant \varepsilon(\lambda),$$

then it holds that

$$\mathsf{Adv}^{\mathit{indist}}_{H_0, H_{\mathsf{p}(\lambda)}, \mathcal{Z}}(\lambda) \leqslant \mathsf{p}(\lambda) \cdot \varepsilon(\lambda).$$

for all $\lambda \geqslant \lambda_0$. In particular it follows that if ε is negligible, H_0 and $H_{\mathsf{p}(\lambda)}$ are computationally indistinguishable.

$\mathcal{F}$-Hybrid Models. So far, I pretended that there are only either "real" protocols conducted by "real" parties, i.e. code running "normal" ITIs, or ideal functionalities which conduct the function of the respective protocol while interacting with dummy parties. One of the strengths of the UC framework, however, is that these worlds can mix and thus allow for modular analysis of security: Protocols π can be conducted in $\mathcal{F}$-hybrid models, which we sometimes denote as $\pi^{\mathcal{F}}$. This means that parties conducting a protocol π furthermore have access to an ideal functionality $\mathcal{F}$ while doing so.

To give the concrete example that is at the core research question of this dissertation: A protocol, aiming for secure communication, can use the ideal functionality for authenticated channels if it is defined in the $\mathcal{F}_{\text{AUTH}}$-hybrid model.

Formally, if a protocol π is defined in the $\mathcal{F}$-hybrid model, each "normal" ITI running the code of π invokes their own dummy party interacting with the functionality $\mathcal{F}$. The dummy party will then not forward inputs and outputs between environment and ideal functionality, but between its parent ITI running π and the ideal functionality, instead.

An overview of the system setup for protocol executions in a hybrid model can be found in Figure 8.

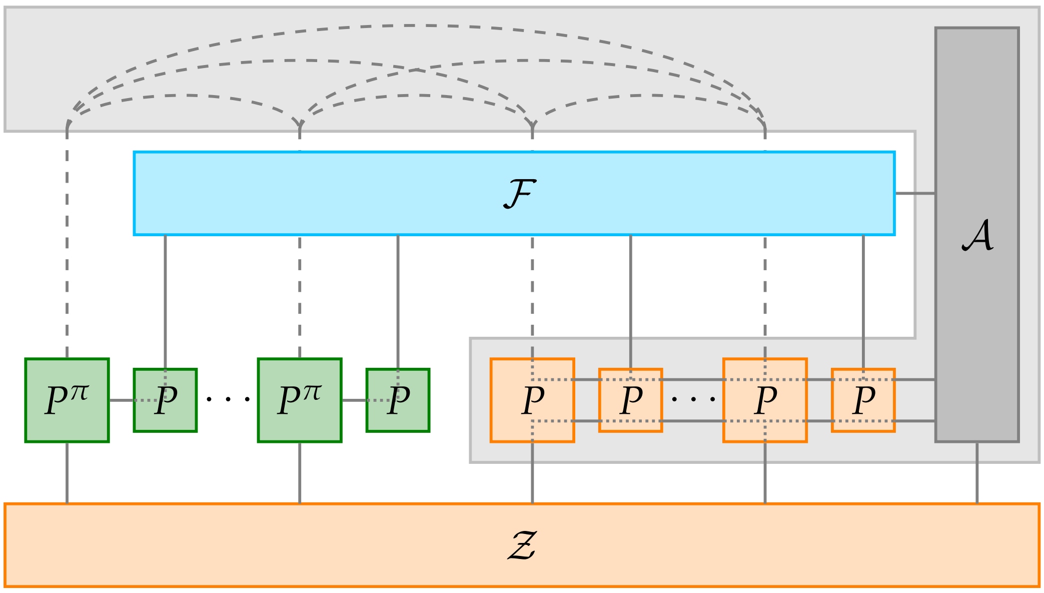


Figure 8: Overview of Protocol Execution in a Hybrid Model

In the later chapters of this dissertation, we will omit this additional dummy and the distinction between the two different ITIs (dummy and non-dummy) for the sake of readability.

Universal Composability. The true value of modular protocols, i.e. using hybrid models, only unfolds in conjunction with the UC theorem. It asserts that protocols can not only be defined in a modular way, but that modular security analysis is also sufficient.

Theorem 1 (The UC Theorem):
Let π_1, π_2 and π_3 be PPT protocols such that

$$\pi_1 \geqslant_{UC} \pi_2$$

holds. Then also

$$\pi_3^{\pi_2 \rightarrow \pi_1} \geqslant_{UC} \pi_3,$$

where $\pi_2 \to \pi_1$ intuitively means that every call of π_3 to π_2 is replaced by an equivalent call to π_1 instead.

In particular, using ideal functionalities $\mathcal{F}_1$ and $\mathcal{F}_2$, if we have a protocol $\pi_1^{\mathcal{F}_2}$ UC-realising $\mathcal{F}_1$ in the $\mathcal{F}_2$-hybrid model and a protocol π_2 UC-realising $\mathcal{F}_2$, then the protocol $\pi_1^{\pi_2}$ will also UC-realise functionality $\mathcal{F}_1$:

$$\left(\pi_1^{\mathcal{F}_2} \geqslant_{UC} \mathcal{F}_1\right) \wedge \left(\pi_2 \geqslant_{UC} \mathcal{F}_2\right) \implies \pi_1^{\pi_2} \geqslant_{UC} \mathcal{F}_1.$$

With this composition theorem, if any larger protocol uses cryptographic building blocks $\mathcal{F}_1, \dots, \mathcal{F}_n$ and the combining protocol $\pi^{\mathcal{F}_1,\dots,\mathcal{F}_n}$ is proven secure in the $(\mathcal{F}_1, \dots, \mathcal{F}_n)$-hybrid model, it remains secure with any instantiations of the building blocks $\mathcal{F}_1, \dots, \mathcal{F}_n$.

To give a more concrete example: For this modular security analysis it is not necessary to, e.g. specify which concrete signature scheme is used within a larger communication protocol and then have a new security proof whenever the protocol is used with a different scheme. Instead, the larger protocol can be proven secure in the $\mathcal{F}_{\text{Sig}}$-hybrid model and is instantly known to be secure with any signature scheme that UC-realises $\mathcal{F}_{\text{Sig}}$.

Throughout this dissertation I use this modular security by working in the $\mathcal{F}_{\text{AUTH}}$-hybrid model to realise secure communication without having to consider concrete realisations of authenticated channels.

On Adaptive and Static Corruption. As with most security frameworks, there are various forms of party corruptions formalised in the UC model. The difference between the two most common corruption modes—adaptive and static corruption—lies in the timing of corruption: While *adaptive corruption* allows the adversary to corrupt ITIs at any point in time, *static corruption* requires the adversary to either corrupt an ITI at once, when the ITI is first activated, or not at all.

A common misconception is that adaptive corruption yields stronger security for honest parties. While it is true that the adaptive corruption mode results in a formally stronger security notion than static corruption, the difference is strongly related to deniability and impacts corrupted parties rather than honest ones.

We can see this by considering two different protocol executions: In the first one, several parties are adaptively corrupted throughout the execution. In the second execution, the same parties are corrupted but this time statically, i.e. from their very first activation onward. Instead of possibly acting maliciously from the start, the adversary lets those statically corrupted parties follow the protocol honestly up to the point where they would have been adaptively corrupted in the first execution.

From the perspective of any honest party both executions are completely indistinguishable. Whether a party that honestly follows the protocol is, in fact, already formally corrupted at this point does not make any difference for them. The difference we observe for both executions instead impacts the corrupted parties. If corruption happens adaptively, then the simulator $\mathcal{S}$ in the security proof needs to come up with internal secrets for the now corrupted party which are consistent with all past inputs/outputs as well as past simulations (which were made without knowledge of these inputs/outputs). This is usually only possible if the cryptographic building blocks offer some type of deniability. E.g. if the simulator used some arbitrary (and computationally indistinguishable) ciphertext c in the simulation back when a party was honest and upon corruption of the party learns that this ciphertext needs to have contained message m, it needs a mechanism to efficiently compute internal key material and randomness which encrypts the message m to c.

As we only concern ourselves with security for honest parties and do not aim for any forms of deniability, we will exclusively consider security under static malicious corruption throughout this dissertation.

Session IDs. An ITI's ID on the identity tape is usually split into two parts: A party ID *pid* and a session ID *sid*. For the most part, we will not explicitly handle session IDs and instead always consider individual sessions where every participating ITI implicitly shares the same *sid*. The one exception is the ideal functionality $\mathcal{F}_{\text{AUTH}}$. Since this functionality handles only one message per session ID, our protocols will usually employ many different instances of this functionality, each identified by their own session ID sid_{AUTH}. Formally—as mentioned in the explanation on $\mathcal{F}$-hybrid models—each party interacting with $\mathcal{F}_{\text{AUTH}}$ invokes a dummy party with the same party ID *pid* but session ID sid_{AUTH} to interact with the respective instance of $\mathcal{F}_{\text{AUTH}}$ with session ID sid_{AUTH}. Because I feel that this technicality would impair readability more than it would enhance formal correctness, I will always omit the dummy parties as well as session IDs in these cases. Whenever a "normal" ITI provides input to $\mathcal{F}_{\text{AUTH}}$, it does so by calling a new instance with freshly drawn random sid_{AUTH}. Whenever a "normal" ITI receives input from an instance of $\mathcal{F}_{\text{AUTH}}$, the specific session ID has no relevance.

2.2.2 Ideal Functionalities

We mainly encounter ideal functionalities in the various UC security proofs throughout this dissertation. For the most part this will only include the authenticated channel functionality $\mathcal{F}_{\text{AUTH}}$ as well as secure communication functionalities $\mathcal{F}_{\text{M-SMT}}$ and $\mathcal{F}_{\text{M-SC}}$. In Chapter 5 we also employ the key registration functionality $\mathcal{F}_{\text{KRK}}$ for any generic transformation based on dual receiver schemes. Although we briefly encounter functionalities $\mathcal{F}_{\text{KEM}}$, $\mathcal{F}_{\text{KEM-DEM}}$, $\mathcal{F}_{\text{SIG}}$, $\mathcal{F}_{\text{CA}}$ and $\mathcal{F}_{\text{CERT}}$ in Section 4.1 we do not require any detailed knowledge about them. Interested readers can find the respective definitions in [NMO06] and [Can04].

There are several implicit writing conventions for ideal functionalities defined in [Can01]. One situation we encounter in many of the following definitions is that the functionality will inform the adversary about some process—like a message that is to be sent—and wait for the adversary's OK before it proceeds. This situation is usually covered by the implicit writing convention that those requests to the adversary carry a unique ID which lets the ideal functionality map responses to requests, even if their content might be ambiguous (e.g. when the same message is sent multiple times) or the adversary might answer in a different order. Instead of applying this convention, I will explicitly use message IDs *mid* for this purpose. Another writing convention I employ throughout all ideal functionality definitions is that whatever type of input is not explicitly covered by the behaviour definition gets completely ignored.

Since there have been conflicting definitions for all of the mentioned functionalities, I make the effort to explicitly recapitulate detailed formal descriptions of the versions we work with later on in this dissertation. The definitions are (sometimes loosely) based on [Can01], [CK02], [CKN03] and [BCNP04] and simplified to be used with *static corruption* only.

The Ideal Functionality $\mathcal{F}_{\text{AUTH}}$

The ideal functionality for authenticated channels is rather simple. It takes a receiver R and message m from some sending party S, reports this in full detail to the adversary asking for permission to deliver the message and—if permission is given—outputs message, sender and receiver information to the receiving party R. Note in particular, that $\mathcal{F}_{\text{AUTH}}$ only deals with the transmission of one single message. Multiple messages require multiple instances (each with a separate session ID *sid*) of the functionality. This definition is simplified to be used with *static corruption* only and is based on [Can01]. More formally:

$\mathcal{F}_{\textbf{AUTH}}$

Provides:
Single-receiver single-message single-sender authenticated message transfer with message space $\mathbf{M}$.

Behaviour:

- Upon first invocation with input $(\texttt{send}, R, m)$ with $m \in \mathbf{M}$ from some party S, provide input $(\texttt{send}, S, R, m)$ to the adversary $\mathcal{A}$.
- Upon receiving input $(\texttt{send ok})$ from adversary $\mathcal{A}$, output $(\texttt{sent}, S, R, m)$ to R.
- Ignore all further inputs.

The Ideal Functionality $\mathcal{F}_{\textsf{M-SMT}}$

For secure message transfer there have been a lot of different definitions over the years. Most of them (cp. the seven different versions in the history of [Can01]) deal—just like $\mathcal{F}_{\text{AUTH}}$—with transmission of only a single message. But it is easily seen that this can be achieved from authenticated channels and IND-CPA secure encryption with new credentials for the encryption of every new message [Can01].

The interesting problem, however, is using the same credentials over and over for different messages, which was firstly captured by a functionality for secure transmission of multiple messages in [CKN03]. This definition, however, is based on an obsolete version of [Can01]. Furthermore it deals with multiple messages and senders, but still only allows for one receiver per instance of the functionality.

To provide some ease of notation with the more holistic public key infrastructure approach which the new notion of IND-SB-CPA security suggests, I will give a definition of $\mathcal{F}_{\text{M-SMT}}$ which deals with multiple receivers, multiple senders *and* multiple messages. The equally feasible alternative would have been to work with a multi-session extension (cp. [CR02]) of a functionality $\mathcal{F}_{\text{SMT}}$ which only transmits a single message. As mentioned above, I also choose to handle the commonly present requests for permissions from the adversary (e.g. to transmit something over a channel which the adversary can block) explicitly via message IDs $mid \in \mathbf{MID}$.

Furthermore I again give a simplified version restricted to *static corruption* only.

$\mathcal{F}_{\textbf{M-SMT}}$

Provides:
Multi-receiver multi-message multi-sender secure message transfer with message space $\mathbf{M}$ and polynomially many parties $P \in \mathbf{P}$.

State:
Function $\mathrm{p}_{\mathrm{msg}} : \mathbf{MID} \to \mathbf{M} \times \mathbf{P}^2$ of pending messages.

Behaviour:

- Upon receiving $(\texttt{send}, R, m)$ with $m \in \mathbf{M}$ and $R \in \mathbf{P}$ from some party $S \in \mathbf{P}$, draw fresh mid, send $(\texttt{send}, mid, S, R)$ to the adversary $\mathcal{A}$ and append $mid \mapsto (m, S, R)$ to $\mathrm{p}_{\mathrm{msg}}$.
- Upon receiving $(\texttt{send ok}, mid)$ from the adversary, look up $(m, S, R) := \mathrm{p}_{\mathrm{msg}}(mid)$. If it exists, output $(\texttt{sent}, S, m)$ to R.

Note that the effects of restricting parties to a polynomial set $\mathbf{P}$ are two-fold:

Firstly, the number of participating parties is restricted to $|\mathbf{P}|$. The facts that $|\mathbf{P}|$ is allowed to be any size which remains polynomial in the security parameter λ and that any UC execution runs in polynomial time and can therefore only ever include a number of parties which is polynomial in λ, might at first glance look like no size restriction exists at all. Note, however, that the set $\mathbf{P}$ is a fixed parameter of any instance of the ideal functionality $\mathcal{F}_{\text{M-SMT}}$ and for any $\mathbf{P}$ there will be a PPT environment $\mathcal{Z}$ which invokes at least $|\mathbf{P}| + 1$ parties and therefore be subject to this restriction. The deciding factor here is that with the above definition, $|\mathbf{P}|$ is determined before we quantify over all PPT environments and can subsequently not be chosen large enough to not restrict any environment. While prior efforts to realise $\mathcal{F}_{\text{SMT}}$ from $\mathcal{F}_{\text{AUTH}}$ and some form of encryption were subject to related restrictions, this was just because they were initially conducted at a time when it was common practice to have a functionality contain a finite set of protocol parties $P_1, \ldots, P_n$ as a parameter. Ideal functionalities in the UC framework have since been refined and the proofs in question work equally well without any a priori restriction on the number of protocol parties. While it should certainly be one of our future goals to eliminate this restriction on $|\mathbf{P}|$, it allows me to present some optimality result instead of only a realisation proof. The polynomial restriction is only necessary to make sure that the challenge sender present in the game-based security definition has non-negligible probability to exist in the UC experiment as well. Hence I feel it constitutes an unfortunately necessary but small and worthwhile drawback for now.

Secondly, the actual party IDs are restricted to elements $P \in \mathbf{P}$. This is a restriction which I choose to make explicit, but which was actually implicitly present with all prior efforts to facilitate secure message transfer from given authenticated channels (cp. Section 3.1): In using IND-CCA2 or IND-RCCA secure classic public key encryption, where the respective party ID is concatenated to the message, IDs are implicitly restricted by the message space $\mathbf{M}$ of the encryption scheme in question. For the later efforts employing tag-based encryption, where party IDs are used as tags, they are obviously restricted by the given tag space. Furthermore, the effects of this restriction are easily mitigated because for any real protocol, some other identifying information (like public keys) will be used which needs to be mapped to formal UC party IDs anyway. Such a translation from identifying information to formal IDs does not pose a problem as all of the information is public.

The Ideal Functionality $\mathcal{F}_{\textsf{M-SC}}$

A second type of ideal secure communication are secure channels. The difference to secure message transfer is that communication is organised in sessions which are explicitly established by both communication partners and can also explicitly expire. This type of definition is more natural to work with when we look at session communication in the KEM-DEM framework. The classic definition of $\mathcal{F}_{\mathrm{SC}}$ for single sessions can be found in [CK02].

Analogous to $\mathcal{F}_{\text{M-SMT}}$, we instead use a multi-session version $\mathcal{F}_{\text{M-SC}}$, where some abort possibilities of the adversary (implicitly present in $\mathcal{F}_{\mathrm{SC}}$) are made explicit as well. Again, this ideal functionality is stated for *static corruption* only.

$\mathcal{F}_{\textbf{M-SC}}$

Provides:
Multiple secure two-party communication sessions with message space $\mathbf{M}$ and polynomially many parties $P \in \mathbf{P}$.

State:

- Almost Boolean function $\mathtt{f}_{\mathrm{act}}\colon \{\{A,B\} \mid A,B \in \mathbf{P}\} \to \{\mathsf{true},\mathsf{false},\mathsf{init}\}$ indicating active sessions. Initialised to $\mathtt{f}_{\mathrm{act}} \equiv \mathsf{false}$.
- Function $\mathrm{p}_{\mathrm{msg}} : \mathbf{MID} \to \mathbf{M} \times \mathbf{P}^2$ of pending messages.

Behaviour:

- Upon receiving $(\mathtt{init}, B)$ from some party A, send $(\mathtt{inited}, A, B)$ to the adversary $\mathcal{A}$ and set $\mathtt{f}_{\mathrm{act}}(\{A,B\}) := \mathsf{init}$ upon the adversary's $\mathtt{ok}$.
- Upon receiving $(\mathtt{establish}, A)$ from party B, check $\mathtt{f}_{\mathrm{act}}(\{A,B\}) = \mathsf{init}$, send $(\mathtt{established}, A, B)$ to $\mathcal{A}$ and set $\mathtt{f}_{\mathrm{act}}(\{A,B\}) := \mathsf{true}$ upon the adversary's $\mathtt{ok}$.
- Upon receiving $(\mathtt{send}, R, m)$ from some party S, check $\mathtt{f}_{\mathrm{act}}(\{S,R\}) = \mathsf{true}$, draw fresh mid, send $(\mathtt{send}, mid, S, R)$ to the adversary $\mathcal{A}$ and append $(mid) \mapsto (m, S, R)$ to $\mathrm{p}_{\mathrm{msg}}$.
- Upon receiving $(\mathtt{send\ ok}, mid)$ from the adversary look up $(m, S, R) := \mathrm{p}_{\mathrm{msg}}(mid)$. If it exists, and if $\mathtt{f}_{\mathrm{act}}(\{S,R\}) = \mathsf{true}$, output $(\mathtt{sent}, S, m)$ to R.
- Upon receiving $(\mathtt{expire}, B)$ from some party A, set $\mathtt{f}_{\mathrm{act}}(\{A,B\}) := \mathsf{false}$.

The Ideal Functionality $\mathcal{F}_{\textsf{KRK}}$

Key registration with knowledge (KRK) does not only facilitate a simple public key infrastructure, it additionally guarantees that parties have knowledge of a secret key which corresponds to their public credentials. There are (at least) three different possibilities for an ideal functionality to provide this, which differ in who ultimately chooses the public and secret keys upon registration:

(1) The registering party asks the functionality for keys.

(2) The registering party provides the functionality with a secret key. The functionality then determines a corresponding public key.

(3) The registering party provides the functionality with a secret and public key pair. The functionality determines whether this is a well-formed key pair.

I think the third option is not only the one which gives the least power to the ideal functionality, but also the most realistic. I will therefore employ this version but keep in mind that possibilities (1) and (2) would serve us equally well. The other versions may in particular be preferable in cases where the public key infrastructure does not permit a way for the functionality to efficiently decide whether a key pair is valid or not. I model this by parameterising the ideal functionality with an efficiently computable Boolean function

$$\mathtt{f}_{\mathrm{Key}} : \mathbf{SK} \times \mathbf{PK} \to \{\mathsf{true}, \mathsf{false}\}.$$

On input of a key pair (sk, pk) this function outputs true for any well-formed keys and false otherwise. Note that if in version (2) there is a deterministic function $\mathtt{f}_{\mathbf{SK}\to\mathbf{PK}}$ which on input sk outputs a corresponding public key pk, this can be transformed to the third case via

$$\mathtt{f}_{\mathrm{Key}}(sk, pk) := [\mathtt{f}_{\mathbf{SK}\to\mathbf{PK}}(sk) = pk].$$

The following formal definition is loosely based on [BCNP04] but more well-defined and slightly different:

$$\mathcal{F}_{\mathbf{KRK}}^{\mathtt{f}_{\mathrm{Key}}}$$

Provides:
Key registration with knowledge.

Parameters:

- Function $\mathtt{f}_{\mathrm{Key}} : (sk, pk) \mapsto \begin{cases} \mathsf{true}, & \text{well-formed key pair} \\ \mathsf{false}, & \text{otherwise} \end{cases}$

State:

- Partial function $\mathrm{p}_{\mathrm{reg}} : \mathbf{MID} \to \mathbf{P} \times \mathbf{SK} \times \mathbf{PK}, mid \mapsto (P, sk, pk)$ of pending registrations.
- Partial function $\mathrm{p}_{\mathrm{ret}} : \mathbf{MID} \to \mathbf{P}^2, mid \mapsto (P_i, P_j)$ of pending retrievals.
- Set $\mathbf{R}$ of registered tuples (P, sk, pk).

Behaviour:

- Upon receiving $(\texttt{register}, sk, pk)$ from a party P, draw fresh mid, send $(\texttt{register}, mid, P, pk)$ to the adversary $\mathcal{A}$ and append $mid \mapsto (P, sk, pk)$ to p_{reg}.
- Upon receiving $(\texttt{register ok}, mid)$ from the adversary $\mathcal{A}$, retrieve $(P, sk, pk) := \mathsf{p}_{\text{reg}}(mid)$, check
  - $\mathtt{f}_{\text{Key}}(sk, pk) = \mathsf{true}$
  - $\nexists\, sk', pk' \colon (P, sk', pk') \in \mathbf{R}$
  - $\nexists\, P', sk' \colon (P', sk', pk) \in \mathbf{R}$

  and append (P, sk, pk) to $\mathbf{R}$ if all checks were successful.
- Upon receiving $(\texttt{retrieve}, P_i)$ from a party P_j, draw fresh mid, send $(\texttt{retrieve}, mid, P_i, P_j)$ to the adversary $\mathcal{A}$ and append $mid \mapsto (P_i, P_j)$ to p_{ret}.
- Upon receiving $(\texttt{retrieve ok}, mid)$ from the adversary $\mathcal{A}$, look up $(P_i, P_j) := \mathsf{p}_{\text{ret}}(mid)$ and $(P_i, sk_i, pk_i) \in \mathbf{R}$. If no such entry exists in $\mathbf{R}$, set $pk_i := \perp$. Send $(\texttt{retrieved}, pk_i, P_i)$ to P_j.

The three checks $\mathcal{F}_{\text{KRK}}$ performs before registering a party's credentials guarantee that only valid key pairs may be registered, that each party registers at most one key pair and that no two parties can share the same public key.

"I bet we can construct a cool proof-of-identity protocol. I'll start by picking two random–"
"Oh good, it's you. Here's the password…"

Randall Munroe, xkcd #1121

CHAPTER 3
Sender-binding Encryption

This chapter is wholly dedicated to the first—and main—research question of my dissertation:

What is the weakest public-key encryption security notion in order to establish multiple secure message transfer from existing authenticated channels?

I will look at this question in the context of UC security, where secure message transfer and authenticated channels are formalised as the ideal functionalities $\mathcal{F}_{\text{M-SMT}}$ and $\mathcal{F}_{\text{AUTH}}$, respectively (cp. Section 2.2.2). Furthermore I will restrict myself to adversaries with static corruption capabilities.

To start off, we firstly look at related work and the prior headway made towards such a weakest public-key encryption security notion in Section 3.1. In the central part of this chapter, Section 3.2, I provide formal definitions of the new sender-binding encryption as well as IND-SB-CPA security concepts and the intuition behind them. The rest of the chapter is dedicated to proving that IND-SB-CPA secure sender-binding encryption is an almost perfect answer to the above research question: I formally prove that my definition is strong enough to UC-realise the secure message transfer functionality $\mathcal{F}_{\text{M-SMT}}$ in the $\mathcal{F}_{\text{AUTH}}$ hybrid model in Section 3.3. Section 3.4 provides the reverse direction. Here I show that—restricted to the commonly used encrypt-then-authenticate principle—IND-SB-CPA secure sender-binding encryption is the weakest possible public-key encryption notion to allow for secure message transfer from authenticated channels.

I have previously published the new definitions and insights from this chapter in [BGMOS22] and [Sch24], respectively.

3.1 Prior Efforts

Firstly, let me give you an overview of how prior works UC-realised (multi-message) secure message transfer from some form of encryption in the $\mathcal{F}_{\text{AUTH}}$ hybrid model. This process started with classic public-key encryption notions and developed to tag-based encryption from there. Although it is not explicitly stated in the original papers, I got the impression that this development was strongly driven by insights of the form "if the adversary is able to do *xyz*, the authenticated channel already prevents any real exploitation of this power". To give a better intuition about the different security notions and constructions I will highlight what these "inconsequential" adversarial powers are throughout this section. For formal definitions of the different schemes and security notions, please refer back to Section 2.1.1.

Secure Message Transfer from Classic Public-key Encryption

In the original version of [Can01], Canetti showed how multi secure message transfer can be realised in the $\mathcal{F}_{\text{PKE}}, \mathcal{F}_{\text{AUTH}}$ hybrid model by simply concatenating the sender ID to the message. It was also shown that $\mathcal{F}_{\text{PKE}}$ can be realised via IND-CCA2 secure encryption, giving rise to an secure message transfer construction via $\mathcal{F}_{\text{AUTH}}$ and IND-CCA2 secure encryption. While [Can01] claimed that a weaker notion than IND-CCA2 would suffice for $\mathcal{F}_{\text{PKE}}$, [CKN03] proved an equivalence of $\mathcal{F}_{\text{PKE}}$ and IND-CCA2. Also in [CKN03] the notion of IND-RCCA secure encryption was introduced. This notion was shown to be strictly weaker than IND-CCA2 security but still sufficient to realise secure message transfer via the same protocol, just without the detour over $\mathcal{F}_{\text{PKE}}$. The core of the protocol used in [Can01] and [CKN03] is the following: When party S wants to send a message m to party R...

(1) S encrypts (S, m) under the public key of R.

(2) S sends the resulting ciphertext c via $\mathcal{F}_{\text{AUTH}}$ to R.

(3) R decrypts c to m' and checks that this is of the form $m' = (S, m)$ for the sender S indicated by $\mathcal{F}_{\text{AUTH}}$.

Both IND-CCA2 security and $\mathcal{F}_{\text{PKE}}$ are designed with the intuition of "best possible encryption security". The weakening idea behind IND-RCCA security is that simple replay attacks of the same message content from a different sender inside a different—e.g. rerandomised—ciphertext do not pose a problem, as long as the sender ID is part of the message within the secure message transfer protocol.[3] In this case, even if the adversary can modify an honestly generated ciphertext to look different and have it sent from another (corrupted) party, it will just be rejected upon decryption, since the adversary is not able to modify the sender ID within the ciphertext.

Secure Message Transfer from Tag-based Encryption

In [MRY04], IND-atag-CCA (also sometimes called "weak CCA" for tag-based encryption) was introduced. MacKenzie, Reiter and Yang show that using party IDs as the set of tags and the following protocol idea, IND-atag-CCA can be used to realise secure message transfer via $\mathcal{F}_{\text{AUTH}}$:

(1) S encrypts m with tag S under the public key of R via the tag-based encryption scheme.

(2) S sends the resulting ciphertext c via $\mathcal{F}_{\text{AUTH}}$ to R.

(3) R decrypts c to m using the sender S indicated by $\mathcal{F}_{\text{AUTH}}$ as the tag.

A slightly weaker notion, IND-stag-CCA security, was later introduced by Kiltz in [Kil06]. Kiltz did not broach the issue of secure message transfer, but rather showed that IND-stag-CCA security suffices to employ the Canetti, Halevi, and Katz transformation (see [CHK04]) to obtain IND-CCA2 security and therefore also secure message transfer in a round about way via the above public-key encryption construction.

[3] For other IND-RCCA applications it is often a case of "As long as the message contains a counter or message ID as well".

The intuition here goes one step further than with IND-RCCA encryption: The actual message content of a ciphertext may be largely modifiable. But as long as the tag that a ciphertext is bound to can not be changed by the adversary, they do not gain any advantage by modifying the message within a ciphertext when sender IDs are used as the tags to construct secure message transfer via $\mathcal{F}_{\text{AUTH}}$. Again, a modified message from the wrong sender is just discarded upon decryption.

3.2 Sender-binding Encryption and IND-SB-CPA Security

Now that we know the extent to which our research question has previously been worked on, I will go on to introduce the concept of sender-binding encryption and the corresponding new security notion of IND-SB-CPA in this section. It is even weaker than the IND-atag-CCA relaxation IND-stag-CCA—which was the latest development we discussed in Section 3.1—but still captures the security needed for secure message transfer via authenticated channels. Although the term sender-binding encryption has not previously been defined, we will see in Remark 1 that all prior realisations of secure message transfer via authenticated channels (based on IND-CCA2, -RCCA, -atag-CCA or -stag-CCA) work by constructing (IND-SB-CPA secure) sender-binding encryption from the underlying encryption scheme. I therefore regard sender-binding encryption as a long overdue unifying definition which is central for the topic of secure message transfer construction.

Definition (SBE):
The interface of a *sender-binding encryption (SBE) scheme* with message space $\mathbf{M}$ and ID space $\mathbf{P}$ is given by a set of three PPT algorithms $(\texttt{gen}, \texttt{enc}, \texttt{dec})$:

$$\begin{aligned} \texttt{gen:} \quad & 1^\lambda \mapsto (sk, pk) \\ \texttt{enc:} \quad & (pk, S, m) \mapsto c \\ \texttt{dec:} \quad & (sk, S, c) \mapsto m. \end{aligned}$$

We expect a sender-binding encryption scheme to fulfil the notion of *correctness*, i.e. that whenever $(sk, pk) \leftarrow \texttt{gen}(1^\lambda)$, $m \in \mathbf{M}$ and $S \in \mathbf{P}$, then

$$m = \texttt{dec}\Big(sk, S, \texttt{enc}(pk, S, m)\Big). \qquad \circ$$

Some remarks are in order about the use case of this definition.

In addition to the inputs present in any common public-key encryption scheme, sender-binding encryption and decryption algorithms use the encrypting party's ID S[4] as well. The ID of a party represents the identification information used within the system. This might be the public key itself, the party's actual name, their e-mail address etc. The only requirement here is uniqueness.

With any public-key encryption scheme the ciphertext is bound to the receiving party who holds the secret key and is able to decrypt the ciphertext. Using the sending party's ID during encryption and decryption additionally binds the ciphertext to the party who created it.

[4] For the encryption mechanism I will sometimes omit the explicit input of the ID S if it is clear from the context which party S is conducting the encryption.

However, binding a ciphertext to the ID of a sending/encrypting party alone does not yet yield obvious benefits. Even if a specific party ID is specified by the protocol, party IDs are public knowledge and malicious parties can insert any ID they want. Sender-binding encryption starts to unfold its benefit when used in conjunction with IDs that are associated with authenticated channels. This channel reliably indicates the true sender S of a message. Checking this against the sender ID bound to the received ciphertext prevents message stealing attacks by different senders, i.e. that the ciphertext was just copied from another (unwitting) sender. The terminology "sender-binding" stems from the example application of secure message transfer via authenticated channels where the ID is taken to be that of the encrypting/sending party. Of course there might be other use cases for sender-binding encryption where the encrypting party does not constitute a "sender". But throughout this dissertation—whenever I talk about sender-binding encryption—I use R and "receiver" to denote the party owning the keys

$$(sk_R, pk_R) := (sk, pk),$$

and S and the term "sender" for the party whose ID is input on encryption and decryption.

Given the definition of a sender-binding encryption scheme we still need to arrive at a meaningful corresponding security notion. To define a sender-binding encryption security notion that is not unnecessarily strong, we need different security properties for the sender ID and message, respectively. In a similar way, security notions for tag-based encryption schemes already differentiate the security properties needed for the tag and message. Considering furthermore, that we can turn any tag-based encryption scheme into sender-binding encryption by choosing the tag space $\mathbf{T}$ to be the set of party IDs $\mathbf{P}$, tag-based encryption seems to be an intuitive starting point to derive our new security notion. Note that even a tag-based encryption scheme with arbitrary tag space $\mathbf{T}$ can easily be used for sender-binding encryption as long as the tag space is at least as large as the set $\mathbf{P}$ of participating parties. To do so a public and injective function $\mathbf{P} \hookrightarrow \mathbf{T}$ is chosen to translate party IDs into tags.

Let me start by introducing the tag-based encryption notion of indistinguishability under given-tag weakly chosen ciphertext attack (IND-gtag-CCA). This is an intuitive weakening of the previously considered IND-stag-CCA, with the sole difference being that the adversary is not allowed to choose the challenge tag but is instead given a randomly drawn tag by the challenger:

Definition (IND-gtag-CCA):
A tag-based encryption scheme $(\mathtt{gen}, \mathtt{enc}, \mathtt{dec})$ satisfies *indistinguishability under given-tag weakly chosen ciphertext attack (IND-gtag-CCA)*, if and only if for any PPT adversary $\mathcal{A}_{\text{gtag-CCA}}$ the advantage to win the IND-gtag-CCA game shown in Figure 9 is negligible in λ. ◦

Using party IDs as tags in tag-based encryption provides a special meaning to these tags. It is this additional meaning which requires us to deviate from tag-based encryption and induces the changes I make to IND-gtag-CCA to arrive at the new notion of IND-SB-CPA for sender-binding encryption: There now is an additional connection between tags and key pairs, as any party ID (tag) is associated to the key pair of this party. Hence there inherently is another ID/tag R corresponding to the key pair

$$(sk_R, pk_R) = (sk, pk)$$

and most likely another key pair (sk_S, pk_S) corresponding to the party with ID $S = \tau^*$. As we are aiming towards the weakest possible notion from which to construct secure message

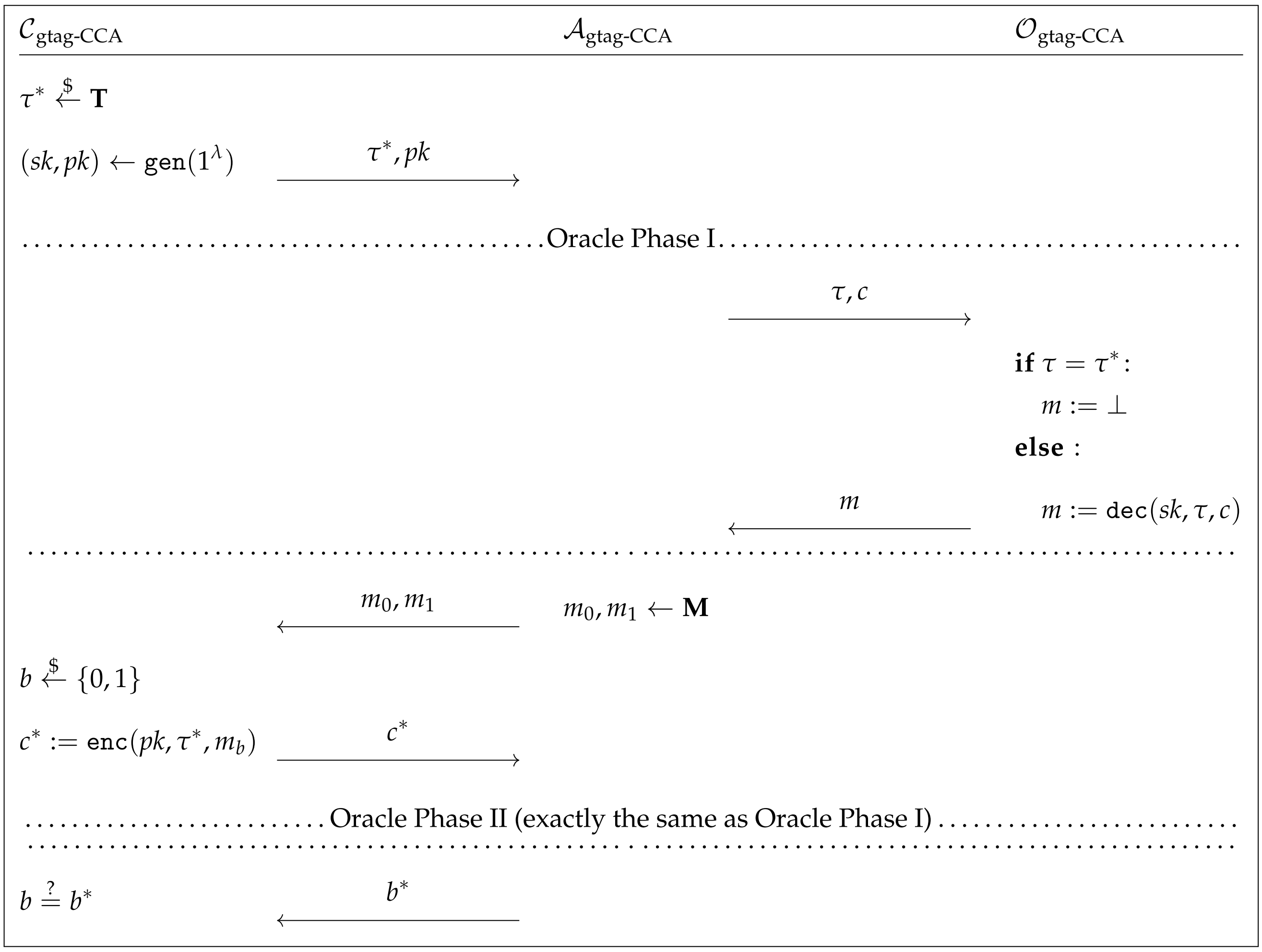

$\mathcal{C}_{\text{gtag-CCA}}$
$\mathcal{A}_{\text{gtag-CCA}}$
$\mathcal{O}_{\text{gtag-CCA}}$
$\tau^* \xleftarrow{\$} \mathbf{T}$
$(sk, pk) \leftarrow \mathtt{gen}(1^\lambda)$
τ^*, pk
Oracle Phase I
τ, c
if $\tau = \tau^*$:
$m := \perp$
else :
m
$m := \mathtt{dec}(sk, \tau, c)$
m_0, m_1
$m_0, m_1 \leftarrow \mathbf{M}$
$b \xleftarrow{\$} \{0,1\}$
$c^* := \mathtt{enc}(pk, \tau^*, m_b)$
c^*
Oracle Phase II (exactly the same as Oracle Phase I)
$b \overset{?}{=} b^*$
b^*

Figure 9: The IND-gtag-CCA Game for Tag-based Encryption

transfer we let both of those be chosen by the challenger instead of giving the adversary any more power. Depending on the underlying encryption scheme it is possible that keys may not be generated independently of the ID (think, e.g. of identity-based encryption) or that public keys are used as IDs themselves. Hence we assume the challenger to randomly generate/draw keys and IDs in a consistent fashion. With the additional key pair (sk_S, pk_S) we also need to define how much decryption power the adversary gets for these keys in the two oracle phases. We choose this to be symmetric with the challenge keys (sk_R, pk_R). Because this gives a weaker notion and is still enough for secure message transfer, we restrict decryption not only for the challenge tag S but for R as well. All in all this adjustment of IND-gtag-CCA to sender-binding encryption yields the following definition:

Definition (IND-SB-CPA):
An sender-binding encryption scheme $(\mathtt{gen}, \mathtt{enc}, \mathtt{dec})$ satisfies *indistinguishability under sender-binding chosen plaintext attack (IND-SB-CPA)*, if and only if for any PPT adversary $\mathcal{A}_{\text{SB-CPA}}$ the advantage to win the IND-SB-CPA game shown in Figure 10 is negligible in λ. ○

Within this context of sender-binding encryption, the new security notion of IND-SB-CPA has a very straight forward intuition: If it was possible to alter a ciphertext

$$c \leftarrow \mathtt{enc}(pk, S, m)$$

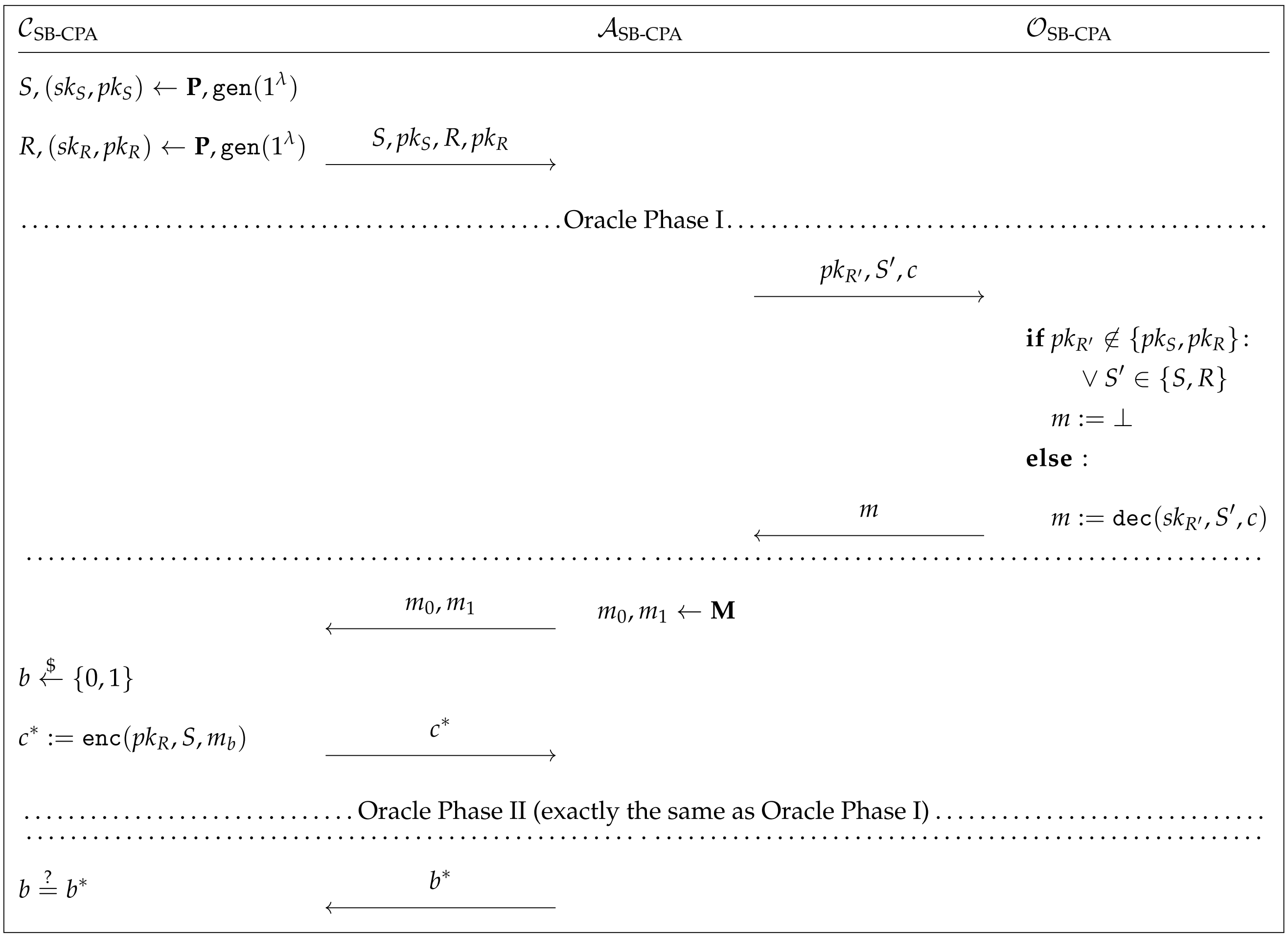

$\mathcal{C}_{\text{SB-CPA}}$
$\mathcal{A}_{\text{SB-CPA}}$
$\mathcal{O}_{\text{SB-CPA}}$
$S, (sk_S, pk_S) \leftarrow \mathbf{P}, \texttt{gen}(1^\lambda)$
$R, (sk_R, pk_R) \leftarrow \mathbf{P}, \texttt{gen}(1^\lambda)$
S, pk_S, R, pk_R
Oracle Phase I
$pk_{R'}, S', c$
if $pk_{R'} \notin \{pk_S, pk_R\}$:
$\vee\, S' \in \{S, R\}$
$m := \perp$
else :
m
$m := \texttt{dec}(sk_{R'}, S', c)$
m_0, m_1
$m_0, m_1 \leftarrow \mathbf{M}$
$b \overset{\$}{\leftarrow} \{0,1\}$
$c^* := \texttt{enc}(pk_R, S, m_b)$
c^*
Oracle Phase II (exactly the same as Oracle Phase I)
$b \overset{?}{=} b^*$
b^*

Figure 10: The IND-SB-CPA Game for Sender-binding Encryption

to some c' which successfully decrypted under another sender ID S' (i.e. $\texttt{dec}(sk_R, S', c') \neq \perp$), message stealing attacks from an external party S' would be possible. Let us look at this in a bit more detail. From Figure 10 we see that the adversary is provided with perfect knowledge (via oracle or its own power) about any ciphertext which involves any other party than just S and R. About communication between S and R, on the other hand, the adversary learns nothing—with the natural exception that encryption only requires public knowledge and can therefore be conducted by the adversary as well. A directed version—where the adversary can additionally decrypt messages from R to S (but not from S to R)—would also naturally suggest itself. But as mentioned before, our choice of a symmetric version is strictly weaker as well as sufficient for secure message transfer. Having no decryption possibilities for the channel (S to R) along which the challenge ciphertext is sent leads me to classifying IND-SB-CPA as some form of CPA security. For more thoughts on these classifications see Section 6.3.

Remark 1 (Unifying Prior Efforts):
As discussed in Section 3.1, there are previous constructions to UC-realise secure message transfer from ideal authenticated channels based on classic public-key encryption as well as tag-based encryption. For classic public-key encryption the sender ID is concatenated to the message, while for tag-based encryption the sender ID is used at the tag when encrypting the message.

With the new sender-binding encryption definition it is easy to see that both of those methods work by firstly transforming classic public-key encryption and tag-based encryption into sender-binding encryption before combining the resulting sender-binding encryption scheme with ideal authenticated channels. It is not quite as obvious that for all prior constructions—based on IND-CCA2/-RCCA public-key encryption or IND-atag-CCA/-stag-CCA tag-based encryption—the resulting sender-binding encryption scheme also satisfies IND-SB-CPA security. However, I provide formal proofs for this in Sections 5.1.1 and 6.1.1, respectively.

Hence we see that the concept of IND-SB-CPA secure sender-binding encryption does in fact unify all prior approaches to UC-realise secure message transfer from existing authenticated channels. ○

But first, let us have a look at how to UC-realise secure message transfer from IND-SB-CPA secure sender-binding encryption and authenticated channels.

3.3 Realising Secure Message Transfer

In this section I prove that IND-SB-CPA secure sender-binding encryption suffices in conjunction with authenticated channels to realise secure message transfer. The proof is conducted in the universal composability (UC) model of Canetti [Can01]—which is explained in more detail in Section 2.2—using *static corruptions* only. This means that the adversary chooses which parties to corrupt when they first activate a party. The adversary is not allowed to corrupt honest parties adaptively during computation.

We will proceed towards the goal of realising secure message transfer in three stages: Firstly, we define a candidate protocol $\pi_{\text{M-SMT}}$ in the $\mathcal{F}_{\text{AUTH}}$-hybrid model which utilises sender-binding encryption. Secondly, we construct a simulator $\mathcal{S}_{\text{M-SMT}}$ aiming to provide indistinguishability between the candidate protocol and the secure message transfer functionality $\mathcal{F}_{\text{M-SMT}}$. The last step is formally proving in the $\mathcal{F}_{\text{AUTH}}$-hybrid model, that indistinguishability from $\mathcal{F}_{\text{M-SMT}}$ is actually achieved by $\pi_{\text{M-SMT}}$ in conjunction with $\mathcal{S}_{\text{M-SMT}}$ if the sender-binding encryption scheme satisfies IND-SB-CPA security.

3.3.1 Protocol $\pi_{\text{M-SMT}}$

Let $(\texttt{gen}, \texttt{enc}, \texttt{dec})$ be a sender-binding encryption scheme. From this we define a secure message transfer protocol $\pi_{\text{M-SMT}}$ as follows: Whenever a party S wants to securely transmit a message m to some party R, they essentially send the encryption

$$c \leftarrow \texttt{enc}(pk_R, S, m)$$

over an authenticated channel to R. When a party R receives a ciphertext c over an authenticated channel from some party S, they decrypt it via

$$m := \texttt{dec}(sk_R, S, c).$$

Although this general principle is very simple, many details—e.g. regarding key generation—need to be taken into account. The formal and more detailed definition looks as follows:

$\pi_{\textbf{M-SMT}}$

Realises:
Multi-receiver multi-message multi-sender secure message transfer with message space $\mathbf{M}$ and polynomially many parties $P \in \mathbf{P}$.

Parameters:

- Sender-binding encryption scheme $(\texttt{gen}, \texttt{enc}, \texttt{dec})$ with message space $\mathbf{M}$.
- Ideal authenticated channel functionality $\mathcal{F}_{\text{AUTH}}$.

State of Party P:

- Personal SBE key pair $(sk_P, pk_P) \in \mathbf{SK} \times \mathbf{PK}$.
- Partial function $\texttt{f}_{\mathbf{PK}} \colon \mathbf{P} \to \mathbf{PK}, P' \mapsto pk_{P'}$ of known public keys.
- Function $\text{p}_{\text{msg}} \colon \mathbf{P} \to \mathbf{M}^*$ of pending messages.

Behaviour of Party P:

```
\\ Being asked to initialise
• Upon receiving output $(\texttt{sent}, S, P, \texttt{init})$ from $\mathcal{F}_{\text{AUTH}}$, if there is no personal key pair $(sk_P, pk_P)$ yet:
    (1) $(sk, pk) \leftarrow \texttt{gen}(1^\lambda)$.
    (2) Set $(sk_P, pk_P)$ to $(sk, pk)$.
    (3) For each party $P' \neq P$: Hand input $(\texttt{send}, P', (\texttt{inited}, pk))$ to a new instance of $\mathcal{F}_{\text{AUTH}}$.
\\ Receiving keys and sending stored messages
• Upon receiving output $(\texttt{sent}, P', P, (\texttt{inited}, pk_{P'}))$ from $\mathcal{F}_{\text{AUTH}}$, if there is no entry $\texttt{f}_{\mathbf{PK}}(P')$ yet:
    (1) Append $P' \mapsto pk_{P'}$ to $\texttt{f}_{\mathbf{PK}}$.
    (2) For any $m \in \text{p}_{\text{msg}}(P')$:
        (1) Remove $m$ from $\text{p}_{\text{msg}}(P')$.
        (2) $c \leftarrow \texttt{enc}(pk_{P'}, P, m)$.
        (3) Hand input $(\texttt{send}, P', c)$ to a new instance of $\mathcal{F}_{\text{AUTH}}$.
\\ Sending messages
• Upon receiving input $(\texttt{send}, R, m)$ with $m \in \mathbf{M}$ from environment $\mathcal{Z}$:
    ◦ If $R = P$ report output $(\texttt{sent}, P, m)$ to the environment.
    ◦ Else if no entry $\texttt{f}_{\mathbf{PK}}(R)$ exists yet:
        (1) Append $m$ to $\text{p}_{\text{msg}}(R)$.
        (2) Hand input $(\texttt{send}, R, \texttt{init})$ to a new instance of $\mathcal{F}_{\text{AUTH}}$.
    ◦ Else:
        (1) $pk_R := \texttt{f}_{\mathbf{PK}}(R)$.
```

(2) $c \leftarrow \texttt{enc}(pk_R, P, m)$.

(3) Hand input $(\texttt{send}, R, c)$ to a new instance of $\mathcal{F}_{\text{AUTH}}$.

\\ Receiving messages

- Upon receiving output $(\texttt{sent}, S, R, c)$ from $\mathcal{F}_{\text{AUTH}}$:

  (1) $m \leftarrow \texttt{dec}(sk, S, c)$.

  (2) Report output $(\texttt{sent}, S, m)$ to the environment $\mathcal{Z}$.

3.3.2 Simulator $\mathcal{S}_{\text{M-SMT}}$

According to the real/ideal paradigm explained in Section 2.2, our protocol $\pi_{\text{M-SMT}}$ realises secure message transfer if and only if for any dummy adversary $\mathcal{D}$ interacting with the real protocol, there exists a simulator $\mathcal{S}$ interacting with the ideal functionality $\mathcal{F}_{\text{M-SMT}}$ such that no environment $\mathcal{Z}$ can distinguish between executions in the real and ideal world. We now construct such a simulator $\mathcal{S}_{\text{M-SMT}}$ which we will later show to achieve indistinguishability for $\pi_{\text{M-SMT}}$ and $\mathcal{F}_{\text{M-SMT}}$ in case the sender-binding encryption scheme satisfies IND-SB-CPA security.

The main idea of the simulator $\mathcal{S}_{\text{M-SMT}}$ is that it almost perfectly simulates the protocol behaviour of all parties and the hybrid functionality $\mathcal{F}_{\text{AUTH}}$ in its head. It takes inputs to and reports messages and outputs from these in-the-head parties to $\mathcal{Z}$ on the one hand and uses them on the other hand to interface with the ideal functionality $\mathcal{F}_{\text{M-SMT}}$. The only case in which the simulator does not have sufficient knowledge to perfectly simulate the protocol in its head is when an honest party S sends a message m to another honest party R: The simulator has no way of knowing the actual message m. In this case $\mathcal{S}_{\text{M-SMT}}$ reports an encryption

$$c \leftarrow \texttt{enc}(pk_R, S, 0)$$

of all zeros to have been send instead.

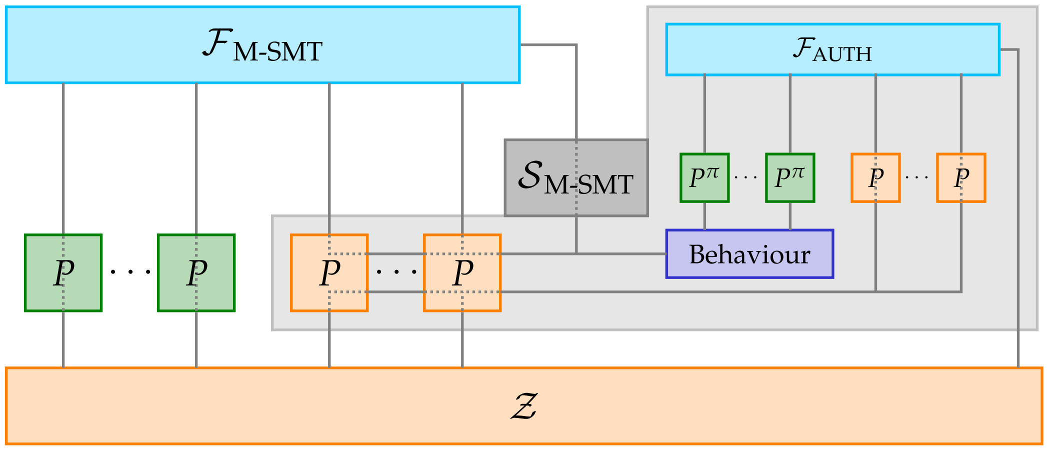

$\mathcal{F}_{\text{M-SMT}}$
$\mathcal{F}_{\text{AUTH}}$
$\mathcal{S}_{\text{M-SMT}}$
P^π ... P^π
P ... P
Behaviour
P ... P
P ... P
$\mathcal{Z}$

Figure 11: Overview of Simulator $\mathcal{S}_{\text{M-SMT}}$

The overall construction of $\mathcal{S}_{\text{M-SMT}}$ is shown in Figure 11. Again there are some more details to keep track of, especially regarding the box labelled "Behaviour" in Figure 11. So let me provide a more formal definition as well:

$\mathcal{S}_{\textbf{M-SMT}}$

Realises:
Multi-receiver multi-message multi-sender secure message transfer with message space $\mathbf{M}$ and polynomially many parties $P \in \mathbf{P}$.

Parameters:
Sender-binding encryption scheme $(\texttt{gen}, \texttt{enc}, \texttt{dec})$ with message space $\mathbf{M}$.

In-the-head Parties:

- Functionality $\mathcal{F}_{\text{AUTH}}$. This functionality communicates in-the-head with all honest in-the-head parties as well as with the environment $\mathcal{Z}$ as adversary.
- Copies of honest parties running the protocol $\pi_{\text{M-SMT}}$, which we will denote as P^π. These parties communicate in-the-head with the in-the-head functionality $\mathcal{F}_{\text{AUTH}}$. Their interface to the environment is played by the simulator (defined in "Behaviour" below).
- Dummy corrupted parties. Whenever the simulator is asked by the environment to call the functionality $\mathcal{F}_{\text{AUTH}}$ in the name of a corrupted party, this in-the-head dummy calls the in-the-head functionality correspondingly and reports all outputs back to the environment $\mathcal{Z}$.

State:
Everything the in-the-head parties store in their states.

Behaviour:

\\ Self-communication

- Upon receiving $(\texttt{send}, \textit{mid}, P, P)$ from $\mathcal{F}_{\text{M-SMT}}$ for honest party P, call $\mathcal{F}_{\text{M-SMT}}$ with input $(\texttt{send ok}, \textit{mid})$.

\\ Message from honest to honest party

- Upon receiving $(\texttt{send}, \textit{mid}, S, R)$ from $\mathcal{F}_{\text{M-SMT}}$ for honest parties $S \neq R$:
    - Start in-the-head party S^π with input $(\texttt{send}, R, 0)$ from the environment $\mathcal{Z}$.
    - If in-the-head party R^π at some point reports output $(\texttt{sent}, S, 0)$, call $\mathcal{F}_{\text{M-SMT}}$ with input $(\texttt{send ok}, \textit{mid})$.[5]

\\ Message from honest to corrupted party

- Upon receiving $(\texttt{send}, \textit{mid}, S, R)$ from $\mathcal{F}_{\text{M-SMT}}$ for honest party S and corrupted party R:
    (1) Call $\mathcal{F}_{\text{M-SMT}}$ with input $(\texttt{send ok}, \textit{mid})$.
    (2) Receive output $(\texttt{sent}, S, m)$ from $\mathcal{F}_{\text{M-SMT}}$ to R.
    (3) Start in-the-head party S^π with input $(\texttt{send}, R, m)$ from the environment $\mathcal{Z}$.

\\ Message from corrupted to honest party

- Upon in-the-head honest party R^π reporting output $(\texttt{sent}, S, m)$ for corrupted party S:
    (1) Call $\mathcal{F}_{\text{M-SMT}}$ with input $(\texttt{send}, R, m)$ in the name of S.
    (2) Receive output $(\texttt{send}, \textit{mid}, S, R)$ from $\mathcal{F}_{\text{M-SMT}}$.
    (3) Call $\mathcal{F}_{\text{M-SMT}}$ with input $(\texttt{send ok}, \textit{mid})$.

3.3.3 Security Theorem and Proof

The last thing left to do is to prove that under static corruption the simulator $\mathcal{S}_{\text{M-SMT}}$ does in fact achieve indistinguishability between the protocol $\pi_{\text{M-SMT}}$ and the ideal functionality $\mathcal{F}_{\text{M-SMT}}$ in the $\mathcal{F}_{\text{AUTH}}$-hybrid model if the sender-binding encryption scheme satisfies IND-SB-CPA security. To do so we will reduce this indistinguishability to the IND-SB-CPA security of the underlying sender-binding encryption scheme. I.e. assuming there is an environment $\mathcal{Z}$ which can efficiently and with non-negligible probability distinguish a real execution of $\pi_{\text{M-SMT}}$ from an ideal experiment with $\mathcal{F}_{\text{M-SMT}}$ and $\mathcal{S}_{\text{M-SMT}}$, we construct an adversary $\mathcal{A}_{\text{SB-CPA}}$ who can win the IND-SB-CPA game with non-negligible probability.

To achieve this let us first take a closer look at what a successfully distinguishing environment needs to do:

Remark 2:
From the definition of the simulator $\mathcal{S}_{\text{M-SMT}}$ we immediately see that if an environment $\mathcal{Z}$ is able to distinguish executions of $\mathcal{F}_{\text{M-SMT}}$ and $\pi_{\text{M-SMT}}$, it can only do so by messages between honest parties $S \neq R$. In this case the simulator prompts its in-the-head sender S^{π} to send an all-zero message to R instead of the actual message m (which the simulator does not know). The environment will therefore receive a message

$$\Big(\texttt{send}, S, R, \texttt{enc}(pk_R, S, 0)\Big)$$

from $\mathcal{F}_{\text{AUTH}}$ (played by $\mathcal{S}_{\text{M-SMT}}$) in the ideal execution, while in the protocol execution it receives

$$\Big(\texttt{send}, S, R, \texttt{enc}(pk_R, S, m)\Big).$$

In all other cases the simulator can perfectly mimic the protocol execution by playing the relevant parties and functionality $\mathcal{F}_{\text{AUTH}}$ in its head.[6] ◦

Let us restrict the distinguishing possibilities even more by introducing a sequence of hybrids which allows us to employ the polynomial hybrid argument from Lemma 2 (cf. Section 2.2.1) in the security proof:

Definition (Hybrids H_i):
Let $i \in \mathbb{N}_0$ be a natural number. The hybrid H_i represents the execution set-up where almost all interactions are handled as in the real world execution of $\pi_{\text{M-SMT}}$. Note that Remark 2 guarantees that these are the same as in the ideal world, apart from encryptions of messages between distinct honest parties. Now the only difference between an execution of $\pi_{\text{M-SMT}}$ and H_i is the following: For the first i messages m_j ($j \leqslant i$) between two honest parties $R_j \neq S_j$, the output

$$\Big(\texttt{send}, S_j, R_j, \texttt{enc}(pk_{R_j}, S_j, 0)\Big)$$

from $\mathcal{F}_{\text{AUTH}}$ to the environment $\mathcal{Z}$ contains an encryption of zeros—as it would in the ideal execution with simulator $\mathcal{S}_{\text{M-SMT}}$—instead of an encryption of the real message m_j. ◦

[5] At this point we assume the simulator to track the protocol executions in their head so they know which *mid* to use. For readability purposes we refrained from introducing notation to explicitly store this.

[6] Please convince yourself from the definition of the simulator $\mathcal{S}_{\text{M-SMT}}$ that it has all the knowledge required for simulation and that activations/outputs of $\mathcal{F}_{\text{M-SMT}}$ will actually occur at the right times.

Note that by this definition H_0 is equal to the real world execution of $\pi_{\text{M-SMT}}$ and two hybrids H_{i-1} and H_i do not differ in anything other than the output of $\mathcal{F}_{\text{AUTH}}$ for the i-th message between two honest parties. Furthermore the limit

$$H_\infty := \lim_{i \to \infty} H_i = \mathsf{EXEC}(\mathcal{F}_{\text{M-SMT}}, \mathcal{S}_{\text{M-SMT}})$$

denotes the ideal world execution of $\mathcal{F}_{\text{M-SMT}}$ with $\mathcal{S}_{\text{M-SMT}}$, where encryptions of zeros are used for all messages m_j, $j \in \mathbb{N}$. In particular, for any p-bounded environment $\mathcal{Z}$, all hybrids H_i with $i \geqslant \mathsf{p}(\lambda)$ are identical to the ideal world execution

$$\mathsf{EXEC}(\mathcal{F}_{\text{M-SMT}}, \mathcal{S}_{\text{M-SMT}}).$$

With this preparatory work, we are finally ready to prove that the protocol $\pi_{\text{M-SMT}}$ does in fact UC-realise secure message transfer:

Theorem 2:
Let the sender-binding encryption scheme $(\mathtt{gen}, \mathtt{enc}, \mathtt{dec})$ *employed in the protocol* $\pi_{M\text{-}SMT}$ *satisfy IND-SB-CPA security. Then under static corruption,* $\pi_{M\text{-}SMT}$ *is a UC realisation of* $\mathcal{F}_{M\text{-}SMT}$ *in the* $\mathcal{F}_{AUTH}$*-hybrid model, i.e.*

$$\pi_{M\text{-}SMT}^{\mathcal{F}_{AUTH}} \geqslant_{UC} \mathcal{F}_{M\text{-}SMT}.$$

Proof:
As mentioned before, we want to employ the polynomial hybrid argument from Lemma 2 for this proof. Hence we will firstly show that for any $i \in \mathbb{N}$ the two consecutive hybrids H_{i-1} and H_i are not only computationally indistinguishable, but that furthermore the advantage

$$\mathsf{Adv}^{\text{indist}}_{H_{i-1}, H_i, \mathcal{Z}_i}(\lambda)$$

is bounded independently of i by a function that is negligible in the security parameter λ.

Let $i \in \mathbb{N}$ and $\mathcal{Z}_i$ be an environment trying to distinguish executions of hybrids H_{i-1} and H_i. We now use environment $\mathcal{Z}_i$ to construct an adversary $\mathcal{A}_{\text{SB-CPA}}$ whose probability to win the IND-SB-CPA game with respect to the underlying sender-binding encryption scheme $\Sigma = (\mathtt{gen}, \mathtt{enc}, \mathtt{dec})$ is directly related to the success probability of $\mathcal{Z}_i$.

First, $\mathcal{A}_{\text{SB-CPA}}$ receives (S, pk_S, R, pk_R) from $\mathcal{C}_{\text{SB-CPA}}$. Then it starts $\mathcal{Z}_i$ in its head, playing all other parties. Again by Remark 2, $\mathcal{Z}_i$ needs to register at least two honest parties (and send a message between them) to distinguish H_{i-1} from H_i. For the two honest parties R and S (randomly chosen by the challenger), $\mathcal{A}_{\text{SB-CPA}}$ does not generate fresh credentials as the honest parties would do, but rather uses pk_S and pk_R from $\mathcal{C}_{\text{SB-CPA}}$.

It is no problem that $\mathcal{A}_{\text{SB-CPA}}$ does not know sk_R and sk_S. The only case they are used is when a corrupted party sends a message to R or S, i.e. when one of them receives output

$$\left(\mathtt{sent}, P, R/S, c\right)$$

for some corrupted party P from the functionality $\mathcal{F}_{\text{AUTH}}$. In this case $\mathcal{A}_{\text{SB-CPA}}$ prompts the oracle $\mathcal{O}_{\text{SB-CPA}}$ with input (pk_S, P, c). Note that it is $P \notin \{S, R\}$. Hence $\mathcal{O}_{\text{SB-CPA}}$ by definition responds with the decryption

$$m := \mathtt{dec}(sk_{S/R}, P, c)$$

and $\mathcal{A}_{\text{SB-CPA}}$ can call $\mathcal{F}_{\text{M-SMT}}$ with input $(\mathtt{send}, S/R, m)$ in the name of P as usual.

For the first $i-1$ messages which are sent between two distinct honest parties, $\mathcal{A}_{\text{SB-CPA}}$ reports encryptions of 0 when $\mathcal{Z}_i$ asks the adversary to see the content of the communication channel. When $\mathcal{Z}_i$ asks for the i-th message m_i to be sent, $\mathcal{A}_{\text{SB-CPA}}$ does the following:

- If m_i is not a message from S to R, give up on $\mathcal{Z}_i$ and guess random output $b \xleftarrow{\$} \{0,1\}$.
- If m_i is to be sent from S to R, hand messages 0 and m_i to $\mathcal{C}_{\text{SB-CPA}}$ and receive challenge ciphertext c^* in return. Report c^* as communication channel content to $\mathcal{Z}_i$. From now on, when a message m is sent between two honest parties, $\mathcal{A}_{\text{SB-CPA}}$ always reports an encryption of m as channel content instead of 0 as before. When $\mathcal{Z}_i$ stops and reports it has run in the hybrid H_i, $\mathcal{A}_{\text{SB-CPA}}$ reports bit $b=0$ to $\mathcal{C}_{\text{SB-CPA}}$. Otherwise, if $\mathcal{Z}_i$ decides on H_{i-1}, it reports $b=1$.

To analyse the success probability of $\mathcal{A}_{\text{SB-CPA}}$, note that the set of parties $\mathbf{P}$ by definition of $\mathcal{F}_{\text{M-SMT}}$ has size $|\mathbf{P}| = |\mathbf{P}(\lambda)|$ which is polynomial in the security parameter λ, identities S and R are drawn uniformly at random from $\mathbf{P}$ and it is information-theoretically impossible for $\mathcal{Z}_i$ to distinguish S and R from other honest parties. This means the probability for m_i to be sent from S to R is at least

$$\mathbb{P}[m_i : S \rightsquigarrow R] \geqslant \frac{1}{|\mathbf{P}|^2}.$$

In case m_i *is* sent from S to R, $\mathcal{A}_{\text{SB-CPA}}$ has the same success probability as $\mathcal{Z}_i$. Together with the even success probability in case m_i is *not* sent from S to R, we get an overall success probability of

$$\mathsf{Adv}^{\text{SB-CPA}}_{\mathcal{A}_{\text{SB-CPA}}}(\lambda) \geqslant \frac{1}{|\mathbf{P}(\lambda)|^2} \cdot \mathsf{Adv}^{\text{indist}}_{H_{i-1},H_i,\mathcal{Z}_i}(\lambda).$$

Now remember that the premise of Theorem 2 is for Σ to satisfy IND-SB-CPA security. Hence there is a negligible function ε which bounds the advantage of any IND-SB-CPA adversary against Σ, i.e.

$$\mathsf{Adv}^{\text{indist}}_{H_{i-1},H_i,\mathcal{Z}_i}(\lambda) \leqslant |\mathbf{P}(\lambda)|^2 \cdot \mathsf{Adv}^{\text{SB-CPA}}_{\mathcal{A}_{\text{SB-CPA}}}(\lambda) \leqslant |\mathbf{P}(\lambda)|^2 \cdot \varepsilon(\lambda).$$

Not only does this prove the success probability of $\mathcal{Z}_i$ to be negligible and H_{i-1} and H_i to be computationally indistinguishable. The upper bound $|\mathbf{P}(\lambda)|^2 \cdot \varepsilon(\lambda)$ is also independent of i which by Lemma 2 means that hybrids H_0 and $H_{\mathsf{p}(\lambda)}$ are computationally indistinguishable as well for any polynomial p.

Now let $\mathcal{Z}$ be an arbitrary PPT environment trying to distinguish executions

$$\mathsf{EXEC}\left(\pi^{\mathcal{F}_{\text{AUTH}}}_{\text{M-SMT}}, \mathcal{D}\right) = H_0 \quad \text{and} \quad \mathsf{EXEC}(\mathcal{F}_{\text{M-SMT}}, \mathcal{S}_{\text{M-SMT}}).$$

There exists a polynomial p in the security parameter λ which bounds the runtime of the environment $\mathcal{Z}$, i.e. executions of hybrid $H_{\mathsf{p}(\lambda)}$ and the ideal world $\mathsf{EXEC}(\mathcal{F}_{\text{M-SMT}}, \mathcal{S}_{\text{M-SMT}})$ are trivially perfectly indistinguishable for $\mathcal{Z}$ as they do not differ before the $\mathsf{p}(\lambda)$-th message. Hence we know by above considerations that

$$\mathsf{EXEC}\left(\pi^{\mathcal{F}_{\text{AUTH}}}_{\text{M-SMT}}, \mathcal{D}\right) \sim_{\mathcal{Z}} H_{\mathsf{p}(\lambda)} \sim_{\mathcal{Z}} \mathsf{EXEC}(\mathcal{F}_{\text{M-SMT}}, \mathcal{S}_{\text{M-SMT}}),$$

which by finite transitivity of indistinguishability concludes our proof. □

3.4 Optimal Weakness

In the last section we have seen that IND-SB-CPA secure sender-binding encryption is sufficient to UC-realise secure message transfer via authenticated channels. In this section we complete the picture by proving that IND-SB-CPA security is also a *necessary* condition for a generalised public-key encryption scheme to facilitate secure message transfer from authenticated channels. At least if we restrict ourselves to the generally used encrypt-then-authenticate principle where messages are firstly encrypted before the resulting encryptions are sent over authenticated channels.

3.4.1 Generalised Public-key Encryption

Before we get to the generic encrypt-then-authenticate protocol, let me formally define what constitutes a generalised public-key encryption scheme. In contrast to the protocol in Section 3.3.1 we want this to allow for arbitrary types of encryption schemes—particularly including classic public-key encryption, tag-based and sender-binding encryption. The aim is to find a general definition which unifies all conceivable forms of different public-key encryption schemes. We therefore augment the input of the public-key encryption scheme's encryption and decryption algorithms by an auxiliary input. There are no restrictions on this auxiliary input other than, of course, that the encrypting/decrypting parties need to know it at the time of encryption/decryption. In particular, the auxiliary input may be variable and depend on the complete history of the party who uses the scheme. We model this by letting the auxiliary input *aux* take the form of

$$aux := f_{aux}(v_P)$$

where f_{aux} is a fixed polynomial-time function and v_P is the internal memory content of party P at the time of computation. We call v_P the "view of party P". v_P may in particular contain information on whether party P conducts an encryption or decryption and thus can lead to completely different *aux* values for the two algorithms. Note also, that although we model f_{aux} as deterministic this does not constitute any loss of generality. All probabilistic components can simply be pushed into the encryption and decryption algorithms themselves.

Definition (Generalized PKE):
A *generalised public-key encryption (PKE) scheme* $\Sigma = (\mathtt{gen}, \mathtt{enc}, \mathtt{dec})$ with message space $\mathbf{M}$ consists of three PPT algorithms:

$$\begin{array}{lr} \mathtt{gen}: & 1^\lambda \mapsto (sk, pk) \\ \mathtt{enc}: & (pk, aux, m) \mapsto c \\ \mathtt{dec}: & (sk, aux, c) \mapsto m \end{array}$$

As always, on input of the security parameter λ, the key generation algorithm gen generates a pair of secret and public keys *sk* and *pk*. The encryption algorithm enc and decryption algorithm dec also follow the definition of a classic public-key encryption scheme but take an additional auxiliary input of the form

$$aux := f_{aux}(v_P),$$

where v_P is the current internal memory content of the party P which conducts the algorithm and f_{aux} is a fixed deterministic polynomial-time function. ○

We do not demand any form of correctness at this point. For our special purpose in Section 3.4.3, correctness will follow from the realisation assumption. For more conventional use cases, though, a correctness property might need to be added to the definition.

The generalised public-key encryption definition might seem like a very small change from classic public-key encryption but it allows for a broad range of different encryption schemes. The only restrictions are:

- Both encryption and decryption take a key as input. This might even be the same, allowing for symmetric rather than public key encryption[7].
- Encryption takes the message as input.
- Decryption takes some (arbitrarily formed) output of the encryption as input.

Apart from that, algorithms, inputs and outputs can take any form they want.

3.4.2 Generic Encrypt-then-Authenticate Protocol

Let (`gen`, `enc`, `dec`) be a generalised public-key encryption scheme and assume the presence of authenticated channels. From these two building blocks the general *encrypt-then-authenticate* principle works as follows: Whenever a party S wants to securely transmit a message m to some party R, they first encrypt the message with the given generalised public-key encryption scheme before sending the resulting ciphertext c over the authenticated channel to R. When a party R receives a ciphertext c over an authenticated channel from some party S, they decrypt it with the generalised public-key encryption scheme again.

For key distribution we somewhat arbitrarily follow the same details as in Section 3.3, where credentials are generated on demand: Whenever a party needs public credentials of another party they send an `init` message over the authenticated channel. When a party receives such a message and has not yet generated credentials, they do so. As soon as credentials are generated, their public part is sent to all other parties via authenticated channel. Note that other types of credential generation and distribution would be just as feasible and only lead to some slight differences in the proof.

Let me give a formal definition of the encrypt-then-authenticate protocol π_{ETA} by presenting it in the UC context (i.e. with an interface to an environment $\mathcal{Z}$) in the $\mathcal{F}_{\text{AUTH}}$-hybrid model:

$\pi_{\textbf{ETA}}$

Realises:
Multi-receiver multi-message multi-sender secure message transfer with message space $\mathbf{M}$ and polynomially many parties $P \in \mathbf{P}$.

[7] Although, of course, symmetric encryption should not be used with our exact encrypt-then-authenticate protocol as the key would be openly sent to all other parties—but the restrictions on a generalised public-key encryption scheme would still allow you to do so.

Parameters:

- Generalised public-key encryption scheme $(\texttt{gen}, \texttt{enc}, \texttt{dec})$ with message space $\mathbf{M}$.
- Ideal authenticated channel functionality $\mathcal{F}_{\text{AUTH}}$.

State of Party P:

- Personal PKE key pair $(sk_P, pk_P) \in \mathbf{SK} \times \mathbf{PK}$.
- Partial function $\mathtt{f}_{\mathbf{PK}} : \mathbf{P} \to \mathbf{PK}$ of known public keys.
- Function $\mathrm{p}_{\mathrm{msg}} : \mathbf{P} \to \mathbf{M}^*$ of pending messages.

Behaviour of Party P:

```
\\ Being asked to initialise
• Upon receiving output $(\texttt{sent}, S, P, \texttt{init})$ from $\mathcal{F}_{\text{AUTH}}$, if there is no personal key pair $(sk_P, pk_P)$ yet:
    (1) $(sk, pk) \leftarrow \texttt{gen}(1^\lambda)$.
    (2) Set $(sk_P, pk_P)$ to $(sk, pk)$.
    (3) For each party $P' \neq P$: Hand input $(\texttt{send}, P', (\texttt{inited}, pk))$ to a new instance of $\mathcal{F}_{\text{AUTH}}$.
\\ Receiving keys and sending stored messages
• Upon receiving output $(\texttt{sent}, P', P, (\texttt{inited}, pk_{P'}))$ from $\mathcal{F}_{\text{AUTH}}$, if there is no entry $\mathtt{f}_{\mathbf{PK}}(P')$ yet:
    (1) Append $P' \mapsto pk_{P'}$ to $\mathtt{f}_{\mathbf{PK}}$.
    (2) For any $m \in \mathrm{p}_{\mathrm{msg}}(P')$:
        (1) Remove $m$ from $\mathrm{p}_{\mathrm{msg}}(P')$.
        (2) $aux := f_{aux}(v_P)$.
        (3) $c \leftarrow \texttt{enc}(pk_{P'}, aux, m)$.
        (4) Hand input $(\texttt{send}, P', c)$ to a new instance of $\mathcal{F}_{\text{AUTH}}$.
\\ Sending messages
• Upon receiving input $(\texttt{send}, R, m)$ with $m \in \mathbf{M}$ from environment $\mathcal{Z}$:
    ◦ If $R = P$ report output $(\texttt{sent}, P, m)$ to the environment.
    ◦ Else if no entry $\mathtt{f}_{\mathbf{PK}}(R)$ exists yet:
        (1) Append $m$ to $\mathrm{p}_{\mathrm{msg}}(R)$.
        (2) Hand input $(\texttt{send}, R, \texttt{init})$ to a new instance of $\mathcal{F}_{\text{AUTH}}$.
    ◦ Else:
        (1) $pk_R := \mathtt{f}_{\mathbf{PK}}(R)$.
        (2) $aux := f_{aux}(v_P)$.
        (3) $c \leftarrow \texttt{enc}(pk_R, aux, m)$.
        (4) Hand input $(\texttt{send}, R, c)$ to a new instance of $\mathcal{F}_{\text{AUTH}}$.
```

\\ Receiving messages

- Upon receiving output $(\texttt{sent}, S, R, c)$ from $\mathcal{F}_{\text{AUTH}}$:
  (1) $aux := f_{aux}(v_P)$.
  (2) $m \leftarrow \texttt{dec}(sk, aux, c)$.
  (3) Report output $(\texttt{sent}, S, m)$ to the environment $\mathcal{Z}$.

Looking back at the prior efforts and our own UC realisation in Sections 3.1 and 3.3, respectively, we see that all efforts to realise secure message transfer from $\mathcal{F}_{\text{AUTH}}$ and some form of encryption follow this encrypt-then-authenticate principle. To see this, any changes to the underlying encryption scheme—like concatenating the sender ID to the message for IND-CCA2 secure encryption or choosing it as the tag for tag-based encryption—are encapsulated inside the encryption scheme $(\texttt{gen}, \texttt{enc}, \texttt{dec})$ that is used within π_{ETA}.

3.4.3 Proving the Optimal Weakness of IND-SB-CPA

With the protocol π_{ETA} we now have all the tools we need to prove that IND-SB-CPA is indeed the weakest possible notion to facilitate secure message transfer from authenticated channels and encryption via the encrypt-then-authenticate principle:

Theorem 3:
Let $\Sigma = (\texttt{gen}, \texttt{enc}, \texttt{dec})$ be a generalised public-key encryption scheme such that

$$\pi_{ETA}^{\mathcal{F}_{AUTH}} \geqslant_{UC} \mathcal{F}_{M\text{-}SMT}$$

holds under arbitrary malicious but static corruption. Then Σ satisfies IND-SB-CPA security.

Proof:
Because this proof this rather long and detailed, I have broken it down into many individual steps which I would like to explain before starting the actual proof. We will successively prove the following statements:

(1) Any public-key encryption scheme Σ which leads to π_{ETA} satisfying

$$\pi_{\text{ETA}}^{\mathcal{F}_{\text{AUTH}}} \geqslant_{\text{UC}} \mathcal{F}_{\text{M-SMT}},$$

must *satisfy correctness* within π_{ETA}. This means if a message m is encrypted, sent, and subsequently decrypted by the receiving party, decryption recovers the original message m with overwhelming probability.

(2) Any public-key encryption scheme Σ which leads to π_{ETA} satisfying

$$\pi_{\text{ETA}}^{\mathcal{F}_{\text{AUTH}}} \geqslant_{\text{UC}} \mathcal{F}_{\text{M-SMT}},$$

must *be a sender-binding encryption scheme*, i.e. completely identify the sender's ID in both the encryption and decryption auxiliary input *aux*. We show this by contradiction, ...

   (a) Firstly, assuming that the auxiliary *encryption* input only depends on sender, receiver and message (S, R, m) rather than any history of the protocol.
   (b) Secondly, relaxing this assumption to an arbitrary *encryption* input $aux = f_{aux}(v_S)$.
   (c) And thirdly, for the auxiliary *decryption* input $aux = f_{aux}(v_S)$.

(3) Any public-key encryption scheme Σ which leads to π_{ETA} satisfying

$$\pi_{\mathrm{ETA}}^{\mathcal{F}_{\mathrm{AUTH}}} \geqslant_{\mathrm{UC}} \mathcal{F}_{\text{M-SMT}},$$

must *be IND-SB-CPA secure*. We again show this by contradiction. Assuming Σ is not IND-SB-CPA secure we show that π_{ETA} does not UC-realise $\mathcal{F}_{\text{M-SMT}}$ either, by...
   (a) Constructing an environment $\mathcal{Z}$ from a (non-negligibly) successful adversary.
   (b) Showing this environment's success probability to be non-negligible as well.

Let us begin with the actual proof:

(1) Putting the correctness property in formal terms, it means that for all environments $\mathcal{Z}$, the probability of an execution

$$\mathsf{EXEC}\left(\pi_{\mathrm{ETA}}^{\mathcal{F}_{\mathrm{AUTH}}}, \mathcal{D}\right)$$

where $\mathcal{Z}$ has a message m sent from some honest party S to another honest party R such that

$$m' \leftarrow \mathtt{dec}\left[sk_R, f_{aux}(v_R), \mathtt{enc}\left(pk_R, f_{aux}(v_S), m\right)\right] \neq m$$

is negligible, when v_S is the view of party S at the time of encryption of m and v_R the view of party R at the time of decryption of $\mathtt{enc}(pk_R, f_{aux}(v_S), m)$. If this was not the case, then $\mathcal{Z}$ would be able to distinguish real protocol and ideal experiment

$$\mathsf{EXEC}\left(\pi_{\mathrm{ETA}}^{\mathcal{F}_{\mathrm{AUTH}}}, \mathcal{D}\right) \not\approx_{\mathcal{Z}} \mathsf{EXEC}(\mathcal{F}_{\text{M-SMT}}, \mathcal{S})$$

since for this non-negligibly probable execution R would output receiving message m from S in the ideal experiment while it outputs $m' \neq m$ in the real world. This is a contradiction to the premise that

$$\pi_{\mathrm{ETA}}^{\mathcal{F}_{\mathrm{AUTH}}} \geqslant_{\mathrm{UC}} \mathcal{F}_{\text{M-SMT}},$$

hence we know that correctness must hold.

(2) We now look at why Σ must be a sender-binding encryption scheme within the confines of π_{ETA} to enable realisation of $\mathcal{F}_{\text{M-SMT}}$. The difficulty at this point is that Σ does not explicitly need to identify parties to do so. A Person, for instance, could not only be uniquely identified by their ID card, but also by information like their name, date and place of birth, fingerprint, visual appearance or number plate. This means we can not just prove that the auxiliary input *aux* contains the sender's party ID—because it does not have to *explicitly* contain it. Instead we show that *aux* contains information that is (with overwhelming probability) a unique identifier of the sending party.
The idea behind the following three proof steps is always the same: If an auxiliary input *aux* did not fully identify sender S, the ciphertext created by S could be sent from some other party without the receiver noticing that something is wrong. Since this is not possible in the ideal experiment without knowledge of the message content, it constitutes a contradiction to π_{ETA} UC-realising $\mathcal{F}_{\text{M-SMT}}$.

(a) We firstly assume that the auxiliary input used for *encryption* is constant for every message triple (S, R, m) of sender, receiver and message, i.e. that it does not depend on the history of S but can be defined as

$$f_{aux}(v_S) := f'_{aux}(S, R, m).$$

Now assume this auxiliary input does not fully identify S. I.e. that there are a receiver R and a message m such that the value $f'_{aux}(S, R, m)$ is not unique for every party S. Instead there are parties $S_1 \neq S_2 \in \mathbf{P}$ such that

$$f'_{aux}(S_1, R, m) = f'_{aux}(S_2, R, m).$$

We will use this assumption to derive a contradiction to our premise by constructing a distinguishing environment $\mathcal{Z}$. Let $\mathcal{Z}$ corrupt party S_2. Furthermore $\mathcal{Z}$ asks S_1 to send message m to receiver R but instructs the adversary $\mathcal{A}_{\mathrm{UC}} \in \{\mathcal{D}, \mathcal{S}\}$ to report back the authenticated channel content

$$c = \mathtt{enc}\Big(pk_R, m, f'_{aux}(S_1, R, m)\Big)$$

and block the actual delivery to R. Now $\mathcal{Z}$ has the adversary send c from corrupted party S_2 to R instead without blocking delivery.
Suppose this was conducted with the real protocol π_{ETA}. Since $f'_{aux}(S_1, R, m) = f'_{aux}(S_2, R, m)$, we also have

$$c \leftarrow \mathtt{enc}\Big(pk_R, f'_{aux}(S_1, R, m), m\Big) \sim \mathtt{enc}\Big(pk_R, f'_{aux}(S_2, R, m), m\Big).$$

This means the input $(sk_R, f_{aux}(v_R), c)$ for the decryption R conducts of c is indistinguishable from what it were if S_2 had honestly encrypted message m to gain a ciphertext for R. By the correctness property shown in (1), this means that with overwhelming probability

$$\begin{aligned} &\mathtt{dec}\Big[sk_R, f_{aux}(v_R), c\Big] \\ =\ &\mathtt{dec}\Big[sk_R, f_{aux}(v_R), \mathtt{enc}\Big(pk_R, f'_{aux}(S_2, R, m), m\Big)\Big] \\ =\ &m \end{aligned}$$

and R will output that it was sent message m by party S_2. In the ideal experiment on the other hand, the view of the simulator $\mathcal{S}$ and hence also the "ciphertext" c is independent of the message content m. Hence the simulator has no way of getting honest party R to output the unknown message m. This discrepancy between the real and ideal world is a contradiction to our premise and the assumption must be false. I.e. if

$$f_{aux}(v_S) := f'_{aux}(S, R, m)$$

used during encryption is constant for every message triple (S, R, m), it uniquely identifies the sending party S.

(b) Now if $f_{aux}(v_S)$ is not constant but *does* depend on the history of S the case is slightly more complicated. The key difference is that the distinguishing environment $\mathcal{Z}$ now needs to firstly set up the right history for the parties as well. So let us again assume that $f_{aux}(v_S)$ used during *encryption* does not uniquely identify party S with overwhelming probability. This means there are parties $S_1 \neq S_2$ and R, a message m and a polynomial-length sequence η of encryptions and a subset of corresponding decryptions that with non-negligible probability results in views v'_1 and v'_2 of S_1 and S_2 respectively such that
$$f_{aux}(v_1^*) = f_{aux}(v_2^*),$$
where v_i^* is gained from v'_i by appending the instruction to encrypt message m for R. We again derive a contradiction by constructing a distinguishing environment $\mathcal{Z}$.
$\mathcal{Z}$ corrupts party S_2 and then firstly gives instructions to all parties according to the encryption/decryption sequence η—including instructions to adversary $\mathcal{A}_{\text{UC}} \in \{\mathcal{D}, \mathcal{S}\}$ to block/delay messages such that decryptions happen at the right point in time or even never at all if they do not appear in η. After completing sequence η, $\mathcal{Z}$ continues just as in case (a): $\mathcal{Z}$ asks S_1 to send message m to receiver R but instructs the adversary $\mathcal{A}_{\text{UC}}$ to report back the authenticated channel content c and block the actual delivery to R. Now $\mathcal{Z}$ has the adversary send c from corrupted party S_2 to R instead without blocking delivery.
Suppose this was conducted with the real protocol π_{ETA}. Then with non-negligible probability S_1 and S_2 have views v_1^* and v_2^* and
$$c \leftarrow \texttt{enc}\Big(pk_R, f_{aux}(v_1^*), m\Big) \sim \texttt{enc}\Big(pk_R, f_{aux}(v_2^*), m\Big)$$
by assumption. This again means that R's decryption input is indistinguishable from what it were if S_2 had honestly encrypted message m to gain a ciphertext for R. Hence R will output that it was sent message m by party S_2 in the real protocol while the simulator has no way of gaining this output in the ideal experiment. This discrepancy between the real and ideal world is a contradiction to our premise and the assumption must be false, i.e. $f_{aux}(v_S)$ used during encryption fully identifies the sender S.

(c) Having seen that the auxiliary input $f_{aux}(v_S)$ during encryption uniquely identifies the encrypting party S, it remains to show that the input $f_{aux}(v_R)$ for *decryption* also uniquely identifies the identity of the sending party. Since we use the same general principle as in (2a) and (2b), we will directly consider the case where $f_{aux}(v_R)$ may depend on the history of R. So assume $f_{aux}(v_R)$ during decryption of a message received from party S does *not* uniquely identify party S with overwhelming probability. This means there are parties $S_1 \neq S_2$ and R, a message m and a polynomial-length sequence η of encryptions and a subset of corresponding decryptions that with non-negligible probability results in views v'_S of S_1 and v'_R of R such that with non-negligible probability encryption of m via
$$c \leftarrow \texttt{enc}\Big(pk_R, f_{aux}(v_S^*), m\Big)$$
yields a ciphertext c such that
$$f_{aux}(v_1^*) = f_{aux}(v_2^*),$$

where v_S^* is gained from v_S' by appending the instruction to encrypt message m for R and v_i^* is gained from v_R' by appending the reception of c from party S_i. We can now derive a contradiction by constructing a distinguishing environment $\mathcal{Z}$, just as we did in (2b):
$\mathcal{Z}$ again corrupts party S_2 and then firstly gives instructions to all parties according to the encryption/decryption sequence η—including instructions to adversary $\mathcal{A}_{\text{UC}} \in \{\mathcal{D}, \mathcal{S}\}$ to block/delay messages such that decryptions happen at the right point in time or even never at all if they do not appear in η. After completing sequence η, $\mathcal{Z}$ asks S_1 to send message m to receiver R and gains the corresponding channel content c from $\mathcal{A}_{\text{UC}} \in \{\mathcal{D}, \mathcal{S}\}$, but instructs $\mathcal{A}_{\text{UC}}$ to block delivery of c to R. Then $\mathcal{Z}$ has the adversary send c from corrupted party S_2 to R instead without blocking delivery.
Suppose this was conducted with the real protocol π_{ETA}. Then with non-negligible probability S_1 has view v_S^* and R has view v_2^* such that

$$f_{aux}(v_1^*) = f_{aux}(v_2^*)$$

by assumption. This means that R's decryption input is indistinguishable from what it were if S_1 had sent ciphertext c to R instead of S_2. Hence R will output that it was sent message m by party S_2 in the real protocol while the simulator could not gain this output in the ideal experiment. This discrepancy between the real and ideal world is a contradiction to our premise and the assumption must be false, i.e. $f_{aux}(v_R)$ used during decryption also fully identifies the sender S.

Together (2a)-(2c) assert that within π_{ETA}, the auxiliary inputs $f_{aux}(v_P)$ always uniquely identify the encrypting/sending party's identity. This means that Σ is at least a sender-binding encryption scheme, but could furthermore include other properties allowed by auxiliary inputs.

(3) Now that we know this we can go on to reduce the IND-SB-CPA security of Σ to the indistinguishability of π_{ETA} and $\mathcal{F}_{\text{M-SMT}}$: Assuming Σ is *not* IND-SB-CPA secure, we show that

$$\pi_{\text{ETA}}^{\mathcal{F}_{\text{AUTH}}} \geqslant_{\text{UC}} \mathcal{F}_{\text{M-SMT}}$$

does not hold either. Hence let $\mathcal{A}_{\text{SB-CPA}}$ be an IND-SB-CPA adversary who has non-negligible advantage

$$\rho := \mathbb{P}[\mathcal{A}_{\text{SB-CPA}} \text{ successful}] - \tfrac{1}{2} > 0$$

to win the IND-SB-CPA game.

(a) Given adversary $\mathcal{A}_{\text{SB-CPA}}$, we construct an environment $\mathcal{Z} = \mathcal{Z}(\mathcal{A}_{\text{SB-CPA}})$ which can distinguish

$$\mathsf{EXEC}\left(\pi_{\text{ETA}}^{\mathcal{F}_{\text{AUTH}}}, \mathcal{D}\right) \not\approx_{\mathcal{Z}} \mathsf{EXEC}(\mathcal{F}_{\text{M-SMT}}, \mathcal{S})$$

for dummy adversary $\mathcal{D}$ and *any* simulator $\mathcal{S}$. Note that "any simulator" includes the possibility of $\mathcal{S}$ being dependent on $\mathcal{A}_{\text{SB-CPA}}$—even to the extent that it might have perfect knowledge about $\mathcal{A}_{\text{SB-CPA}}$.

For the definition of the environment $\mathcal{Z}$ we again denote the adversary which it interacts with in the UC experiment by $\mathcal{A}_{\mathrm{UC}} \in \{\mathcal{D}, \mathcal{S}\}$. Note furthermore that for the sake of brevity, we will not have $\mathcal{Z}$ give output

$$\mathsf{EXEC}\left(\pi_{\mathrm{ETA}}^{\mathcal{F}_{\mathrm{AUTH}}}, \mathcal{D}\right) \quad \text{or} \quad \mathsf{EXEC}(\mathcal{F}_{\text{M-SMT}}, \mathcal{S})$$

but instead identify these formal outputs with the more descriptive terms of "REAL" and "IDEAL", respectively. Now let $\mathcal{Z}$ behave according to the following instructions. Note that by $(*)$ we denote responses which are handled in this way at any point in time, $\|\!:\!(\_)\!:\!\|$ denotes actions which are repeated as often as they get prompted and (Ω) denotes a final action which causes $\mathcal{Z}$ to halt afterwards.

$(*)$ As a general rule: Whenever the adversary $\mathcal{A}_{\mathrm{UC}}$ reports input

$$(\texttt{send}, P, P', m)$$

from $\mathcal{F}_{\mathrm{AUTH}}$, immediately ask adversary $\mathcal{A}_{\mathrm{UC}}$ to allow transmission by answering $(\texttt{send ok})$.

(1) Draw random challenge parties $S, R \xleftarrow{\$} \mathbf{P}$ from the set of all parties $\mathbf{P}$, corrupt all other parties $P \in \mathbf{P} \setminus \{S, R\}$ and start the UC experiment[8].

Ask the adversary $\mathcal{A}_{\mathrm{UC}}$ to provide inputs

$$(\texttt{send}, S, \texttt{init}) \text{ and}$$
$$(\texttt{send}, R, \texttt{init})$$

to new instances of $\mathcal{F}_{\mathrm{AUTH}}$ in the name of some corrupted party P. Receive pk_S, pk_R in response and start $\mathcal{A}_{\text{SB-CPA}}$ in the head with input (S, pk_S, R, pk_R).

$\|\!:\!(2)\!:\!\|$ Upon receiving oracle query $(pk_{R'}, S', c)$ from $\mathcal{A}_{\text{SB-CPA}}$, check that $pk_{R'} \in \{pk_S, pk_R\}$ and $S' \notin \{S, R\}$. If any check fails report back $m := \bot$. If the checks pass, ask the adversary $\mathcal{A}_{\mathrm{UC}}$ to provide input

$$(\texttt{send}, R', c)$$

to a new instance of $\mathcal{F}_{\mathrm{AUTH}}$ in the name of S'. Note that S' is corrupted. Upon output $(\texttt{sent}, S', m)$ from party R', report m back to $\mathcal{A}_{\text{SB-CPA}}$. An overview of how $\mathcal{Z}$ handles oracle queries for the example of $R' = R$ can be found in Figure 12.

(3) Upon receiving challenge messages m_0, m_1 from $\mathcal{A}_{\text{SB-CPA}}$ draw random bit $b^* \xleftarrow{\$} \{0, 1\}$ and give input

$$(\texttt{send}, R, m_b^*)$$

to party S. When the UC adversary subsequently reports channel content

$$(\texttt{send}, S, R, c^*)$$

hand challenge c^* back to $\mathcal{A}_{\text{SB-CPA}}$. An overview of how the challenge is created is shown in Figure 13.

[8] Note that we need at least three parties to exists in this universe. This is by no means a significant constraint, as—of course—two parties can always communicate securely if they are the only ones in existence.

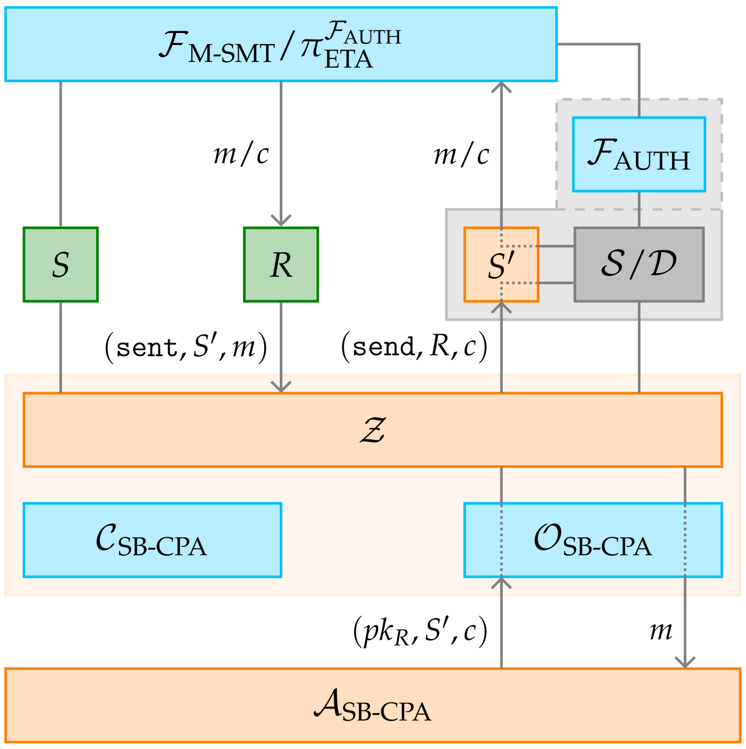

$\mathcal{F}_{\text{M-SMT}} / \pi_{\text{ETA}}^{\mathcal{F}_{\text{AUTH}}}$
m/c
m/c
$\mathcal{F}_{\text{AUTH}}$
S
R
S'
$\mathcal{S}/\mathcal{D}$
$(\texttt{sent}, S', m)$
$(\texttt{send}, R, c)$
$\mathcal{Z}$
$\mathcal{C}_{\text{SB-CPA}}$
$\mathcal{O}_{\text{SB-CPA}}$
(pk_R, S', c)
m
$\mathcal{A}_{\text{SB-CPA}}$

Figure 12: Environment $\mathcal{Z}$ Handling Oracle Queries

‖:(4):‖ Conduct the second oracle phase just like the first: Upon receiving oracle query $(pk_{R'}, S', c)$ from $\mathcal{A}_{\text{SB-CPA}}$, check that $pk_{R'} \in \{pk_S, pk_R\}$ and $S' \notin \{S, R\}$. If any check fails report back $m := \perp$. If the checks pass ask the adversary $\mathcal{A}_{\text{UC}}$ to hand input

$$(\texttt{send}, R', c)$$

to a new instance of $\mathcal{F}_{\text{AUTH}}$ in the name of S'. Upon output $(\texttt{sent}, S', m)$ from party R', report m back to $\mathcal{A}_{\text{SB-CPA}}$.

(5) If $\mathcal{A}_{\text{SB-CPA}}$ reports bit $b = b^*$, output REAL, otherwise if $b \neq b^*$ output IDEAL.

(Ω) If $\mathcal{A}_{\text{SB-CPA}}$ should halt without answering, flip a fair coin on whether to output REAL or IDEAL.

(b) Let us evaluate the success probability of $\mathcal{Z}$: First, consider the case the UC experiment is conducted in the real world with protocol π_{ETA}. I.e. $\mathcal{Z}$ interacts with dummy adversary $\mathcal{D}$ and parties S, R running the real protocol. In this case the interaction between $\mathcal{Z}$ and $\mathcal{A}_{\text{SB-CPA}}$ exactly corresponds to the IND-SB-CPA game. As $\mathcal{Z}$ outputs REAL if $\mathcal{A}_{\text{SB-CPA}}$ successfully outputs $b = b^*$ and also in half of the cases where $\mathcal{A}_{\text{SB-CPA}}$ does not output anything, the environment has a success probability of at least

$$\begin{aligned}\mathbb{P}\big[\mathsf{REAL} \leftarrow \mathcal{Z} \bigm| \mathsf{REAL}\big] &\geqslant \mathbb{P}\big[b = b^* \bigm| b \leftarrow \mathcal{A}_{\text{SB-CPA}}\big] \\ &= \tfrac{1}{2} + \rho.\end{aligned}$$

Secondly, let the UC experiment be conducted with $\mathcal{F}_{\text{M-SMT}}$ in the ideal world. Here, $\mathcal{Z}$ interacts with dummy parties S, R and a simulator $\mathcal{S}$. This case requires some more

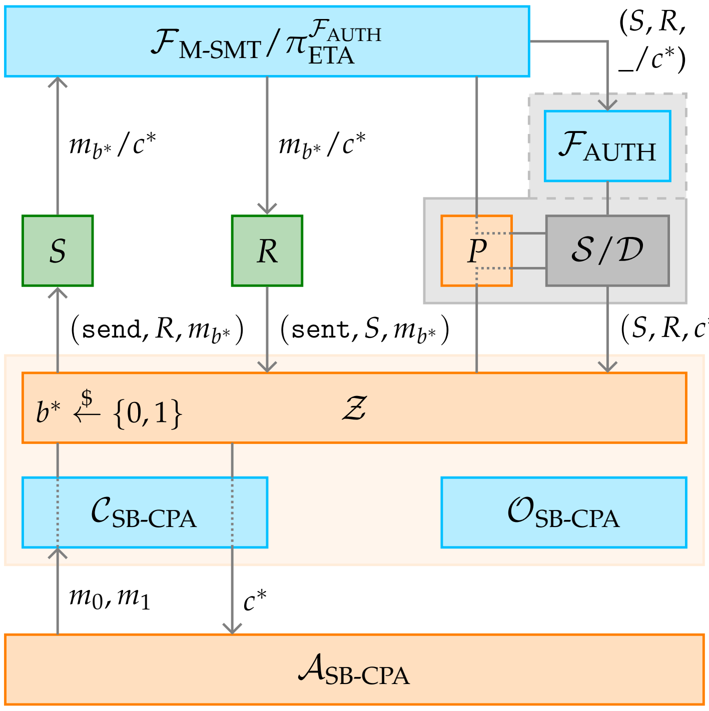

$\mathcal{F}_{\text{M-SMT}} / \pi_{\text{ETA}}^{\mathcal{F}_{\text{AUTH}}}$
$(S, R, \_/c^*)$
$\mathcal{F}_{\text{AUTH}}$
m_{b^*}/c^*
m_{b^*}/c^*
S
R
P
$\mathcal{S}/\mathcal{D}$
$(\texttt{send}, R, m_{b^*})$
$(\texttt{sent}, S, m_{b^*})$
(S, R, c^*)
$b^* \xleftarrow{\$} \{0,1\}$
$\mathcal{Z}$
$\mathcal{C}_{\text{SB-CPA}}$
$\mathcal{O}_{\text{SB-CPA}}$
m_0, m_1
c^*
$\mathcal{A}_{\text{SB-CPA}}$

Figure 13: Environment $\mathcal{Z}$ Creating Challenge Ciphertext

consideration. As neither public keys of S and R nor answers to oracle queries or the challenge ciphertext c^* need to be constructed according to Σ and the IND-SB-CPA game, we have no guarantees on the behaviour of $\mathcal{A}_{\text{SB-CPA}}$. In a first step, we assume $\mathcal{A}_{\text{SB-CPA}}$ to nevertheless continue the game and output a bit b upon termination. Now the important point to notice here is that the view of the simulator $\mathcal{S}$ is information theoretically independent from the challenge bit b^*: $\mathcal{S}$ only receives the same output

$$(\texttt{send}, mid, S, R)$$

from the ideal functionality $\mathcal{F}_{\text{M-SMT}}$—regardless of which message was sent. This means even if $\mathcal{S}$ might have perfect knowledge about $\mathcal{A}_{\text{SB-CPA}}$ and therefore know the challenge messages m_0 and m_1 the challenge ciphertext c^* which $\mathcal{S}$ generates is also independent of b^*. Hence the view of $\mathcal{A}_{\text{SB-CPA}}$ will be information theoretically independent of b^* as well—yielding a perfectly level success probability of

$$\mathbb{P}\big[b = b^* \bigm| b \leftarrow \mathcal{A}_{\text{SB-CPA}}\big] = \tfrac{1}{2}$$

in this case. Together with $\mathcal{Z}$'s coin flip in case $\mathcal{A}_{\text{SB-CPA}}$ should abort the game and not output a bit b we have

$$\mathbb{P}\big[\mathsf{IDEAL} \leftarrow \mathcal{Z} \bigm| \mathsf{IDEAL}\big] = \tfrac{1}{2}.$$

This yields the overall success probability of

$$\begin{aligned}\mathbb{P}\big[\mathcal{Z} \text{ successful}\big] &= \mathbb{P}\big[\mathsf{REAL}\big] \cdot \mathbb{P}\big[\mathsf{REAL} \leftarrow \mathcal{Z} \bigm| \mathsf{REAL}\big] \\ &\quad + \mathbb{P}\big[\mathsf{IDEAL}\big] \cdot \mathbb{P}\big[\mathsf{IDEAL} \leftarrow \mathcal{Z} \bigm| \mathsf{IDEAL}\big] \\ &\geqslant \tfrac{1}{2} \cdot (\tfrac{1}{2} + \rho) + \tfrac{1}{2} \cdot \tfrac{1}{2} \\ &= \tfrac{1}{2} + \tfrac{1}{2}\rho,\end{aligned}$$

which is non-negligibly better than guessing.

Hence we have a contradiction to

$$\pi_{\text{ETA}}^{\mathcal{F}_{\text{AUTH}}}(\Sigma) \geqslant_{\text{UC}} \mathcal{F}_{\text{M-SMT}}$$

which means the initial assumption that Σ might not be IND-SB-CPA secure can not hold. This concludes the proof of Theorem 3. $\square$

In this chapter we set out to find the weakest public-key encryption security notion in order to establish multiple secure message transfer from existing authenticated channels in the UC model. After considering the headway made by previous efforts, I have introduced and discussed the new concepts of sender-binding encryption and IND-SB-CPA security. We have seen that IND-SB-CPA secure sender-binding encryption is indeed strong enough to UC-realise the secure message transfer functionality $\mathcal{F}_{\text{M-SMT}}$ under static corruption from ideal authenticated channels. Furthermore, we found that under restriction to the encrypt-then-authenticate principle the reverse is also true, making IND-SB-CPA secure sender-binding encryption the optimally weak public-key encryption concept in this case.

My data is safe behind six layers of symmetric and public-key algorithms.

Randall Munroe, xkcd #1269

CHAPTER 4

Sender-binding Key Encapsulation

We have seen in the last chapter, that sender-binding encryption is the right public-key encryption concept to construct secure communication from authenticated channels and that IND-SB-CPA security is in some sense the weakest security notion to realise secure message transfer from authenticated channels in the UC model. So far, however, we ignored that many real world applications have moved on to using hybrid instead of public-key encryption. This leads us to the second research question of this dissertation:

What are the weakest public-key key encapsulation mechanism and data encapsulation mechanism security notions in order to establish secure communication from existing authenticated channels?

Again I start by presenting related work and its prior efforts to answer this research question. To make further headway I then bring sender-binding ideas to the real world efficient KEM-DEM framework for hybrid encryption. This allows me to develop the concept of a sender-binding key encapsulation mechanism with corresponding IND-SB-CPA security in Section 4.2. I then go on to show that both in single-message and session communication settings, this new security notion is sufficiently strong to UC-realise secure communication from authenticated channels if paired with IND-OT and IND-CPA secure data encapsulation mechanisms, respectively. Those proofs can be found in Sections 4.3 and 4.4. The question whether IND-SB-CPA secure sender-binding key encapsulation constitutes the optimally weak notion in this context had to be left open at this point.

I have previously published the new definitions and insights from this chapter in [SBB+23b].

4.1 Prior Efforts

While there are lots of papers pertaining to the general topic of hybrid encryption via the KEM-DEM framework, most of the works focus on concrete efficient constructions of key encapsulation mechanisms, such as the hybrid encryption scheme by Cramer and Shoup [CS02], the Kurosawa-Desmedt-KEM [KD04] or the newly standardised Kyber-KEM [BDK+18]. For this section, however, we stay with the main contribution of my dissertation and instead consider those papers which give proofs on what levels of secure communication can be reached with various key and data encapsulation security notions.

As mentioned in Section 2.1.2, there are two branches of KEM-DEM-based hybrid encryption: Single-message and session communication. We start in the more common single-message setting, where each symmetric key is used for only one message. Unfortunately, the question of how strong (or weak) encryption should be for secure communication seems to have been largely ignored in this area. Instead, constructing IND-CCA2 secure public-key encryption is commonly seen as the only goal, regardless of the fact that in practice, encryption schemes are usually paired with authentication via digital signatures to gain secure communication. Hence the main security analysis in prior works is usually limited to constructing an IND-CCA2 secure public-key encryption scheme from successively weaker (and more efficient) key and data encapsulation notions.

When originally introducing the KEM-DEM framework, Shoup showed that combining key and data encapsulation mechanisms which satisfy the respective notions of IND-CCA2 security yields an IND-CCA2 secure public-key encryption as a result [Sho01; CS02]. Also in [CS02] it was shown that if one relaxes the security of the data encapsulation mechanism to one-time-IND-CCA2 security (sometimes called IND-OTCCA [HHK10]), the construction still suffices for an IND-CCA2 secure public-key encryption, as each symmetric key will only be used once. In [HHK10], Herranz, Hofheinz and Kiltz give an overview of all previously proposed game-based key and data encapsulation security notions and comprehensively identify which combinations lead to which security notions for the resulting public-key encryption. One main finding was that IND-CCA2 security could so far *only* be reached via IND-CCA2 secure key encapsulation in conjunction with (one-time-)IND-CCA2 secure data encapsulation. All other combinations result in less secure public-key encryption schemes. Kurosawa and Desmedt managed to present a KEM-DEM construction for an IND-CCA2 secure public-key encryption scheme in [KD04] where the employed key encapsulation mechanism is not IND-CCA2 secure [CHH+09]. However, it was shown in [KT14] that the Kurosawa-Desmedt-KEM is not far off, as it becomes IND-CCA2 secure with a slight twist. Abe et al. modify the KEM-DEM framework to a new tag-KEM-DEM framework [AGKS05] (cf. Section 2.1.2). They show that for this type of hybrid encryption, IND-CCA2 secure key encapsulation together with only IND-OT secure data encapsulation yields IND-CCA2 public-key encryption as well. They also show that the aforementioned Kurosawa-Desmedt-KEM can be considered a tag-based key encapsulation mechanism in which case it actually satisfies the tag-based key encapsulation version of IND-CCA2 security. A similar not quite IND-CCA2 secure key encapsulation mechanism was used in [BIL21] as well.

In contrast to these works we remember Chapter 3, where we saw that IND-CCA2 security is unnecessarily strong to realise secure communication and hence do not try to construct IND-CCA2 secure public-key encryption in this chapter. Aiming for the weaker but sufficient notion of IND-SB-CPA secure sender-binding encryption, I develop the corresponding key encapsulation notion of a sender-binding key encapsulation mechanism with corresponding IND-SB-CPA security. We see in Section 4.3 that in combination with the weakest possible data encapsulation—satisfying only IND-OT security—this new notion still yields IND-SB-CPA security for the sender-binding encryption scheme constructed via the classic KEM-DEM framework. Using such a weak data encapsulation mechanism was previously only possible via the more complex tag-KEM-DEM framework. Furthermore we show that if the sender-binding key encapsulation is viewed as a (simpler) version of tag-based key encapsulation, the key encapsulation version of IND-SB-CPA security is strictly weaker than the IND-CCA2 notion employed in the tag-KEM-DEM framework.

Although Canetti and Krawczyk consider various UC and non-UC security notions for key exchange and session key security in [CK01; CK02], Nagao, Manabe and Okamoto were the first to take the KEM-DEM framework into the world of UC security [NMO06]. They also make the switch to session communication where each symmetric key is used not only for multiple messages but bi-directional communication as well.

Nagao, Manabe and Okamoto firstly introduce an ideal functionality $\mathcal{F}_{\text{KEM}}$ capturing the security intuitively expected from key encapsulation. They prove a generic protocol to UC-realise $\mathcal{F}_{\text{KEM}}$ if and only if the key encapsulation mechanism used in the protocol is IND-CCA2 secure. In a second step a complete KEM-DEM functionality $\mathcal{F}_{\text{KEM-DEM}}$ is defined and similarly shown that it is UC-realised by a generic data encapsulation protocol in the $\mathcal{F}_{\text{KEM}}$-hybrid model if and only if the data encapsulation mechanism satisfies IND-CCA2 security. Lastly, it is shown that using $\mathcal{F}_{\text{KEM-DEM}}$ in conjunction with the signature and certification functionalities $\mathcal{F}_{\text{SIG}}$ and $\mathcal{F}_{\text{CA}}$ suffices without any other cryptographic building blocks to UC-realise a single-session bi-directional secure channel $\mathcal{F}_{\text{SC}}$. An overview of this process is shown in Figure 14.

$$\begin{tikzcd}
\boxed{\text{IND-CCA2 KEM}} \arrow[r, Leftrightarrow] & \boxed{\pi_{\text{KEM}} \geqslant_{\text{UC}} \mathcal{F}_{\text{KEM}}} \arrow[dr] & & \\
& \boxed{\text{IND-CCA2 DEM}} \arrow[r, Leftrightarrow] & \boxed{\pi_{\text{DEM}}^{\mathcal{F}_{\text{KEM}}} \geqslant_{\text{UC}} \mathcal{F}_{\text{KEM-DEM}}} \arrow[dr] & \\
& & & \boxed{\pi_{\mathcal{F}_{\text{SIG}},\mathcal{F}_{\text{CA}}}^{\mathcal{F}_{\text{KEM-DEM}}} \geqslant_{\text{UC}} \mathcal{F}_{\text{SC}}}
\end{tikzcd}$$

Figure 14: Overview of Secure Channel Realisation from [NMO06]

The additional functionalities $\mathcal{F}_{\text{SIG}}$ and $\mathcal{F}_{\text{CA}}$ are used for authentication during the key exchange and could equally be substituted by $\mathcal{F}_{\text{AUTH}}$, as it was shown in [Can04] that such a use of signatures combined with certification already UC-realises $\mathcal{F}_{\text{AUTH}}$. The two equivalences between respective IND-CCA2 security notions and ideal key encapsulation and KEM-DEM functionalities from [NMO06] could be taken to indicate that IND-CCA2 is a necessary condition for achieving secure channels via the session KEM-DEM framework. In Section 4.4 I prove that this is not actually true. The main point to note here is that authentication in the form of $\mathcal{F}_{\text{SIG}}$ and $\mathcal{F}_{\text{CA}}$ is only used for the key exchange and added after the fact to the KEM-DEM functionality to realise $\mathcal{F}_{\text{SC}}$. Directly using the key encapsulation mechanism on an authenticated channel and binding the key ciphertext to the sender lets us achieve the same level of security with significantly less security requirements on the key encapsulation mechanism. Hence for the proof in Section 4.4 we skip the detour via $\mathcal{F}_{\text{KEM}}$ and $\mathcal{F}_{\text{KEM-DEM}}$ and directly prove that an IND-SB-CPA secure key encapsulation mechanism combined with $\mathcal{F}_{\text{AUTH}}$ and IND-CPA secure data encapsulation UC-realise a secure channel.

4.2 Sender-binding Key Encapsulation and IND-SB-CPA Security

In this section I explain a new sender-binding type of key encapsulation mechanism and develop the corresponding security notion of IND-SB-CPA security. As the first step, let us look at what it means for a key encapsulation mechanism to be called sender-binding:

Definition (SB-KEM):
A *sender-binding key encapsulation mechanism (SB-KEM)* is given by a set of three PPT algorithms $(\texttt{gen}, \texttt{enc}, \texttt{dec})$ with

$$\begin{aligned} \texttt{gen}: &\quad 1^\lambda \mapsto (sk, pk) \\ \texttt{enc}: &\quad (pk, S) \mapsto (K, C) \\ \texttt{dec}: &\quad (sk, S, C) \mapsto K, \end{aligned}$$

such that the correctness property holds, i.e.

$$K = \texttt{dec}(sk, S, C)$$

whenever $(sk, pk) \leftarrow \texttt{gen}(1^\lambda)$ and $(K, C) \leftarrow \texttt{enc}(pk, S)$. ○

Note that so far, this is only the traditional key encapsulation mechanism interface enhanced by a party ID as input for encapsulation and decryption. Although the denomination suggests this, the "sender" and "binding" part only become meaningful with the respective security notion. Any classic key encapsulation mechanism instantly satisfies this definition when its input is adjusted to incorporate a party ID, regardless of whether this ID specifies some sender, receiver or just a random party, regardless of whether there is any binding property or the ID can be easily exchanged, even regardless of whether this ID is used at all in the protocol. As with sender-binding encryption, the intended use, however, is that the sending or encapsulating party inserts its *own* ID upon encapsulation. This ID is then non-malleably bound to an otherwise malleable ciphertext of the encapsulated key and decryption is only successful if the *same* ID is used. These properties are expressed in the following IND-SB-CPA notion for sender-binding key encapsulation mechanisms. It is analogous to the corresponding sender-binding encryption notion introduced in Section 3.2, only adapted to fit the key encapsulation setting.

Definition (IND-SB-CPA for SB-KEMs):
A sender-binding key encapsulation mechanism $(\texttt{gen}, \texttt{enc}, \texttt{dec})$ with key space $\mathbf{K}$ satisfies *indistinguishability under sender-binding chosen plaintext attack (IND-SB-CPA)*, if and only if for any PPT adversary $\mathcal{A}_{\text{SB-CPA}}$ the probability to win the IND-SB-CPA game shown in Figure 15 is negligible in the security parameter λ. ○

I would like to remark several things about this definition.
Firstly, IND-SB-CPA security for sender-binding key encapsulation looks very different from other key encapsulation security notions at first glance because it has only one oracle phase instead of two. This is not due to less oracle access but because in our case, the two are equivalent: For IND-SB-CPA, the first and second oracle phase permit exactly the same oracle queries (in contrast to IND-CCA2 for instance). Furthermore, in the key encapsulation setting the adversary does not generate any outputs between oracle phases I and II. Hence within the

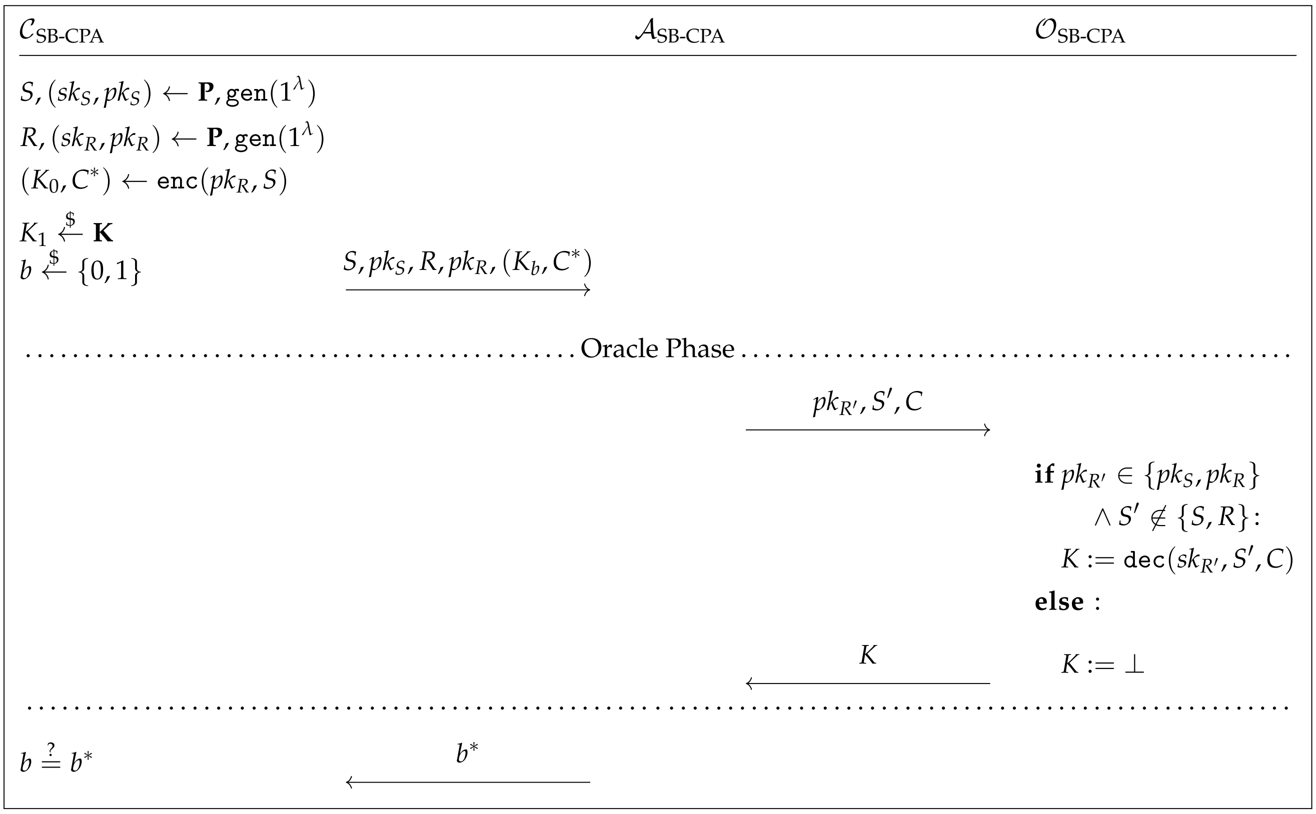

$\mathcal{C}_{\text{SB-CPA}}$
$\mathcal{A}_{\text{SB-CPA}}$
$\mathcal{O}_{\text{SB-CPA}}$
$S, (sk_S, pk_S) \leftarrow \mathbf{P}, \texttt{gen}(1^\lambda)$
$R, (sk_R, pk_R) \leftarrow \mathbf{P}, \texttt{gen}(1^\lambda)$
$(K_0, C^*) \leftarrow \texttt{enc}(pk_R, S)$
$K_1 \xleftarrow{\$} \mathbf{K}$
$b \xleftarrow{\$} \{0,1\}$
$S, pk_S, R, pk_R, (K_b, C^*)$
Oracle Phase
$pk_{R'}, S', C$
$\textbf{if } pk_{R'} \in \{pk_S, pk_R\}$
$\wedge S' \notin \{S, R\}:$
$K := \texttt{dec}(sk_{R'}, S', C)$
$\textbf{else}:$
K
$K := \bot$
$b \overset{?}{=} b^*$
b^*

Figure 15: The IND-SB-CPA Game for Sender-binding Key Encapsulation

IND-SB-CPA game the adversary can "save up" all oracle queries it would make in the first oracle phase and ask them in the second oracle phase instead. I therefore decided to simplify the definition by only including the second oracle phase.

Secondly, note that although the IND-SB-CPA security notion for sender-binding key encapsulation contains a key pair (sk_S, pk_S) for party S, no such keys need to exist in any protocol. Especially in the session communication setting—but also if communication is one-directional in the single-message setting—only one party needs to have a key pair for the sender-binding key encapsulation mechanism to set up a symmetrically encrypted session. The reason behind the existence of these keys in the security notion is that it makes the notion strictly weaker than if (sk_S, pk_S) were not picked by the challenger. Intuitively, an IND-SB-CPA secure sender-binding key encapsulation mechanism does not need to guarantee anything if S's keys may be adversarially chosen rather than honestly (and secretly) generated. This can clearly be seen later on when considering the generic dual receiver construction of sender-binding key encapsulation in Section 5.2.1: For this construction each encapsulated key is decryptable by both the receiver *and* sender. Hence the adversary choosing or knowing sk_S would completely break the encapsulation.

4.3 Realising Secure Message Transfer

In this section I prove that the sender-binding key encapsulation version of IND-SB-CPA security is—in conjunction with IND-OT secure data encapsulation and authenticated channels—

strong enough to facilitate the realisation of secure message transfer. Since we already saw the same for IND-SB-CPA secure sender-binding encryption with authenticated channels in Section 3.3, we can now build on this insight and only fill in the gap: We show that an IND-SB-CPA secure sender-binding key encapsulation mechanism combined via the KEM-DEM framework with an IND-OT secure data encapsulation mechanism yields an IND-SB-CPA secure sender-binding encryption scheme.

Hence let $(\texttt{gen}, \texttt{enc}, \texttt{dec})$ be a sender-binding key encapsulation mechanism and $(\texttt{DEM.enc}, \texttt{DEM.dec})$ be a compatible data encapsulation mechanism. We construct a sender-binding encryption scheme via the KEM-DEM principle as $\texttt{Gen} \equiv \texttt{gen}$ and:

$\texttt{Enc}(pk_R, S, m)$:
- $(K, C) \leftarrow \texttt{enc}(pk_R, S)$.
- $c \leftarrow \texttt{DEM.enc}(K, m)$.

$\hookrightarrow$ Return (C, c).

$\texttt{Dec}\big(sk_R, S, (C, c)\big)$:
- $K := \texttt{dec}(sk_R, S, C)$.
- $m := \texttt{DEM.dec}(K, c)$.

$\hookrightarrow$ Return m.

Theorem 4:
If the sender-binding key encapsulation mechanism $(\texttt{gen}, \texttt{enc}, \texttt{dec})$ *is IND-SB-CPA secure and the data encapsulation mechanism* $(\texttt{DEM.enc}, \texttt{DEM.dec})$ *is IND-OT secure, the resulting sender-binding encryption scheme* $(\texttt{Gen}, \texttt{Enc}, \texttt{Dec})$ *is IND-SB-CPA secure as well.*

Proof:
Assume there is an adversary $\mathcal{A}_{\text{SBE}}$ for the sender-binding encryption version of the IND-SB-CPA game with success probability

$$\mathbb{P}[\mathcal{A}_{\text{SBE}} \text{ successful}] = \tfrac{1}{2} + \rho,$$

where ρ is non-negligible in the security parameter λ. We use this to construct an adversary $\mathcal{A}_{\text{SB-KEM}}$ for the sender-binding key encapsulation version of the IND-SB-CPA game as follows: $\mathcal{A}_{\text{SB-KEM}}$ is started with input

$$\Big(S, pk_S, R, pk_R, (K_b, C^*)\Big)$$

by the key encapsulation challenger $\mathcal{C}_{\text{SB-KEM}}$ and hands (S, pk_S, R, pk_R) on to $\mathcal{A}_{\text{SBE}}$. For any valid oracle query $(pk_{R'}, S', (C, c))$ from $\mathcal{A}_{\text{SBE}}$ the data encapsulation key is decrypted via the sender-binding key encapsulation oracle $\mathcal{O}_{\text{SB-KEM}}$ and subsequently used for decryption of c. When $\mathcal{A}_{\text{SB-KEM}}$ receives challenge messages m_0, m_1 the adversary $\mathcal{A}_{\text{SB-KEM}}$ draws a challenge bit b' uniformly at random from $\{0, 1\}$ and determines the challenge as

$$c^* \leftarrow \texttt{DEM.enc}(K_b, m_{b'}).$$

The following second oracle phase is conducted exactly as the first one was. Finally, in case $\mathcal{A}_{\text{SBE}}$ correctly answers with b', $\mathcal{A}_{\text{SB-KEM}}$ chooses to answer the challenger with $b^* = 0$, else it answers with $b^* = 1$. The detailed reduction is shown in Figure 16.

Let us briefly analyse the success probability of $\mathcal{A}_{\text{SB-KEM}}$. If $b = 0$, $\mathcal{A}_{\text{SB-KEM}}$ has the same success probability that $\mathcal{A}_{\text{SBE}}$ has. If $b = 1$ we claim that the success probability can only negligibly differ from being even.

We again show this by contradiction, this time with a reduction to the IND-OT secure data encapsulation mechanism: Assume that when the game is conducted with $b = 1$, $\mathcal{A}_{\text{SBE}}$ has a

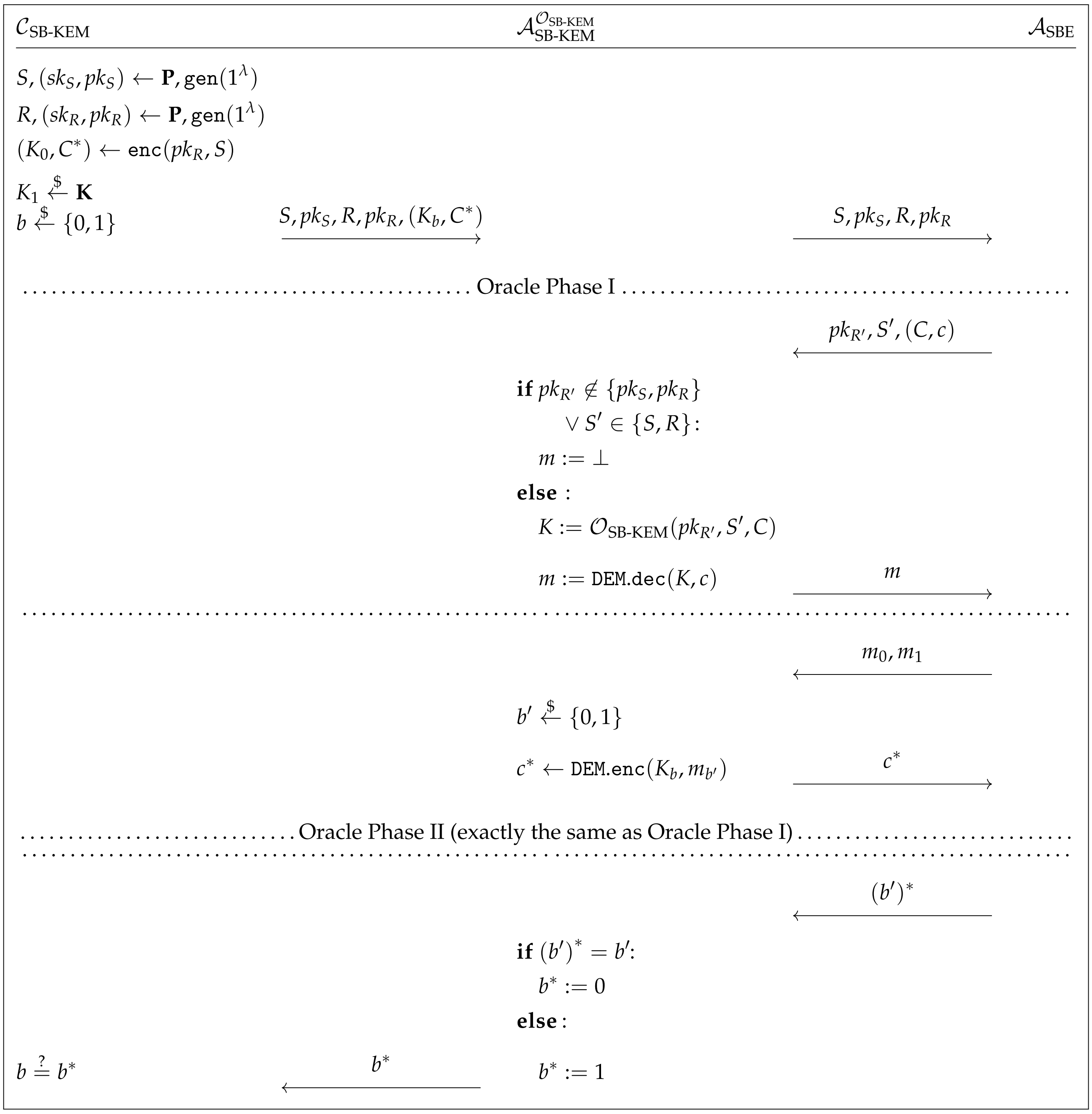

$\mathcal{C}_{\text{SB-KEM}}$
$\mathcal{A}_{\text{SB-KEM}}^{\mathcal{O}_{\text{SB-KEM}}}$
$\mathcal{A}_{\text{SBE}}$
$S, (sk_S, pk_S) \leftarrow \mathbf{P}, \mathtt{gen}(1^\lambda)$
$R, (sk_R, pk_R) \leftarrow \mathbf{P}, \mathtt{gen}(1^\lambda)$
$(K_0, C^*) \leftarrow \mathtt{enc}(pk_R, S)$
$K_1 \xleftarrow{\$} \mathbf{K}$
$b \xleftarrow{\$} \{0,1\}$
$S, pk_S, R, pk_R, (K_b, C^*)$
S, pk_S, R, pk_R
Oracle Phase I
$pk_{R'}, S', (C, c)$
$\textbf{if } pk_{R'} \notin \{pk_S, pk_R\}$
$\vee S' \in \{S, R\}$:
$m := \perp$
$\textbf{else}$:
$K := \mathcal{O}_{\text{SB-KEM}}(pk_{R'}, S', C)$
$m := \mathtt{DEM.dec}(K, c)$
m
m_0, m_1
$b' \xleftarrow{\$} \{0,1\}$
$c^* \leftarrow \mathtt{DEM.enc}(K_b, m_{b'})$
c^*
Oracle Phase II (exactly the same as Oracle Phase I)
$(b')^*$
$\textbf{if } (b')^* = b'$:
$b^* := 0$
$\textbf{else}$:
$b^* := 1$
b^*
$b \overset{?}{=} b^*$

Figure 16: Reduction from Sender-binding Encryption to Sender-binding Key Encapsulation

success probability non-negligibly different from guessing—w.l.o.g. better (rather than worse) than one half. We use $\mathcal{A}_{\text{SBE}}$ to construct an adversary $\mathcal{A}_{\text{DEM}}$ against the data encapsulation IND-OT game: $\mathcal{A}_{\text{DEM}}$ does not get any input from the challenger. It firstly draws S and R, generates (sk_S, pk_S) and (sk_R, pk_R), and hands (S, pk_S, R, pk_R) to $\mathcal{A}_{\text{SBE}}$. Every valid oracle query $(pk_{R'}, S', (C, c))$ is answered by using the corresponding secret key with

$$m := \mathtt{Dec}\Big(sk_{R'}, S', (C, c)\Big).$$

When $\mathcal{A}_{\text{SBE}}$ chooses challenge messages m_0, m_1 they are handed on to the data encapsulation challenger $\mathcal{C}_{\text{DEM}}$ who responds with a corresponding challenge c^*. This challenge is paired with an output C^* from $\mathtt{enc}(pk_R, S)$ and handed to $\mathcal{A}_{\text{SBE}}$. The second oracle phase, again, is handled exactly as the first one was. Finally the answer b^* from $\mathcal{A}_{\text{SBE}}$ is passed on to the challenger. The detailed reduction is shown in Figure 17.

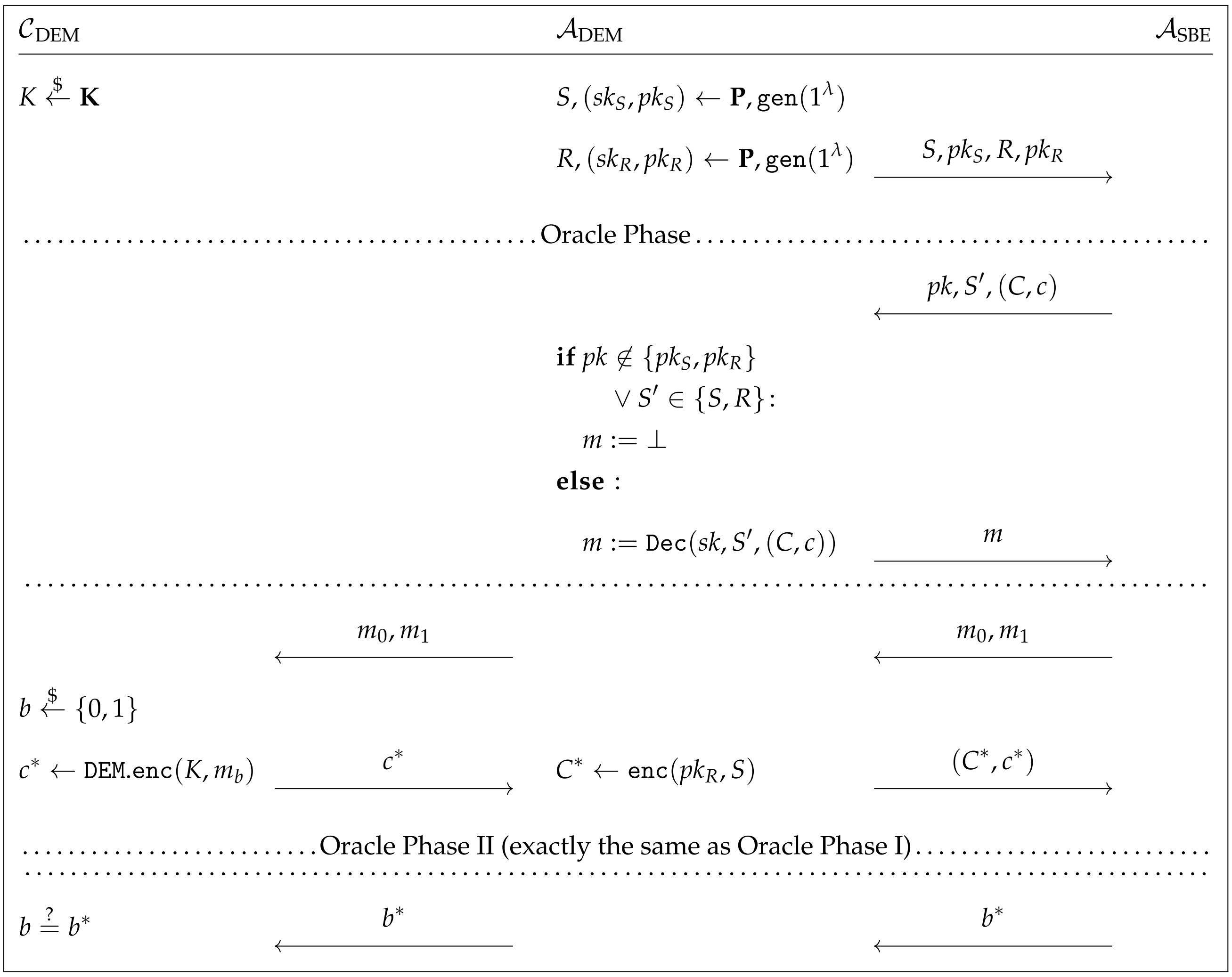

$\mathcal{C}_{\text{DEM}}$
$\mathcal{A}_{\text{DEM}}$
$\mathcal{A}_{\text{SBE}}$
$K \xleftarrow{\$} \mathbf{K}$
$S, (sk_S, pk_S) \leftarrow \mathbf{P}, \mathtt{gen}(1^\lambda)$
$R, (sk_R, pk_R) \leftarrow \mathbf{P}, \mathtt{gen}(1^\lambda)$
S, pk_S, R, pk_R
Oracle Phase
$pk, S', (C, c)$
if $pk \notin \{pk_S, pk_R\}$
$\vee S' \in \{S, R\}$:
$m := \perp$
else :
$m := \mathtt{Dec}(sk, S', (C, c))$
m
m_0, m_1
m_0, m_1
$b \xleftarrow{\$} \{0, 1\}$
$c^* \leftarrow \mathtt{DEM.enc}(K, m_b)$
c^*
$C^* \leftarrow \mathtt{enc}(pk_R, S)$
(C^*, c^*)
Oracle Phase II (exactly the same as Oracle Phase I)
$b \overset{?}{=} b^*$
b^*
b^*

Figure 17: Reduction from Sender-binding Encryption to Data Encapsulation

This reduction to the underlying IND-OT secure data encapsulation mechanism shows that for $b = 1$ in the first reduction, the adversary $\mathcal{A}_{\text{SBE}}$ cannot perform non-negligibly better or worse than guessing. Hence, paired with the case $b = 0$, the adversary $\mathcal{A}_{\text{SB-KEM}}$ has the non-negligible success probability

$$\mathbb{P}\left[\mathcal{A}_{\text{SB-KEM}} \text{ successful}\right] \geqslant \tfrac{1}{2} + \tfrac{1}{2}\rho - \varepsilon$$

where ε is negligible in the security parameter λ. □

Corollary 1:
An IND-SB-CPA secure sender-binding key encapsulation mechanism and compatible IND-OT secure data encapsulation mechanism suffice to UC-realise the secure message transfer functionality $\mathcal{F}_{M\text{-}SMT}$ in the $\mathcal{F}_{AUTH}$-hybrid model under static corruption.

The proof of this corollary follows directly from Theorems 2 and 4.

4.4 Realising Secure Channels

At this point we make the switch from single-message to session communication. This means a symmetric key is exchanged via the key encapsulation mechanism and subsequently used by both parties to send multiple messages encrypted under this same key via the corresponding data encapsulation mechanism. The benefits are less overhead for key exchanges and that only one communication partner needs credentials for the key encapsulation mechanism to gain bi-directional communication. The employed data encapsulation mechanism, on the other hand, needs to be stronger than for single-message KEM-DEM.

In this section I prove that an IND-SB-CPA secure key encapsulation mechanism in conjunction with an IND-CPA secure data encapsulation mechanism suffices to UC-realise the secure channel functionality $\mathcal{F}_{\text{M-SC}}$ in the $\mathcal{F}_{\text{AUTH}}$-hybrid model under static corruption. I do so by first defining a protocol $\pi_{\text{M-SC}}$ and corresponding simulator $\mathcal{S}_{\text{M-SC}}$ in Sections 4.4.1 and 4.4.2, respectively, before stating the actual security theorem and conducting the proof in Section 4.4.3.

4.4.1 Protocol $\pi_{\text{M-SC}}$

Let $(\texttt{gen}, \texttt{enc}, \texttt{dec})$ be a sender-binding key encapsulation mechanism and $(\texttt{DEM.enc}, \texttt{DEM.dec})$ a compatible data encapsulation mechanism. The idea behind $\pi_{\text{M-SC}}$ is the following: To establish a session between parties P and P', a new symmetric key is generated and encapsulated by P via

$$(K, C) \leftarrow \texttt{enc}(pk_{P'}, P).$$

The resulting ciphertext C is sent to P' via authenticated channel. When decryption

$$K \leftarrow \texttt{dec}(sk_{P'}, P, C)$$

is successful, both parties can encrypt messages to the other party via $\texttt{DEM.enc}$ using K and send them via authenticated channel as well. All details can be found in the following formal definition.

$\pi_{\textbf{M-SC}}$

Realises:
Multiple secure two-party communication sessions with message space $\mathbf{M}$ and polynomially many parties $P \in \mathbf{P}$.

Parameters:

- Key encapsulation mechanism $(\texttt{gen}, \texttt{enc}, \texttt{dec})$.
- Compatible data encapsulation mechanism $(\texttt{DEM.enc}, \texttt{DEM.dec})$ with message space $\mathbf{M}$.
- Ideal authenticated channel functionality $\mathcal{F}_{\text{AUTH}}$.

State of party P:

- Personal key encapsulation key pair $(sk_P, pk_P) \in \mathbf{PK} \times \mathbf{SK}$.
- Partial function $\mathtt{f}_{\mathbf{PK}} \colon \mathbf{P} \to \mathbf{PK}, P' \mapsto pk_{P'}$ of known public key encapsulation keys.
- Partial function $\mathtt{f}_{\mathbf{K}} \colon \mathbf{P} \to \mathbf{K}, P' \mapsto K$ of known symmetric data encapsulation session keys.
- Almost Boolean function $\mathtt{f}_{\mathsf{act}} \colon \mathbf{P} \to \{\mathsf{true}, \mathsf{false}, \mathsf{init}\}$ indicating active sessions. Initialised to $\mathtt{f}_{\mathsf{act}} \equiv \mathsf{false}$.

Behaviour of Party P:

\\\\ Initialisation

- Upon input $(\mathtt{sent}, P', P, pk)$ from $\mathcal{F}_{\text{AUTH}}$, append $P' \mapsto pk_{P'}$ to $\mathtt{f}_{\mathbf{PK}}$ if this entry does not yet exist.
- Upon input $(\mathtt{init}, P')$ from the environment:
  (1) If no personal key encapsulation key pair (sk_P, pk_P) exists yet, set $(sk_P, pk_P) \leftarrow \mathtt{gen}(1^\lambda)$.
  (2) Check that $\mathtt{f}_{\mathsf{act}}(P') = \mathsf{false}$ and set $\mathtt{f}_{\mathsf{act}}(P') := \mathsf{init}$.
  (3) Hand input $(\mathtt{send}, P', pk_P)$ to a new instance of $\mathcal{F}_{\text{AUTH}}$.
- Upon input $(\mathtt{establish}, P')$ from the environment:
  (1) Look up $pk_{P'} := \mathtt{f}_{\mathbf{PK}}(P')$.
  (2) $(K, C) \leftarrow \mathtt{enc}(pk_{P'}, P)$.
  (3) Check that $\mathtt{f}_{\mathsf{act}}(P') = \mathsf{false}$, set $\mathtt{f}_{\mathsf{act}}(P') = \mathsf{true}$ and append $P' \mapsto K$ to $\mathtt{f}_{\mathbf{K}}$.
  (4) Hand input $(\mathtt{send}, P', C)$ to a new instance of $\mathcal{F}_{\text{AUTH}}$.
- Upon input $(\mathtt{sent}, P', P, C)$ from $\mathcal{F}_{\text{AUTH}}$:
  (1) Look up personal key encapsulation key pair (sk_P, pk_P).
  (2) $K := \mathtt{dec}(sk, P', C)$.
  (3) Check that $\mathtt{f}_{\mathsf{act}}(P') = \mathsf{init}$, set $\mathtt{f}_{\mathsf{act}}(P') = \mathsf{true}$ and append $P' \mapsto K$ to $\mathtt{f}_{\mathbf{K}}$.

\\\\ Data Exchange

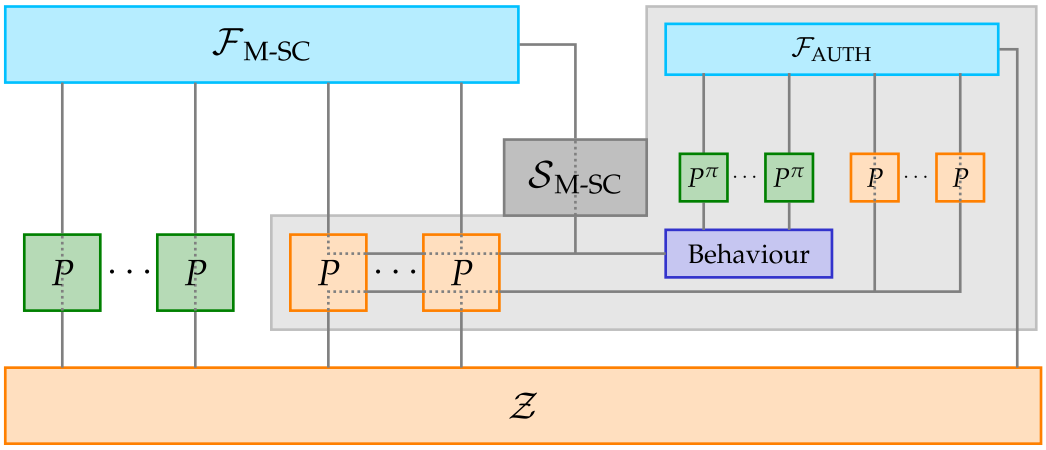
- Upon input $(\mathtt{send}, P', m)$ with $m \in \mathbf{M}$ from environment $\mathcal{Z}$:
  (1) Check $\mathtt{f}_{\mathsf{act}}(P') = \mathsf{true}$, look up $K := \mathtt{f}_{\mathbf{K}}(P')$ and set $c \leftarrow \mathtt{DEM.enc}(K, m)$.
  (2) Hand input $(\mathtt{send}, P', c)$ to a new instance of $\mathcal{F}_{\text{AUTH}}$.
- Upon input $(\mathtt{sent}, P', P, c)$ from $\mathcal{F}_{\text{AUTH}}$:
  (1) Check $\mathtt{f}_{\mathsf{act}}(P') = \mathsf{true}$, look up $K := \mathtt{f}_{\mathbf{K}}(P')$ and set $m \leftarrow \mathtt{DEM.dec}(K, c)$.
  (2) Output $(\mathtt{sent}, P', m)$ to the environment.

\\\\ Session Expiration

- Upon input $(\mathtt{expire}, P')$ from the environment:
  (1) Check $\mathtt{f}_{\mathsf{act}} = \mathsf{true}$ and hand input $(\mathtt{send}, P', \mathtt{expire})$ to a new instance of $\mathcal{F}_{\text{AUTH}}$.
  (2) Erase $\mathtt{f}_{\mathbf{K}}(P')$ and set $\mathtt{f}_{\mathsf{act}}(P') := \mathsf{false}$.
- Upon input $(\mathtt{sent}, P', P, \mathtt{expire})$ from $\mathcal{F}_{\text{AUTH}}$ erase $\mathtt{f}_{\mathbf{K}}(P')$ and set $\mathtt{f}_{\mathsf{act}}(P') := \mathsf{false}$.

4.4.2 Simulator $\mathcal{S}_{\text{M-SC}}$

To show that protocol $\pi_{\text{M-SC}}$ realises $\mathcal{F}_{\text{M-SC}}$ we need to construct a simulator $\mathcal{S}_{\text{M-SC}}$ which interacts with $\mathcal{F}_{\text{M-SC}}$ in such a way that no environment $\mathcal{Z}$ can distinguish this ideal world from an interaction with the real protocol and dummy adversary $\mathcal{D}$. The idea behind our simulator $\mathcal{S}_{\text{M-SC}}$ is striving for near perfect simulation, just as the simulator we constructed in Section 3.3.2: It plays all honest parties (conducting protocol $\pi_{\text{M-SC}}$) as well as the functionality $\mathcal{F}_{\text{AUTH}}$ in its head, using $\mathcal{F}_{\text{M-SC}}$'s outputs to give them mock inputs from $\mathcal{Z}$ and using their outputs in turn to determine inputs to $\mathcal{F}_{\text{M-SC}}$. An overview can be found in Figure 18. For proof simplicity



$\mathcal{F}_{\text{M-SC}}$
$\mathcal{F}_{\text{AUTH}}$
$\mathcal{S}_{\text{M-SC}}$
P^π … P^π
P … P
Behaviour
P … P
P … P
$\mathcal{Z}$

Figure 18: Overview of Simulator $\mathcal{S}_{\text{M-SC}}$

purposes—that become apparent later on—the simulator swaps symmetric keys for random values if the two involved parties are both honest. The only situations in which $\mathcal{S}_{\text{M-SC}}$ is unable to provide perfect simulation due to lack of knowledge are messages between two honest parties. In this case it sends encryptions of zeros instead. The formal definition of $\mathcal{S}_{\text{M-SC}}$ looks as follows:

$\mathcal{S}_{\textbf{M-SC}}$

Realises:
Multiple secure two-party communication sessions with message space $\mathbf{M}$ and polynomially many parties $P \in \mathbf{P}$.

Parameters:
- Key encapsulation mechanism $(\texttt{gen}, \texttt{enc}, \texttt{dec})$ with key space $\mathbf{K}$.
- Compatible data encapsulation mechanism $(\texttt{DEM.enc}, \texttt{DEM.dec})$ with message space $\mathbf{M}$.

In-the-head Parties:
- Functionality $\mathcal{F}_{\text{AUTH}}$. This functionality communicates in-the-head with all honest in-the-head parties as well as with the environment $\mathcal{Z}$ as adversary.
- Copies of honest parties running a modified version of the protocol $\pi_{\text{M-SC}}$, which we will denote as P^π. These parties communicate in-the-head with the in-the-head functionality $\mathcal{F}_{\text{AUTH}}$. Their interface to the environment is played by the simulator (defined in "Behaviour" below). The modification from $\pi_{\text{M-SC}}$ looks as follows:

- ◦ Upon input $(\texttt{establish}, P')$ from the environment:
  - (3) Check that $\texttt{f}_{\text{act}}(P') = \mathsf{false}$, set $\texttt{f}_{\text{act}}(P') = \mathsf{true}$, ask $\mathcal{S}$ for freshly drawn random key $K_1 \xleftarrow{\$} \mathbf{K}$ for parties $\{P, P'\}$ and append $P' \mapsto K_1$ to $\texttt{f}_{\mathbf{K}}$.
- ◦ Upon input $(\texttt{sent}, P', P, C)$ from $\mathcal{F}_{\text{AUTH}}$:
  - (3) Check that $\texttt{f}_{\text{act}}(P') = \mathsf{init}$, set $\texttt{f}_{\text{act}}(P') = \mathsf{true}$, ask $\mathcal{S}$ for key K_1 corresponding to parties $\{P, P'\}$ and append $P' \mapsto K_1$ to $\texttt{f}_{\mathbf{K}}$.

- Dummy corrupted parties. Whenever the simulator is asked by the environment $\mathcal{Z}$ to call functionality $\mathcal{F}_{\text{AUTH}}$ in the name of a corrupted party, this in-the-head dummy calls the in-the-head functionality correspondingly and reports all outputs back to the environment $\mathcal{Z}$.

State:

- Everything the in-the-head parties and functionalities store in their states.
- Partial function $\{\{P, P'\} \mid P, P' \text{ honest}\} \to \mathbf{K}, \{P, P'\} \mapsto K_1$ of symmetric DEM session keys.

Behaviour:

\\ Initialisation by honest party

- Upon receiving $(\texttt{inited}, A, B)$ from $\mathcal{F}_{\text{M-SC}}$ for honest party A, start in-the-head party A^π with input $(\texttt{init}, B)$ from the environment $\mathcal{Z}$.
- Upon receiving $(\texttt{established}, A, B)$ from $\mathcal{F}_{\text{M-SC}}$ for honest party B, start in-the-head party B^π with input $(\texttt{establish}, A)$ from the environment $\mathcal{Z}$.

\\ Initialisation by corrupted party

- Upon in-the-head party B^π receiving output $(\texttt{sent}, A, B, pk)$ from $\mathcal{F}_{\text{AUTH}}$ for corrupted A, call $\mathcal{F}_{\text{SC}}$ with input $(\texttt{init}, B)$ in the name of A.
- Upon in-the-head party A^π setting $\texttt{f}_{\text{act}}(B)$ from init to true, call $\mathcal{F}_{\text{M-SC}}$ with input $(\texttt{establish}, A)$ in the name of B.

\\ Message from honest to honest party

- Upon receiving $(\texttt{send}, mid, S, R)$ from $\mathcal{F}_{\text{M-SC}}$ for honest parties S and R:
  1. (1) Start in-the-head party S^π with input $(\texttt{send}, R, 0)$ from the environment $\mathcal{Z}$.
  2. (2) If in-the-head party R^π at some point reports output $(\texttt{sent}, S, 0)$, call $\mathcal{F}_{\text{M-SC}}$ with input $(\texttt{send ok}, mid)$.[9]

\\ Message from honest to corrupted party

- Upon receiving $(\texttt{send}, mid, S, R)$ from $\mathcal{F}_{\text{M-SC}}$ for honest party S and corrupted party R:
  1. (1) Call $\mathcal{F}_{\text{M-SC}}$ with input $(\texttt{send ok}, mid)$.
  2. (2) Receive output $(\texttt{sent}, S, m)$ from $\mathcal{F}_{\text{M-SC}}$ to R.
  3. (3) Start in-the-head party S^π with input $(\texttt{send}, R, m)$ from the environment $\mathcal{Z}$.

\\ Message from corrupted to honest party

- Upon in-the-head honest party R^π reporting output $(\texttt{sent}, S, m)$:
  (1) Call $\mathcal{F}_{\text{M-SC}}$ with input $(\texttt{send}, R, m)$ in the name of S.
  (2) Receive output $(\texttt{send}, \mathit{mid}, S, R)$ from $\mathcal{F}_{\text{M-SC}}$.
  (3) Call $\mathcal{F}_{\text{M-SC}}$ with input $(\texttt{send ok}, \mathit{mid})$.

4.4.3 Security Theorem and Proof

Now that we have constructed both protocol $\pi_{\text{M-SC}}$ and simulator $\mathcal{S}_{\text{M-SC}}$ it remains to show that under the right assumptions they make the real and ideal world indistinguishable for any environment $\mathcal{Z}$. Similar to Section 3.3.3 we do so by first explicitly stating the differences between the simulator's efforts and perfect simulation. Then we go on to define several hybrid experiments which help us conduct the proof of our security theorem.

Remark 3:
It is easy to see that the simulator $\mathcal{S}_{\text{M-SC}}$ provides nearly perfect simulation. The two notable exceptions are:

(1) Symmetric keys of sessions between two honest parties: The modification of protocol $\pi_{\text{M-SC}}$ for the in-the-head honest parties P^π changes the session keys for each session between two honest parties. While a session key K is generated and the corresponding ciphertext C sent via $\mathcal{F}_{\text{AUTH}}$—just like in the real protocol—all messages of the session are encrypted with a randomly drawn and unrelated key

$$K_1 \xleftarrow{\$} \mathbf{K}.$$

(2) Message content between two honest parties: Let S, R and m be the honest parties and message in question. In this case the environment will receive a message

$$\Big(\texttt{send}, S, R, \texttt{DEM.enc}(K_1, 0)\Big)$$

from $\mathcal{F}_{\text{AUTH}}$ (played by $\mathcal{S}_{\text{M-SC}}$) in the ideal execution, while in the protocol execution it receives

$$\Big(\texttt{send}, S, R, \texttt{DEM.enc}(K, m)\Big).$$

Hence any environment $\mathcal{Z}$ which distinguishes experiments $\mathsf{EXEC}(\pi_{\text{M-SC}}, \mathcal{D})$ and $\mathsf{EXEC}(\mathcal{F}_{\text{M-SC}}, \mathcal{S}_{\text{M-SC}})$ can only do so by session keys or messages between honest parties. ○

Before we proceed to the security theorem and proof we now define several hybrid experiments which allow us to utilise the polynomial hybrid argument from Lemma 2 (cf. Section 2.2.1). This simplifies our security proof as we only need to deal with pairs of consecutive hybrids instead of the larger difference between the real and ideal experiment.

9 We assume the simulator to internally track the protocol executions to know which *mid* to use.

Definition (Hybrids H^{-},$H^{\vdash}_{i}$,$H^{\dashv}_{i,j}$):
Instead of just one infinite hybrid chain as in Section 3.3.3, we define two different infinite chains of hybrids for this slightly more involved security proof.

- We use a "middle" hybrid H^{-} where all honest parties swap encapsulated session keys K for randomly drawn K_1's, while still using ciphertexts C corresponding to K. I.e. parties conduct the same modified protocol as the simulator's in-the-head honest parties P^{π}. This means session keys of two honest parties are handled exactly as in the ideal experiment. Note that in contrast to the ideal experiment, for every message m between two honest parties S and R in H^{-} an output

  $$\Big(\texttt{send}, S, R, \texttt{DEM.enc}(K_1, m)\Big)$$

  is handed to environment $\mathcal{Z}$ which contains an encryption of m and *not* an encryption of 0.

- Let $i \in \mathbb{N}_0$ be a natural number or zero. We define $H^{\vdash}_{i}$ to be almost identical to the real world execution of $\pi_{\text{M-SC}}$ with the sole difference that for the first i sessions between two honest parties, the encapsulated key K is swapped for a randomly drawn K_1. Hence we have $H^{\vdash}_{0} = \mathsf{EXEC}(\pi_{\text{M-SC}}, \mathcal{D})$ and

  $$\lim_{i\to\infty} H^{\vdash}_{i} = H^{-}.$$

- For our second sequence of hybrids we firstly define an auxiliary sequence $(a_n)_{n\in\mathbb{N}}$ where each element a_n is a tuple (i, j) of natural numbers. Let $a_1 := (1, 1)$ and define a_{n+1} recursively from $a_n =: (i, j)$ by

  $$a_{n+1} := \begin{cases} (i+1, j-1), & j > 1 \\ (1, i+1), & j = 1. \end{cases}$$

  Note that $(a_n)_{n\in\mathbb{N}}$ actually equals all of $\mathbb{N}^2$ where the elements (i, j) are firstly ordered by the sum $(i + j)$ and secondly by i.
  From this auxiliary sequence we recursively define a sequence of hybrids $(H^{\dashv}_{n})_{n\in\mathbb{N}_0}$. Let $H^{\dashv}_{0} := H^{-}$. For any $n \in \mathbb{N}$ we define $H^{\dashv}_{n}$ to be equal to $H^{\dashv}_{n-1}$ with the exception that for the j-th message in the i-th session between two honest parties, an encryption of zeros is sent over the authenticated channel instead of an encryption containing the real message—just as the ideal execution does for every message—where $(i, j) := a_n$.
  Hence in addition to $H^{\dashv}_{0} = H^{-}$ we have

  $$\lim_{n\to\infty} H^{\dashv}_{n} = \mathsf{EXEC}(\mathcal{F}_{\text{M-SC}}, \mathcal{S}_{\text{M-SC}}).$$ ◦

The above hybrid definitions give us the following double-chain of hybrids connecting the real world execution of $\pi_{\text{M-SC}}$ and the ideal experiment with $\mathcal{F}_{\text{M-SC}}$:

$$\mathsf{EXEC}\Big(\pi^{\mathcal{F}_{\text{AUTH}}}_{\text{M-SC}}, \mathcal{D}\Big) = H^{\vdash}_{0}, H^{\vdash}_{1}, \ldots \to H^{-} = H^{\dashv}_{0}, H^{\dashv}_{1}, \ldots \to \mathsf{EXEC}(\mathcal{F}_{\text{M-SC}}, \mathcal{S}_{\text{M-SC}}).$$

Note that for any $i \in \mathbb{N}$ the hybrids $H^{\vdash}_{i-1}$ and $H^{\vdash}_{i}$ only differ in the i-th session between two honest parties, where the encapsulated key K is swapped for a randomly drawn key K_1. For any $n \in \mathbb{N}$ the difference between hybrids $H^{\dashv}_{n-1}$ and $H^{\dashv}_{n}$ with $a_n = (i, j)$ is limited to the j-th message m_j of the i-th session between two honest parties: An encryption of m_j is sent over the authenticated channel for $H^{\dashv}_{n-1}$ while an encryption of zeros is used in $H^{\dashv}_{n}$.

Now we are finally ready to formally state and prove that $\pi_{\text{M-SC}}$ UC-realises secure channels:

Theorem 5:
Let the sender-binding key encapsulation mechanism $(\texttt{gen}, \texttt{enc}, \texttt{dec})$ *and data encapsulation mechanism* $(\texttt{DEM.enc}, \texttt{DEM.dec})$ *employed in the protocol* $\pi_{M\text{-}SC}$ *satisfy IND-SB-CPA and IND-CPA security, respectively. Then under static corruption,* $\pi_{M\text{-}SC}$ *is a UC realisation of* $\mathcal{F}_{M\text{-}SC}$ *in the* $\mathcal{F}_{AUTH}$*-hybrid model, i.e.*

$$\pi^{\mathcal{F}_{AUTH}}_{M\text{-}SC} \geqslant_{UC} \mathcal{F}_{M\text{-}SC}.$$

Proof:
We conduct the proof in two steps, we separately show that...

(1) $\dots \mathsf{EXEC}(\pi^{\mathcal{F}_{\text{AUTH}}}_{\text{M-SC}}, \mathcal{D})$ is indistinguishable from H^{-}.

(2) $\dots H^{-}$ is indistinguishable from $\mathsf{EXEC}(\mathcal{F}_{\text{M-SC}}, \mathcal{S}_{\text{M-SC}})$.

We reduce the first step to the IND-SB-CPA security of the underlying sender-binding key encapsulation mechanism and the second step to the IND-CPA security of the data encapsulation mechanism. For both parts we employ Lemma 2 to go from consecutive hybrids to a polynomial stretch of the corresponding infinite hybrid chains.

(1) We firstly show that for any $i \in \mathbb{N}$ the two consecutive hybrids $H^{\vdash}_{i-1}$ and $H^{\vdash}_{i}$ are computationally indistinguishable and that furthermore the advantage

$$\mathsf{Adv}^{\text{indist}}_{H^{\vdash}_{i-1}, H^{\vdash}_{i}, \mathcal{Z}_i}(\lambda)$$

is bounded independently of i by a function which is negligible in the security parameter λ. Let i be a natural number and $\mathcal{Z}_i$ be an environment trying to distinguish executions of hybrids $H^{\vdash}_{i-1}$ and $H^{\vdash}_{i}$. We now use environment $\mathcal{Z}_i$ to construct an adversary $\mathcal{A}_{\text{SB-CPA}}$ whose probability to win the IND-SB-CPA game with respect to the underlying sender-binding key encapsulation mechanism $\Sigma = (\texttt{gen}, \texttt{enc}, \texttt{dec})$ is directly related to the success probability of $\mathcal{Z}_i$.
The adversary $\mathcal{A}_{\text{SB-CPA}}$ is started by $\mathcal{C}_{\text{SB-CPA}}$ with input

$$\Big(S, pk_S, R, pk_R, (K_b, C^*)\Big)$$

and in turn starts $\mathcal{Z}_i$ in its head, playing all other parties just like they would conduct hybrid $H^{\vdash}_{i-1}$ or $H^{\vdash}_{i}$. If $\mathcal{Z}_i$ corrupts either S or R, the adversary aborts the simulation and guesses $b \in \{0, 1\}$ uniformly at random. When $\mathcal{Z}_i$ asks honest party S or R to initialise for the first time, $\mathcal{A}_{\text{SB-CPA}}$ inserts pk_S/pk_R as S/R's public key respectively for the key encapsulation mechanism. Every time in-the-head party S or R is send a ciphertext C encrypted under pk_S/pk_R by some corrupted party P, $\mathcal{A}_{\text{SB-CPA}}$ decrypts it via the IND-SB-CPA oracle of the sender-binding key encapsulation mechanism. This is possible since S and R are honest and hence $P \notin \{S, R\}$. Since honest parties only get interface inputs from $\mathcal{Z}_i$, $\mathcal{A}_{\text{SB-CPA}}$ already knows the content of all ciphertexts C sent from honest parties to S and R

and does not need the oracle to decrypt them.
If the i-th request of $(\texttt{establish}, P)$ by $\mathcal{Z}$ to establish a session between two honest parties is not made to S with $P = R$, $\mathcal{A}_{\text{SB-CPA}}$ aborts the simulation and guesses $b \xleftarrow{\$} \{0,1\}$ uniformly at random. Since S and R were randomly drawn by the challenger from the set $\mathbf{P}$, whose size $|\mathbf{P}| = |\mathbf{P}(\lambda)|$ is polynomial in the security parameter, S and R are information-theoretically indistinguishable from other honest parties by $\mathcal{Z}_i$. Furthermore by Remark 3 the environment $\mathcal{Z}_i$ needs a message between honest parties to distinguish anything, $\mathcal{A}_{\text{SB-CPA}}$ has a polynomial chance of at least

$$\mathbb{P}[\text{continue simulation}] \geqslant \frac{1}{|\mathbf{P}|^2}$$

to not have aborted the simulation up to this point.
If $\mathcal{A}_{\text{SB-CPA}}$ does not abort, it inserts the challenge cipher C^* into the message

$$\left(\texttt{send}, P', C^*\right)$$

from S to R via $\mathcal{F}_{\text{AUTH}}$ and has S and R use challenge key K_b as the data encapsulation key throughout this session. For all following sessions $\mathcal{A}_{\text{SB-CPA}}$ uses the encapsulated session keys K just as $H^{\vdash}_{i-1}$ and $H^{\vdash}_{i}$ both specify. When $\mathcal{Z}_i$ halts, $\mathcal{A}_{\text{SB-CPA}}$ outputs $b = 0$ if $\mathcal{Z}_i$ outputs $H^{\vdash}_{i-1}$, and $b = 1$ if $\mathcal{Z}_i$ outputs $H^{\vdash}_{i}$.
$\mathcal{A}_{\text{SB-CPA}}$ wins the IND-SB-CPA game for sender-binding key encapsulation in an even half of the cases where it aborted the simulation. Additionally, whenever it did not abort, $\mathcal{A}_{\text{SB-CPA}}$ has the same success probability as $\mathcal{Z}_i$. Combining both cases, $\mathcal{A}_{\text{SB-CPA}}$ has an advantage of at least

$$\mathsf{Adv}^{\text{SB-CPA}}_{\mathcal{A}_{\text{SB-CPA}}}(\lambda) \geqslant \frac{1}{|\mathbf{P}(\lambda)|^2} \cdot \mathsf{Adv}^{\text{indist}}_{H^{\vdash}_{i-1}, H^{\vdash}_{i}, \mathcal{Z}_i}(\lambda).$$

Now remember that one premise of Theorem 5 is for the sender-binding key encapsulation mechanism Σ to satisfy IND-SB-CPA security. Hence there is a negligible function ε which bounds the advantage of any IND-SB-CPA adversary against Σ, i.e.

$$\mathsf{Adv}^{\text{indist}}_{H^{\vdash}_{i-1}, H^{\vdash}_{i}, \mathcal{Z}_i}(\lambda) \leqslant |\mathbf{P}(\lambda)|^2 \cdot \mathsf{Adv}^{\text{SB-CPA}}_{\mathcal{A}_{\text{SB-CPA}}}(\lambda) \leqslant |\mathbf{P}(\lambda)|^2 \cdot \varepsilon(\lambda).$$

Not only does this prove the success probability of $\mathcal{Z}_i$ to be negligible and $H^{\vdash}_{i-1}$ and $H^{\vdash}_{i}$ to be computationally indistinguishable. The upper bound $|\mathbf{P}(\lambda)|^2 \cdot \varepsilon(\lambda)$ is also independent of i which by Lemma 2 means that hybrids $H^{\vdash}_{0}$ and $H^{\vdash}_{\mathsf{p}(\lambda)}$ are computationally indistinguishable as well for any polynomial p.
Now let $\mathcal{Z}$ be an arbitrary PPT environment trying to distinguish executions

$$\mathsf{EXEC}\left(\pi^{\mathcal{F}_{\text{AUTH}}}_{\text{M-SC}}, \mathcal{D}\right) = H^{\vdash}_{0} \quad \text{and} \quad H^{-}.$$

There exists a polynomial p in the security parameter λ which bounds the runtime of the environment $\mathcal{Z}$, i.e. executions of hybrids $H^{\vdash}_{\mathsf{p}(\lambda)}$ and H^{-} are trivially perfectly indistinguishable for $\mathcal{Z}$ as they do not differ before the $\mathsf{p}(\lambda)$-th session. Hence we know by above considerations that

$$\mathsf{EXEC}\left(\pi^{\mathcal{F}_{\text{AUTH}}}_{\text{M-SC}}, \mathcal{D}\right) \sim_{\mathcal{Z}} H^{\vdash}_{\mathsf{p}(\lambda)} \sim_{\mathcal{Z}} H^{-},$$

and by finite transitivity of indistinguishability the real world execution is indistinguishable from our middle hybrid H^-.

(2) This second part of the proof follows the same structure as the first: Through a reduction to the IND-CPA security of the underlying data encapsulation mechanism we show that two consecutive hybrids $H^{\dashv}_{n-1}$ and $H^{\dashv}_{n}$ are computationally indistinguishable with a universal negligible bound on the distinguishing advantage for any $n \in \mathbb{N}$. Then we employ Lemma 2 to infer the computational indistinguishability of H^- and $\mathsf{EXEC}(\mathcal{F}_{\text{M-SC}}, \mathcal{S}_{\text{M-SC}})$.
Let n be a natural number and $\mathcal{Z}_n$ be an environment trying to distinguish executions of hybrids $H^{\dashv}_{n-1}$ and $H^{\dashv}_{n}$. We now use environment $\mathcal{Z}_n$ to construct an adversary $\mathcal{A}_{\text{CPA}}$ whose probability to win the IND-CPA game with respect to the underlying data encapsulation mechanism $\Sigma = (\texttt{DEM.enc}, \texttt{DEM.dec})$ is directly related to the success probability of $\mathcal{Z}_n$.
After the challenger $\mathcal{C}_{\text{CPA}}$ has randomly drawn the challenge key, the adversary $\mathcal{A}_{\text{CPA}}$ is started without input and in turn starts $\mathcal{Z}_n$ in its head, playing all other parties just like they would conduct hybrid $H^{\dashv}_{n-1}$ or $H^{\dashv}_{n}$. Let $(i, j) := a_n$. When $\mathcal{Z}_n$ asks for the i-th session between two honest parties—call them S and R—to be established, $\mathcal{A}_{\text{CPA}}$ does not draw a fresh random session key K_1 but rather inserts the (unknown) challenge key instead. This is no problem as communacation is handled via $\mathcal{F}_{\text{AUTH}}$ and $\mathcal{A}_{\text{CPA}}$ knows the content of any message that $\mathcal{Z}_n$ asks S or R to send and can use the encryption oracle to simulate the corresponding channel content.
For the j-th message m_j of this session—which by Remark 3 has to be send by an honest party and hence S or R—$\mathcal{A}_{\text{CPA}}$ hands m_j and 0 to the challenger and in return obtains ciphertext c^* which it uses as the channel content reported to $\mathcal{Z}_n$. Now $\mathcal{A}_{\text{CPA}}$ continues to use encryptions of zeros for all further messages of this session, just as both $H^{\dashv}_{n-1}$ and $H^{\dashv}_{n}$ require. Whenever the challenge ciphertext c^* is sent to S or R within this session again, $\mathcal{A}_{\text{CPA}}$ acts as if the decryption oracle had yielded message m_j.
When $\mathcal{Z}_n$ halts, $\mathcal{A}_{\text{CPA}}$ outputs $b = 0$ if $\mathcal{Z}_n$ outputs $H^{\dashv}_{n-1}$, and $b = 1$ if $\mathcal{Z}_n$ outputs $H^{\dashv}_{n}$. With this construction, $\mathcal{A}_{\text{CPA}}$ wins the data encapsulation IND-CPA game whenever $\mathcal{Z}_n$ successfully distinguishes $H^{\dashv}_{n-1}$ and $H^{\dashv}_{n}$, i.e.

$$\mathsf{Adv}^{\text{CPA}}_{\mathcal{A}_{\text{CPA}}}(\lambda) = \mathsf{Adv}^{\text{indist}}_{H^{\dashv}_{n-1}, H^{\dashv}_{n}, \mathcal{Z}_n}(\lambda).$$

Now remember that the second premise of Theorem 5 is for the data encapsulation mechanism Σ to satisfy IND-CPA security. Hence there is a negligible function ε which bounds the advantage of any IND-CPA adversary against Σ. We get

$$\mathsf{Adv}^{\text{indist}}_{H^{\dashv}_{n-1}, H^{\dashv}_{n}, \mathcal{Z}_n}(\lambda) = \mathsf{Adv}^{\text{CPA}}_{\mathcal{A}_{\text{CPA}}}(\lambda) \leqslant \varepsilon(\lambda).$$

Not only does this prove the success probability of $\mathcal{Z}_n$ to be negligible and $H^{\dashv}_{n-1}$ and $H^{\dashv}_{n}$ to be computationally indistinguishable. The upper bound $\varepsilon(\lambda)$ is also independent of n which by Lemma 2 means that hybrids $H^{\dashv}_{0}$ and $H^{\dashv}_{\mathsf{p}(\lambda)}$ are computationally indistinguishable as well for any polynomial p.
Now let $\mathcal{Z}$ be an arbitrary PPT environment trying to distinguish executions

$$H^- = H^{\dashv}_{0} \quad \text{and} \quad \mathsf{EXEC}(\mathcal{F}_{\text{M-SC}}, \mathcal{S}_{\text{M-SC}}).$$

There exists a polynomial p in the security parameter λ which bounds the runtime of the environment $\mathcal{Z}$. In particular, $\mathcal{Z}$ can send at most $(\mathsf{p}(\lambda) - i)$ messages in the i-th session between honest parties, as it needs to initialise the i different sessions as well as send messages. Due to the Gaussian formula for triangular numbers, we know that any $a_n = (i, j)$ with $i + j \leqslant \mathsf{p}(\lambda)$ has index

$$n \leqslant \sum_{k=1}^{\mathsf{p}(\lambda)-1} k = \frac{\mathsf{p}(\lambda)(\mathsf{p}(\lambda)-1)}{2} \leqslant \mathsf{p}^2(\lambda).$$

I.e. executions of the hybrid $H^{\dashv}_{\mathsf{p}^2(\lambda)}$ and the ideal experiment $\mathsf{EXEC}(\mathcal{F}_{\text{M-SC}}, \mathcal{S}_{\text{M-SC}})$ are trivially perfectly indistinguishable for $\mathcal{Z}$ as they do not differ before the $\mathsf{p}(\lambda)$-th activation. Hence we know by above considerations that

$$H^{-} \sim_{\mathcal{Z}} H^{\dashv}_{\mathsf{p}^2(\lambda)} \sim_{\mathcal{Z}} \mathsf{EXEC}(\mathcal{F}_{\text{M-SC}}, \mathcal{S}_{\text{M-SC}}),$$

and by finite transitivity of indistinguishability our middle hybrid H^{-} is indistinguishable from the ideal experiment.

With these two steps again finite transitivity of computational indistinguishability concludes our proof of Theorem 5. □

In this chapter we considered the question how weak the components of hybrid encryption are allowed to be to still gain secure communication from given authenticated channels. We transferred the sender-binding concept from Chapter 3 to KEM-DEM hybrid encryption, developing sender-binding key encapsulation with a respective IND-SB-CPA security notion. This sender-binding key encapsulation notion is not only strong enough to facilitate IND-SB-CPA secure sender-binding encryption if combined with the weakest possible data encapsulation, gaining secure message transfer via the UC realisation from Section 3.3. It is also strong enough in combination with IND-CPA secure data encapsulation to UC-realise secure channels if we consider KEM-DEM hybrid encryption in its session communication flavour.

My new secure texting app only allows people named Alice to send messages to people named Bob.

Randall Munroe, xkcd #2691

CHAPTER 5
Generic Transformations

In this chapter we generically construct IND-SB-CPA secure sender-binding encryption as well as sender-binding key encapsulation from various established security notions. These include IND-RCCA classic public-key encryption, and IND-sID-CPA secure identity-based encryption and IND-CPA secure dual receiver encryption and key encapsulation. The generic transformations will broaden our intuitive understanding of the new IND-SB-CPA security notions as well as—in the case of dual receiver encryption—provide a background for some concrete efficient constructions I discuss in Chapter 7. Some of the transformations from this chapter also provide us with counterexamples for when we look at the theoretic relations between different security notions in Chapter 6.

We firstly look at transformations to sender-binding encryption in Section 5.1 before moving on to sender-binding key encapsulation in Section 5.2.

I have previously published insights from this chapter in [BGMOS22] and [SBB+23b], respectively.

5.1 IND-SB-CPA Secure Sender-binding Encryption

There are several ways to generically transform other types of encryption into a sender-binding encryption scheme. Remember that in addition to the transformations I provide in this section, sender-binding encryption can be trivially constructed from tag-based encryption as described in Section 3.2. We will later see in Section 6.1.1 which tag-based encryption security notions are strong enough to imply IND-SB-CPA security in this case. Together with the transformation from classic public-key encryption—following directly in Section 5.1.1—this shows that previous efforts to realise secure message transfer from authenticated channels (cf. Section 3.1) uniformly construct an IND-SB-CPA secure sender-binding encryption scheme.

After briefly looking at classic public-key encryption I will discuss a generic transformation based on IND-CPA secure sound dual receiver encryption in Section 5.1.2. This is not only the technically most interesting transformation in this section, but also the basis for all current attempts to provide concrete efficient constructions of an IND-SB-CPA secure sender-binding encryption scheme. Those constructions are discussed in Chapter 7.

Lastly, we see in Section 5.1.3 that IND-SB-CPA secure sender-binding encryption can also be generically constructed from IND-sID-CPA secure identity-based encryption.

5.1.1 From IND-RCCA Secure Classic Public-key Encryption

As explained in Section 3.1, IND-RCCA variants are one of the previously weakest security notions used in conjunction with authenticated channels to UC-realise secure message transfer. For this section we take "IND-RCCA" to be the weakest among the various IND-RCCA definitions from [BMPR20] (cf. Section 2.1.1) and show that this is sufficient to easily construct an IND-SB-CPA secure sender-binding encryption scheme. This means in particular that all stronger definitions of IND-RCCA as well as IND-CCA2 (which implies IND-RCCA) can be used to achieve IND-SB-CPA security via this construction.

So let $(\mathtt{gen}, \mathtt{enc}, \mathtt{dec})$ be a public-key encryption scheme. We define a new sender-binding encryption scheme $(\mathtt{Gen}, \mathtt{Enc}, \mathtt{Dec})$ via:

$$\begin{aligned}
&\mathtt{Gen} \equiv \mathtt{gen} \\
&\mathtt{Enc}\colon\ (pk_R, S, m) \mapsto \mathtt{enc}(pk_R, m\|S) \\
&\mathtt{Dec}\colon\ (sk_R, S, c) \mapsto \begin{cases} \bot, \ \nexists m\colon \mathtt{dec}(sk_R, c) = m\|S \\ m, \ \mathtt{dec}(sk_R, c) = m\|S \end{cases}
\end{aligned}$$

By encrypting the sender's ID together with the message, the receiver can check on decryption whether this corresponds to the sender they expected or not. Of course, any malicious sender can insert any party ID they want into the ciphertext, but intuitively this construction guarantees that encrypted messages can not just be copied and used by another party as if they had encrypted it themselves. This is exactly what we need for IND-SB-CPA security.

Lemma 3:
If the public-key encryption scheme $(\mathtt{gen}, \mathtt{enc}, \mathtt{dec})$ *is IND-RCCA secure, then the sender-binding encryption scheme* $(\mathtt{Gen}, \mathtt{Enc}, \mathtt{Dec})$ *is IND-SB-CPA secure.*

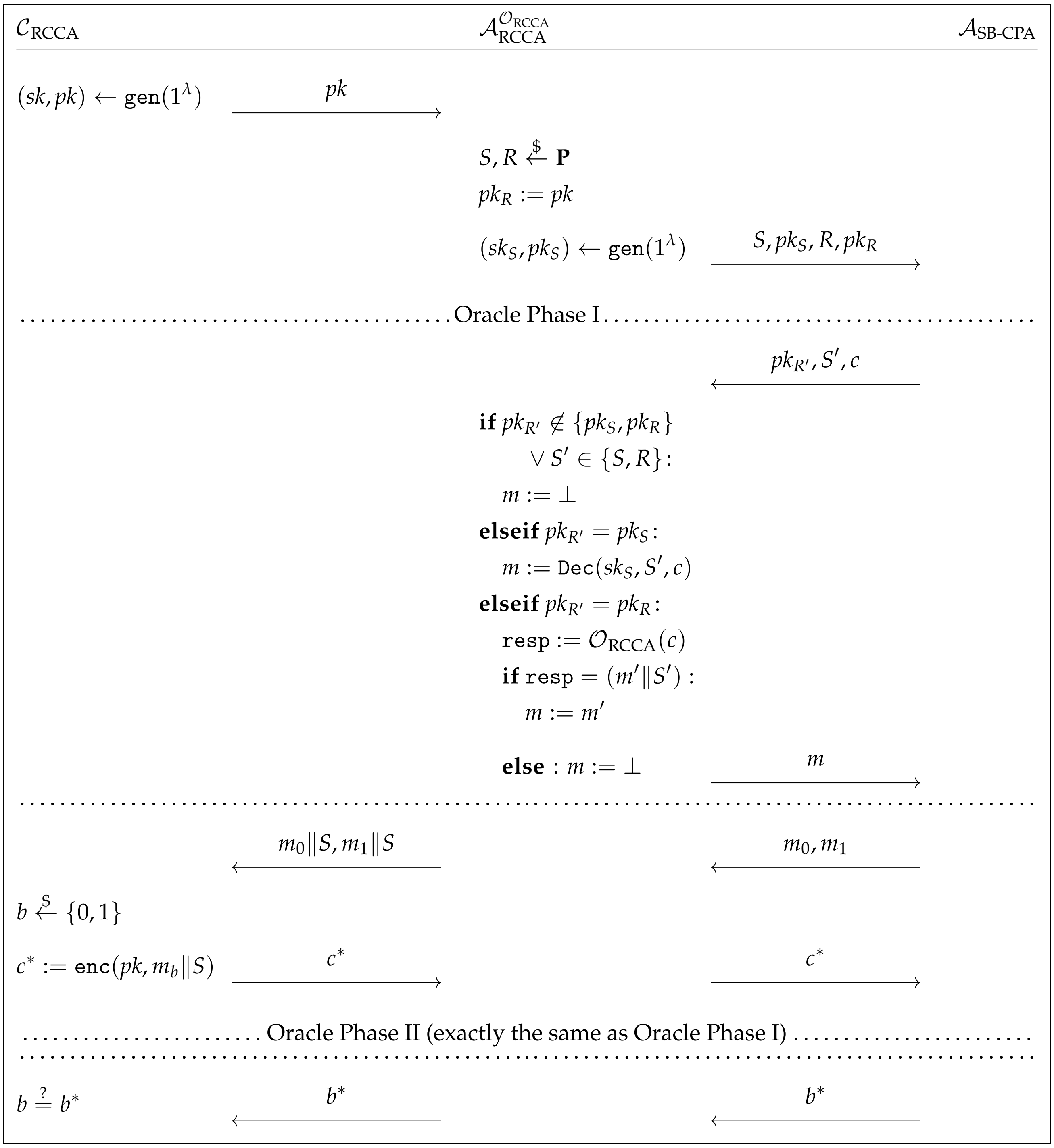
Proof:
Assume that $\mathcal{A}_{\text{SB-CPA}}$ is an adversary which has non-negligible success probability in winning the IND-SB-CPA game with respect to the sender-binding encryption scheme $(\mathtt{Gen}, \mathtt{Enc}, \mathtt{Dec})$. With this we construct an adversary $\mathcal{A}_{\text{RCCA}}$ for the IND-RCCA game with respect to $(\mathtt{gen}, \mathtt{enc}, \mathtt{dec})$ as shown in Figure 19.

We still need to show that the responses from $\mathcal{A}_{\text{RCCA}}$ to $\mathcal{A}_{\text{SB-CPA}}$ are indistinguishable from the responses $\mathcal{O}_{\text{SB-CPA}}$ would give. For the first oracle phase, this is easy to see as $\mathcal{A}_{\text{RCCA}}$ does exactly the same as $\mathcal{O}_{\text{SB-CPA}}$ would. The only difference is that messages to R get decrypted with the help of $\mathcal{O}_{\text{RCCA}}$. Nevertheless, this decryption exactly amounts to

$$m := \mathtt{Dec}(sk_{R'}, S', c).$$

Although Oracle Phase II is identical to Oracle Phase I in the behaviour of $\mathcal{A}_{\text{RCCA}}$, there is actually a difference in the responses of $\mathcal{O}_{\text{RCCA}}$ in case $pk_{R'} = pk_R$: If c is an encryption of one of the challenge messages $(m_0\|S)$, $(m_1\|S)$, the oracle $\mathcal{O}_{\text{RCCA}}(c)$ will reply with $\mathtt{test}$ rather than a decrypted message. Note, however, that since $S' \neq S$ in this case, a ciphertext c containing $(m_b\|S)$ would yield $\mathtt{Dec}(sk_{R'}, S', c) = \bot$. This is exactly the answer $\mathcal{A}_{\text{RCCA}}$ gives to $\mathcal{A}_{\text{SB-CPA}}$. Therefore, there is no difference in the output of the first or second oracle phase to $\mathcal{O}_{\text{SB-CPA}}$ in the view of $\mathcal{A}_{\text{SB-CPA}}$.

Please note that $\mathcal{A}_{\text{RCCA}}$ has exactly the same success probability as $\mathcal{A}_{\text{SB-CPA}}$, which is non-negligible by assumption. □


$\mathcal{C}_{\text{RCCA}}$
$\mathcal{A}_{\text{RCCA}}^{\mathcal{O}_{\text{RCCA}}}$
$\mathcal{A}_{\text{SB-CPA}}$
pk
S, pk_S, R, pk_R
Oracle Phase I
$pk_{R'}, S', c$
m
$m_0\|S, m_1\|S$
m_0, m_1
c^*
c^*
Oracle Phase II (exactly the same as Oracle Phase I)
b^*
b^*

```
$\mathcal{C}_{\text{RCCA}}$:
  $(sk, pk) \leftarrow \texttt{gen}(1^\lambda)$

$\mathcal{A}_{\text{RCCA}}^{\mathcal{O}_{\text{RCCA}}}$:
  $S, R \xleftarrow{\$} \mathbf{P}$
  $pk_R := pk$
  $(sk_S, pk_S) \leftarrow \texttt{gen}(1^\lambda)$

Oracle Phase I:
  if $pk_{R'} \notin \{pk_S, pk_R\}$
      $\lor S' \in \{S, R\}$:
    $m := \bot$
  elseif $pk_{R'} = pk_S$:
    $m := \texttt{Dec}(sk_S, S', c)$
  elseif $pk_{R'} = pk_R$:
    $\texttt{resp} := \mathcal{O}_{\text{RCCA}}(c)$
    if $\texttt{resp} = (m' \| S')$ :
      $m := m'$
    else : $m := \bot$

$\mathcal{C}_{\text{RCCA}}$:
  $b \xleftarrow{\$} \{0, 1\}$
  $c^* := \texttt{enc}(pk, m_b \| S)$

  $b \overset{?}{=} b^*$
```

Figure 19: Reduction from IND-SB-CPA for SBE to IND-RCCA for PKE

5.1.2 From IND-CPA Secure Sound Dual Receiver Encryption

In this section we generically construct an IND-SB-CPA secure sender-binding encryption scheme from dual receiver encryption. This dual receiver encryption transformation provides the background for all concrete sender-binding encryption constructions we discuss in Section 7.1. We furthermore use the transformation in Section 6.1.1 to show that IND-SB-CPA does not in fact imply IND-gtag-CCA but is a strictly weaker security notion.

Originally meant to encrypt a message to two receivers, we use dual receiver encryption in such a way that one of those ciphertexts is encrypted for the sender itself. This, together with a public key infrastructure using key registration with knowledge, results in an encryption where the sender necessarily knows the plaintext. Without key registration with knowledge there is no guarantee that the sender owns the private key corresponding to his public key, so plaintext knowledge could not be guaranteed.

We require the underlying dual receiver encryption scheme to be sound, IND-CPA secure and compatible with the key registration functionality $\mathcal{F}_{\text{KRK}}$, which is taken care of by the following remark. For the definition of dual receiver encryption, its soundness, and the definition of $\mathcal{F}_{\text{KRK}}$ please refer back to Section 2.1.1 and Section 2.2.2 respectively.

Remark 4:
Throughout this section we assume dual receiver encryption schemes to permit efficiently computable Boolean functions $\mathtt{f}_{\text{Key}}$. On input of a (possible) key pair (sk, pk) this function decides whether the keys "belong together", i.e. whether they could have been output by the encryption scheme's key generation algorithm gen or might just be an unrelated pair of values:

$$\mathtt{f}_{\text{Key}} : (sk, pk) \mapsto \begin{cases} \mathsf{true}, & (sk, pk) \leftarrow \mathtt{gen}(1^\lambda) \\ \mathsf{false}, & \text{else.} \end{cases}$$

This is necessary for the scheme to be used in conjunction with the registration functionality $\mathcal{F}_{\text{KRK}}$. In Section 2.2.2 I have discussed $\mathcal{F}_{\text{KRK}}$ a bit more and also explained how we can easily dispose of the need for a function $\mathtt{f}_{\text{Key}}$ if we are happy for the registration functionality to (partially) generate the keys for the registering parties. ◦

Let $(\mathtt{gen}, \mathtt{enc}, \mathtt{dec})$ be a dual receiver encryption scheme which admits a function $\mathtt{f}_{\text{Key}}$. We define a new encryption scheme $(\mathtt{Gen}, \mathtt{Enc}, \mathtt{Dec})$ via:

$\mathtt{Gen}(1^\lambda)$ executed by party P:
- $(sk, pk) \leftarrow \mathtt{gen}(1^\lambda)$.
- Register (sk, pk) with $\mathcal{F}_{\text{KRK}}^{\mathtt{f}_{\text{Key}}}$.

$\hookrightarrow$ Return $(SK, PK) := ((sk, pk), P)$.

$\mathtt{Enc}(PK_R, S, m) = \mathtt{Enc}(R, S, m)$ executed by party S:
- Retrieve pk_R and pk_S from $\mathcal{F}_{\text{KRK}}^{\mathtt{f}_{\text{Key}}}$.

$\hookrightarrow$ Return $c \leftarrow \mathtt{enc}(pk_R, pk_S, m)$.

$\mathtt{Dec}(SK_R, S, c) = \mathtt{Dec}\big((sk_R, pk_R), S, c\big)$ executed by party R:
- Retrieve pk_S from $\mathcal{F}_{\text{KRK}}^{\mathtt{f}_{\text{Key}}}$.

$\hookrightarrow$ Return $m := \mathtt{dec}(sk_R, pk_R, pk_S, c)$.

At this point I would like to provide some additional explanations of this construction with regards to the use of $\mathcal{F}_{\text{KRK}}$.

Remark 5 (Use of $\mathcal{F}_{\text{KRK}}$):
(1) Note firstly, that the secret key sk is handed to the ideal functionality $\mathcal{F}_{\text{KRK}}$, but is not accessible to any other party. Realisations of $\mathcal{F}_{\text{KRK}}$ without a trusted party would probably use the public key in conjunction with a zero knowledge proof of knowledge of the secret key instead of the key itself. Secret keys are *not* registered to be handed out to other parties.

(2) For any communication partner P, it is sufficient to retrieve the key pk_P once from $\mathcal{F}_{\text{KRK}}$ and then store it internally. There is no need to communicate with $\mathcal{F}_{\text{KRK}}$ again for any communication with P.

(3) The sender's public key pk_S which is retrieved during encryption can also be stored internally after generation to make its retrieval from $\mathcal{F}_{\text{KRK}}$ superfluous.

(4) I would like to point out that the realisation of authenticated channels is usually constructed from a public key infrastructure. The same public key infrastructure can be augmented with zero-knowledge proofs to guarantee key registration with knowledge and the possibly expensive operation for registering a public key and proving the knowledge of the according secret key has to be performed only once at the beginning. ◦

Now let me further give some intuition about the construction before we move on to formalities. Choosing one of the receivers for dual receiver encryption to be the sender themselves and having them encrypt a message under their own key might seem counter intuitive at first, but has one crucial benefit: It guarantees to the other (actual) receiver that even if the sender might not have constructed the ciphertext themselves but rather copied it from somewhere else, they have knowledge about the plaintext since they are able to decrypt as well. This is guaranteed by the registration with $\mathcal{F}_{\text{KRK}}$ in conjunction with the soundness property of the underlying dual receiver encryption scheme.

Lemma 4:
If the dual receiver encryption scheme $(\texttt{gen}, \texttt{enc}, \texttt{dec})$ *is IND-CPA secure and sound, then in the* $\mathcal{F}_{\text{KRK}}^{\texttt{f}_{Key}}$ *hybrid model* $(\texttt{Gen}, \texttt{Enc}, \texttt{Dec})$ *is an IND-SB-CPA secure sender-binding encryption scheme.*

Proof:
Assuming that $(\texttt{gen}, \texttt{enc}, \texttt{dec})$ is a sound dual receiver encryption scheme with key function $\texttt{f}_{\text{Key}}$ and assuming we have an adversary $\mathcal{A}_{\text{SB-CPA}}$ who has non-negligible success probability in winning the IND-SB-CPA game with respect to $(\texttt{Gen}, \texttt{Enc}, \texttt{Dec})$, we construct an adversary $\mathcal{A}_{\text{CPA}}$ with non-negligible success probability in winning the dual receiver encryption IND-CPA game with respect to $(\texttt{gen}, \texttt{enc}, \texttt{dec})$. Note that in this case, $\mathcal{A}_{\text{CPA}}$ not only fields $\mathcal{A}_{\text{SB-CPA}}$'s queries to $\mathcal{O}_{\text{SB-CPA}}$ but also plays the role of $\mathcal{F}_{\text{KRK}}$ and has therefore access to registered keys. In the reduction shown in Figure 20 we do not explicitly state this, but all interactions with $\mathcal{F}_{\text{KRK}}$ are handled exactly as the functionality itself would. The only exceptions are that an instantaneous `ok` is assumed whenever the functionality would ask the adversary for some permission and that in the first phase the adversary $\mathcal{A}_{\text{CPA}}$ itself "registers" the keys pk_S and pk_R for S and R respectively without providing corresponding secret keys.

Since $\mathcal{A}_{\text{CPA}}$ has access to the internal state of $\mathcal{F}_{\text{KRK}}$, they can look up the keys $(sk_{S'}, pk_{S'})$ for any oracle query (R', S', c). If no such keys have been registered, decryption of the ciphertext would result in $\perp$. If keys have been registered, they can be used to correctly decrypt the ciphertext as the soundness of dual receiver encryption (see Section 2.1.1 for definition) guarantees

$$\texttt{dec}(sk_{S'}, pk_{S'}, pk_{R'}, c) = \texttt{dec}(sk_{R'}, pk_{R'}, pk_{S'}, c).$$

Hence it is no problem for $\mathcal{A}_{\text{CPA}}$ to respond with correct decryptions exactly as $\mathcal{O}_{\text{SB-CPA}}$ would. This gives $\mathcal{A}_{\text{CPA}}$ the same non-negligible success probability as $\mathcal{A}_{\text{SB-CPA}}$. □

$\mathcal{C}_{\text{CPA}}$		$\mathcal{A}_{\text{CPA}}$		$\mathcal{A}_{\text{SB-CPA}}$
$(sk_1, pk_1) \leftarrow \mathtt{gen}(1^\lambda)$				
$(sk_2, pk_2) \leftarrow \mathtt{gen}(1^\lambda)$	$\xrightarrow{pk_1, pk_2}$			
		$S, R \xleftarrow{\$} \mathbf{P}$		
		$(pk_S, pk_R) := (pk_1, pk_2)$		
		$\mathcal{F}^{\mathtt{f}_{\text{Key}}}_{\text{KRK}} \overset{\text{reg.}}{\leftarrow} pk_S, pk_R$	$\xrightarrow{S, S, R, R}$	
		.... Oracle Phase I		
			$\xleftarrow{R', S', c}$	
		if $R' \notin \{S, R\} \vee S' \in \{S, R\}$:		
		$m := \perp$		
		else :		
		$(sk_{S'}, pk_{S'}) \leftarrow \mathcal{F}^{\mathtt{f}_{\text{Key}}}_{\text{KRK}}(S')$		
		$m := \mathtt{dec}(sk_{S'}, pk_{S'}, pk_{R'}, c)$	$\xrightarrow{m}$	
		....		
	$\xleftarrow{m_0, m_1}$		$\xleftarrow{m_0, m_1}$	
$b \xleftarrow{\$} \{0,1\}$				
$c^* := \mathtt{enc}(pk_R, pk_S, m_b)$	$\xrightarrow{c^*}$		$\xrightarrow{c^*}$	
		.... Oracle Phase II (exactly the same as Oracle Phase I)		
		....		
$b \overset{?}{=} b^*$	$\xleftarrow{b^*}$		$\xleftarrow{b^*}$	

Figure 20: Reduction from IND-SB-CPA for SBE to IND-CPA for DRE

Having proven that the above construction yields IND-SB-CPA secure sender-binding encryption, let us discuss the possibility to weaken the assumption of soundness in the underlying dual receiver encryption and the connection between our construction an the concept of plaintext awareness.

Remark 6 (Weakly Decryption Consistent DRE):
Using dual receiver encryption in the asymmetric fashion of our construction—where one of the receivers is the sender itself—we notice that the soundness of dual receiver encryption is more than we require: We would be perfectly happy for soundness to hold *only* when the first receiver is able to successfully decrypt. I.e. if the first receiver outputs $\perp$ on decryption of c, we do not care whether the second receiver (who in the above sender-binding encryption construction is the sender) outputs $\perp$ as well or decrypts the ciphertext c to some valid message m.

This weaker notion of dual receiver encryption soundness is equivalent to so called *weak decryption consistency* [NHKJ12]. Note that (particularly) in this case, decryption algorithms between the first and second receiver might differ, i.e. a weakly decryption consistent dual receiver encryption scheme might be given by PPT algorithms $(\mathtt{gen}, \mathtt{enc}, \mathtt{dec}, \mathtt{dec}')$. It is easy to see that with the above construction of sender-binding encryption from a dual receiver encryption scheme, we already achieve IND-SB-CPA security from an IND-CCVA2 secure and weakly decryption consistent dual receiver encryption.

Such an IND-CCVA2 secure weakly decryption consistent dual receiver encryption scheme can be constructed by taking an IND-CCA2 secure public-key encryption scheme $(\mathtt{gen}, \mathtt{enc}, \mathtt{dec})$, where $\mathtt{enc}$ is deterministic with the last argument representing the randomness used. From this we define a dual receiver encryption scheme $(\mathtt{Gen}, \mathtt{Enc}, \mathtt{Dec}, \mathtt{Dec}')$ via:

$$\begin{aligned}
&\mathtt{Gen} \equiv \mathtt{gen} \\
&\mathtt{Enc}: \qquad (pk, pk', m) \mapsto \Big(\mathtt{enc}\big(pk, (m,s); r\big), \mathtt{enc}\big(pk', m; s\big)\Big) \\
&\mathtt{Dec}: \quad \big(sk, pk, pk', (c, c')\big) \mapsto \begin{cases} \bot, & c' \neq \mathtt{enc}(pk', m; s) \\ m, & \text{otherwise} \end{cases} \\
&\qquad \text{where } (m,s) := \mathtt{dec}(sk, c) \\
&\mathtt{Dec}': \; \big(sk', pk, pk', (c, c')\big) \mapsto \mathtt{dec}(sk', c').
\end{aligned}$$

Right now, I am not aware of any IND-CCVA2 secure partially sound dual receiver encryption construction which does not employ IND-CCA2 secure public-key encryption. Since there are easier ways to construct a IND-SB-CPA secure sender-binding encryption from IND-CCA2 (e.g. Section 5.1.1) I will not go into this weaker dual receiver encryption notion any further. ◦

Remark 7 (Connection to Plaintext Awareness):
The way we use dual receiver encryption to get a sender-binding encryption scheme is somewhat related to the concept of plaintext awareness (cf. [BR95; BP04]). The (non-equivalent) notions from these two papers do not intrinsically satisfy IND-SB-CPA, however. They focus on the property that knowledge about all oracle queries made by the adversary, or respectively the adversary's input and randomness, is enough to permit the existence of an efficient plaintext extractor. Which is not sufficient to prevent message stealing attacks or satisfy IND-SB-CPA security. The key difference is that an adversary effectively has knowledge about the plaintext in any valid ciphertext it can *construct* from only the public key of the receiver. For IND-SB-CPA on the other hand, the adversary would have to be plaintext-aware about any ciphertext that a party under their control could validly send to the receiver—regardless of who constructed it or where it came from. Note that from two of the plaintext awareness definitions in [BR95; BP04], IND-SB-CPA can be constructed anyway (cf. Section 5.1.1) as they imply IND-CCA2 which in turn implies IND-RCCA.

The notion of registration-based plaintext awareness introduced by [HLM03], on the other hand, is a lot closer to the sender-binding encryption construction I presented in this section. It states that if an adversary can successfully construct a ciphertext c from some party P to R, then the adversary's knowledge about the key registration of P is enough (together with public keys) to decrypt c. This does not only prohibit message stealing attacks and implies IND-SB-CPA in case the oracle has the same knowledge about key registration that the adversary has (as in the above dual receiver encryption construction). It also implies another property which is irrelevant for IND-SB-CPA: If the adversary has no knowledge about the key registration of a party (e.g. for the party S in the IND-SB-CPA game), then they can either not construct a valid ciphertext from this party to R, or the ciphertext they constructed can already be decrypted with public keys only. IND-SB-CPA security does not require this. ◦

5.1.3 From IND-sID-CPA Secure Identity-based Encryption

Let $(\texttt{gen}, \texttt{ext}, \texttt{enc}, \texttt{dec})$ be an identity-based encryption scheme. Furthermore let $\texttt{f}_{\text{ID}}$ be the (publicly known) function of the parties' identity-based encryption IDs. We assume this function to be efficiently computable and injective (i.e. no two parties share the same ID). We define a new sender-binding encryption scheme $(\texttt{Gen}, \texttt{Enc}, \texttt{Dec})$ by:

$$\begin{aligned} &\texttt{Gen} \equiv \texttt{gen} \text{ (i.e. } (SK, PK) := (msk, mpk)) \\ &\texttt{Enc}: \quad (PK_R, S, m) = (mpk_R, S, m) \mapsto \texttt{enc}\Big(mpk_R, \texttt{f}_{\text{ID}}(S), m\Big) \\ &\texttt{Dec}: \quad (SK_R, S, c) = (msk_R, S, c) \mapsto \texttt{dec}\Big(\texttt{ext}(msk_R, \texttt{f}_{\text{ID}}(S)), c\Big). \end{aligned}$$

In this construction we use the underlying identity-based encryption scheme "the wrong way round", generating one instance of the scheme for each receiver. This intuitively corresponds to the simpler and less efficient idea of having the receiver use a different key pair for each sender (which would also satisfy IND-SB-CPA security). An interesting question would be whether we could construct a kind of double identity-based encryption scheme, where each ID can be used to extract a master secret key for a complete identity-based encryption scheme, i.e.

$$(msk_R, mpk_R) = \Big(\texttt{ext}_{\text{master}}\big(\texttt{f}_{\text{ID}}(R)\big), \texttt{f}_{\text{ID}}(R)\Big)$$

as well as to extract individual user secret keys via

$$\texttt{ext}\Big(msk_R, \texttt{f}_{\text{ID}}(S)\Big).$$

Unfortunately this goes beyond the scope of this dissertation, so let us instead satisfy ourselves that we can actually achieve IND-SB-CPA security by the above construction:

Lemma 5:
If the identity-based encryption scheme $(\texttt{gen}, \texttt{ext}, \texttt{enc}, \texttt{dec})$ *is IND-sID-CPA secure, the sender-binding encryption scheme* $(\texttt{Gen}, \texttt{Enc}, \texttt{Dec})$ *is IND-SB-CPA secure.*

Proof:
Assuming an adversary $\mathcal{A}_{\text{SB-CPA}}$ who has non-negligible probability in winning the IND-SB-CPA game with respect to $(\texttt{Gen}, \texttt{Enc}, \texttt{Dec})$, we construct an adversary $\mathcal{A}_{\text{sID-CPA}}$ with non-negligible success probability in winning the IND-sID-CPA game with respect to $(\texttt{gen}, \texttt{ext}, \texttt{enc}, \texttt{dec})$ as shown in Figure 21.

In this example the behaviour in both oracle phases is again identical and corresponds to the same steps $\mathcal{O}_{\text{SB-CPA}}$ would take on input $(mpk_{R'}, S', c)$. To decrypt messages sent to R, $\mathcal{A}_{\text{sID-CPA}}$ enlists the help of $\mathcal{O}_{\text{sID-CPA}}$ instead of extracting $usk_{S'}$ itself. $\mathcal{O}_{\text{sID-CPA}}$ will always provide $\mathcal{A}_{\text{sID-CPA}}$ with the user secret key $usk_{S'}$, as it was checked earlier that $S' \notin \{S, R\}$ and therefore $\texttt{f}_{\text{ID}}(S') \neq id_S$.

Note that $\mathcal{A}_{\text{sID-CPA}}$ has exactly the same success probability as $\mathcal{A}_{\text{SB-CPA}}$—which is non-negligible by assumption—because messages, challenge and the bit b^* are all just forwarded.□

Remark 8 (Weaker IBE):
Looking closely at the above reduction we notice that a weaker notion than IND-sID-CPA security would suffice: The ability for $\mathcal{A}_{\text{sID-CPA}}$ to choose the sender's identity-based encryption ID id_S is completely unnecessary.[10] If the sender ID was chosen by the challenger $\mathcal{C}_{\text{sID-CPA}}$

[10] Note that "sender" in this case denotes the *receiving* party in the classic identity-based encryption scheme. For this argument I will continue to call this party "sender" and "S" to hopefully increase readability and avoid unnecessary ambiguity.

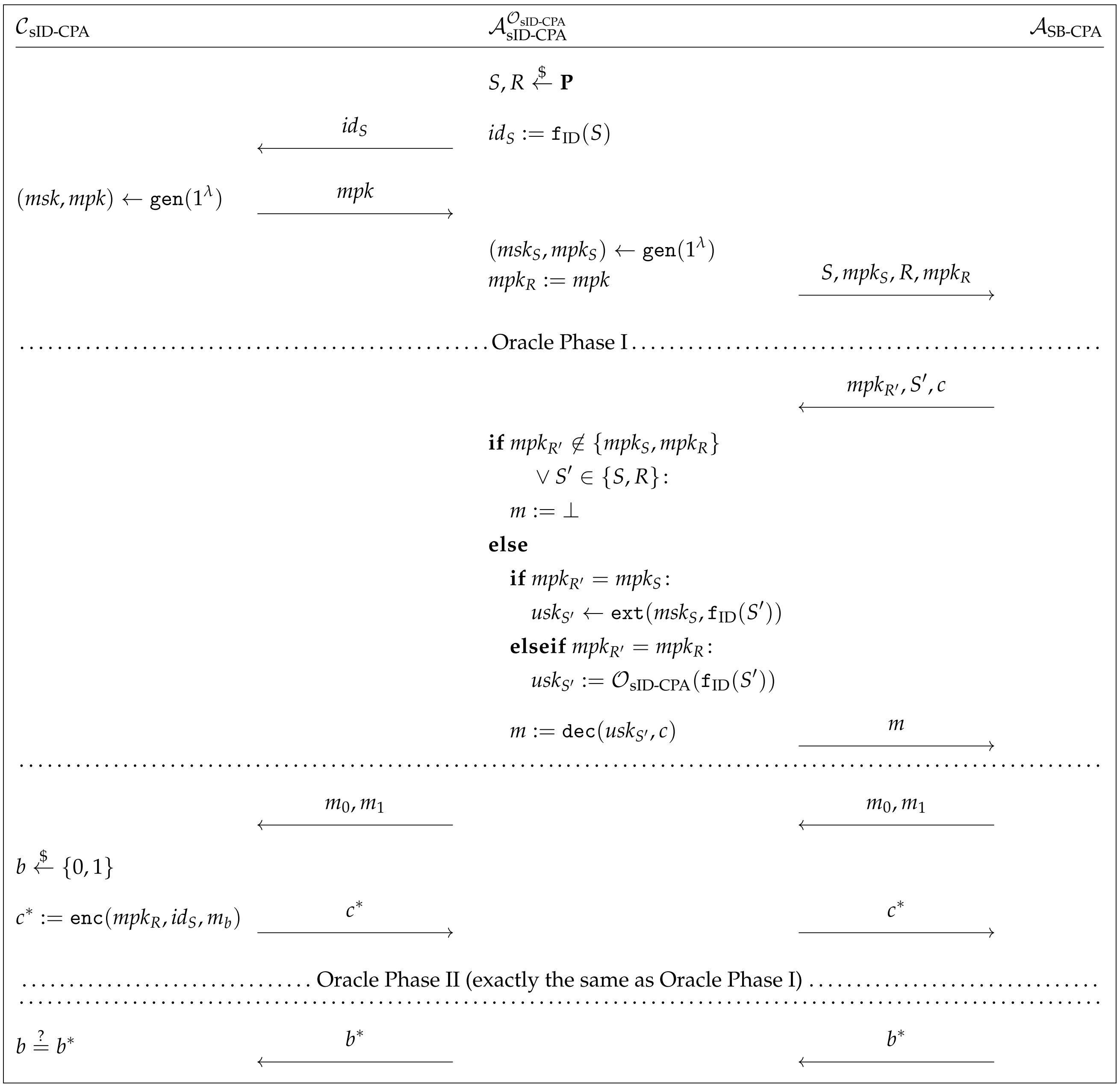

$\mathcal{C}_{\text{sID-CPA}}$
$\mathcal{A}_{\text{sID-CPA}}^{\mathcal{O}_{\text{sID-CPA}}}$
$\mathcal{A}_{\text{SB-CPA}}$
$S, R \xleftarrow{\$} \mathbf{P}$
id_S
$id_S := \mathtt{f}_{\text{ID}}(S)$
$(msk, mpk) \leftarrow \mathtt{gen}(1^\lambda)$
mpk
$(msk_S, mpk_S) \leftarrow \mathtt{gen}(1^\lambda)$
$mpk_R := mpk$
S, mpk_S, R, mpk_R
Oracle Phase I
$mpk_{R'}, S', c$
if $mpk_{R'} \notin \{mpk_S, mpk_R\}$
$\vee S' \in \{S, R\}$:
$m := \perp$
else
if $mpk_{R'} = mpk_S$:
$usk_{S'} \leftarrow \mathtt{ext}(msk_S, \mathtt{f}_{\text{ID}}(S'))$
elseif $mpk_{R'} = mpk_R$:
$usk_{S'} := \mathcal{O}_{\text{sID-CPA}}(\mathtt{f}_{\text{ID}}(S'))$
$m := \mathtt{dec}(usk_{S'}, c)$
m
m_0, m_1
m_0, m_1
$b \xleftarrow{\$} \{0,1\}$
$c^* := \mathtt{enc}(mpk_R, id_S, m_b)$
c^*
c^*
Oracle Phase II (exactly the same as Oracle Phase I)
$b \overset{?}{=} b^*$
b^*
b^*

Figure 21: Reduction from IND-SB-CPA for SBE to IND-sID-CPA for IBE

together with the parameters (msk, mpk) at the start of the game, the reduction would work equally well. This is not only because the credentials (msk_S, mpk_S) are independent of the sender's ID and could still be chosen by $\mathcal{A}_{\text{sID-CPA}}$. The main reason is that we only need $\mathcal{A}_{\text{sID-CPA}}$ to have as much power over choosing the sender and its credentials as $\mathcal{A}_{\text{SB-CPA}}$ has in the IND-SB-CPA game. Digressing to some of the notation I will later introduce in Section 6.3: From any identity-based encryption scheme which is IND-set-CPA for choosing mode

$$\text{set} = (\mathcal{P}, \textit{time}, \text{Receiver}),$$

we get an IND-set′-SB-CPA secure sender-binding encryption scheme with choosing mode

$$\text{set}' = (\mathcal{P}, \textit{time}, \text{Sender})$$

from the above construction. In particular, an

$$\text{IND-}\Big(\mathcal{C}, \text{Start}, \big((msk, mpk), \text{Receiver } R, (msk_R, mpk_R)\big)\Big)\text{-CPA}$$

secure identity-based encryption scheme with the respective extraction oracles suffices to construct IND-SB-CPA secure sender-binding encryption. ○

5.2 IND-SB-CPA Secure Sender-binding Key Encapsulation

Unlike with sender-binding and classic public-key encryption, it is not immediately possible to construct a sender-binding key encapsulation mechanism from classic key encapsulation. This is because encapsulation does not take any variable input we could divert from its intended use to also input a sender ID, as we can do with messages for public-key encryption. If we did enhance the only input—the public key—by the sender's ID in such a way that encapsulation remained secure and such that the ID would be checkable upon decapsulation, we would be back with the (inefficient) idea that each receiver uses a different key pair for each possible sender.

On the other hand, the transformation from tag-based key encapsulation to sender-binding key encapsulation is so straightforward again that I decided to only include it in Section 6.2, where I discuss which tag-based key encapsulation notions imply IND-SB-CPA security.

In this section I will only briefly show how an IND-SB-CPA secure sender-binding key encapsulation mechanism can be constructed via IND-CPA secure and sound dual receiver key encapsulation. This transformation works very similarly to the corresponding sender-binding encryption transformation in Section 5.1.2.

5.2.1 From IND-CPA Secure Sound Dual Receiver Key Encapsulation

To use a dual receiver key encapsulation mechanism in conjunction with the key registration functionality $\mathcal{F}_{\text{KRK}}$, we assume it to permit an efficiently computable Boolean function $\mathtt{f}_{\text{Key}}$ (cf. Remark 4 in Section 5.1.2). Hence let $(\mathtt{gen}, \mathtt{enc}, \mathtt{dec})$ be a dual receiver key encapsulation mechanism with key function $\mathtt{f}_{\text{Key}}$. We define a new sender-binding key encapsulation mechanism $(\mathtt{Gen}, \mathtt{Enc}, \mathtt{Dec})$ via:

```
$\mathtt{Gen}(1^\lambda)$ executed by party $P$:
  • $(sk, pk) \leftarrow \mathtt{gen}(1^\lambda)$.
  • Register $(sk, pk)$ with $\mathcal{F}_{\text{KRK}}^{\mathtt{f}_{\text{Key}}}$.
  $\hookrightarrow$ Return $(SK, PK) := ((sk, pk), P)$.

$\mathtt{Enc}(PK_R, S) = \mathtt{Enc}(R, S)$ executed by party $S$:
  • Retrieve $pk_R$ and $pk_S$ from $\mathcal{F}_{\text{KRK}}^{\mathtt{f}_{\text{Key}}}$.
  $\hookrightarrow$ Return $(K, C) \leftarrow \mathtt{enc}(pk_R, pk_S)$.

$\mathtt{Dec}(SK_R, S, C) = \mathtt{Dec}((sk_R, pk_R), S, C)$ executed by party $R$:
  • Retrieve $pk_S$ from $\mathcal{F}_{\text{KRK}}^{\mathtt{f}_{\text{Key}}}$.
  $\hookrightarrow$ Return $K := \mathtt{dec}(sk_R, pk_R, pk_S, C)$.
```

The intuition behind this construction is the same as when a sender-binding encryption scheme is constructed from dual receiver encryption: By encapsulating the key such that both sender and receiver may decapsulate it with their respective secret keys, soundness of the dual receiver key encapsulation guarantees to the receiver that the sender has knowledge of the key regardless of who might have constructed the ciphertext C.

Lemma 6:
If the dual receiver key encapsulation mechanism $(\texttt{gen}, \texttt{enc}, \texttt{dec})$ *is IND-CPA secure and sound, then in the* $\mathcal{F}_{KRK}^{\mathtt{f}_{Key}}$ *hybrid model* $(\texttt{Gen}, \texttt{Enc}, \texttt{Dec})$ *is an IND-SB-CPA secure sender-binding key encapsulation mechanism.*

Proof:
We conduct the proof by contradiction. Let $(\texttt{gen}, \texttt{enc}, \texttt{dec})$ be a sound dual receiver key encapsulation mechanism with key function $\mathtt{f}_{\text{Key}}$ and $\mathcal{A}_{\text{SB-CPA}}$ be an adversary which has non-negligible success probability in winning the IND-SB-CPA game for sender-binding key encapsulation with respect to $(\texttt{Gen}, \texttt{Enc}, \texttt{Dec})$. We use $\mathcal{A}_{\text{SB-CPA}}$ to construct an adversary $\mathcal{A}_{\text{CPA}}$ with non-negligible success probability in winning the IND-CPA game for dual receiver key encapsulation mechanisms with respect to $(\texttt{gen}, \texttt{enc}, \texttt{dec})$. The detailed reduction is shown in Figure 22.

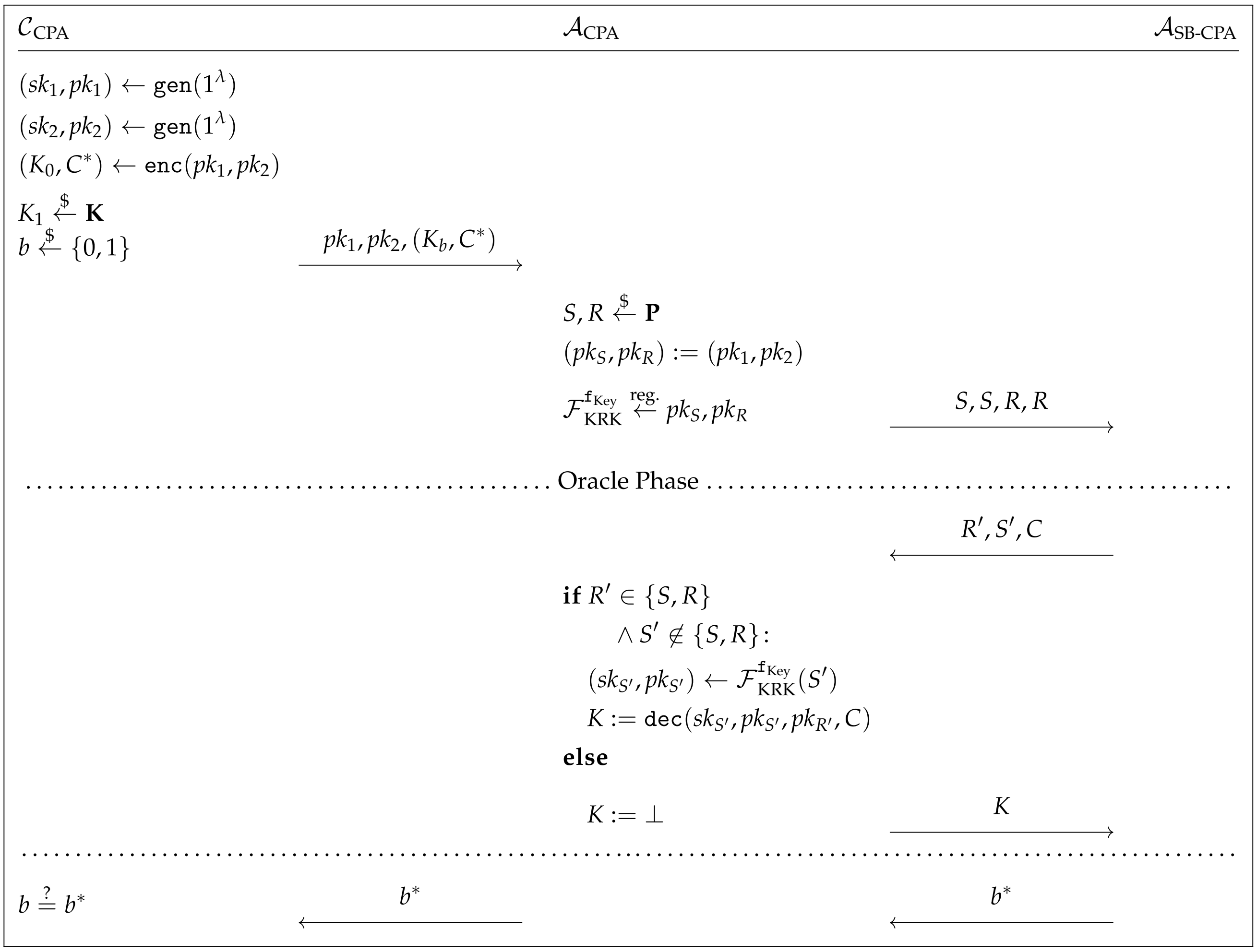

$\mathcal{C}_{\text{CPA}}$
$\mathcal{A}_{\text{CPA}}$
$\mathcal{A}_{\text{SB-CPA}}$
$(sk_1, pk_1) \leftarrow \texttt{gen}(1^\lambda)$
$(sk_2, pk_2) \leftarrow \texttt{gen}(1^\lambda)$
$(K_0, C^*) \leftarrow \texttt{enc}(pk_1, pk_2)$
$K_1 \overset{\$}{\leftarrow} \mathbf{K}$
$b \overset{\$}{\leftarrow} \{0,1\}$
$\xrightarrow{pk_1, pk_2, (K_b, C^*)}$
$S, R \overset{\$}{\leftarrow} \mathbf{P}$
$(pk_S, pk_R) := (pk_1, pk_2)$
$\mathcal{F}_{\text{KRK}}^{\mathtt{f}_{\text{Key}}} \overset{\text{reg.}}{\leftarrow} pk_S, pk_R$
$\xrightarrow{S, S, R, R}$
Oracle Phase
$\xleftarrow{R', S', C}$
if $R' \in \{S, R\}$
$\wedge\, S' \notin \{S, R\}$:
$(sk_{S'}, pk_{S'}) \leftarrow \mathcal{F}_{\text{KRK}}^{\mathtt{f}_{\text{Key}}}(S')$
$K := \texttt{dec}(sk_{S'}, pk_{S'}, pk_{R'}, C)$
else
$K := \bot$
$\xrightarrow{K}$
$b \overset{?}{=} b^*$
$\xleftarrow{b^*}$
$\xleftarrow{b^*}$

Figure 22: Reduction from IND-SB-CPA for SB-KEM to IND-CPA for DR-KEM

Key point in this proof is that while $\mathcal{A}_{\text{CPA}}$ has to provide $\mathcal{A}_{\text{SB-CPA}}$ with the correct answers to any oracle queries it makes, it also acts as $\mathcal{F}_{\text{KRK}}^{\mathtt{f}_{\text{Key}}}$ for $\mathcal{A}_{\text{SB-CPA}}$ and hence has access to any keys $\mathcal{A}_{\text{SB-CPA}}$ registers. Note that we let $\mathcal{A}_{\text{CPA}}$ handle all interactions of $\mathcal{A}_{\text{SB-CPA}}$ with $\mathcal{F}_{\text{KRK}}^{\mathtt{f}_{\text{Key}}}$

exactly as the functionality itself would do, with the exception that instantaneous `ok`'s are assumed whenever the functionality would ask the adversary for permission. Now for any oracle query (R', S', C), $\mathcal{A}_{\text{CPA}}$ looks up the keys $(sk_{S'}, pk_{S'})$ which $\mathcal{A}_{\text{SB-CPA}}$ must have registered for the decapsulation not to fail. Due to the soundness property of the dual receiver key encapsulation mechanism those keys can now be used to correctly decrypt the key

$$K = \texttt{dec}(sk_{S'}, pk_{S'}, pk_{R'}, C) = \texttt{dec}(sk_{R'}, pk_{R'}, pk_{S'}, C)$$

and answer the oracle query. This gives $\mathcal{A}_{\text{CPA}}$ the same non-negligible success probability as $\mathcal{A}_{\text{SB-CPA}}$. □

In this chapter we have discussed various ways to generically transform other forms of public-key encryption and key encapsulation into IND-SB-CPA secure sender-binding encryption and sender-binding key encapsulation, respectively. We have seen that IND-SB-CPA secure sender-binding encryption can be constructed from classic public-key encryption if it satisfies the comparably strong notion of IND-RCCA security as well as from the weakest established dual receiver encryption and identity-based encryption schemes. For sender-binding key encapsulation we looked at an analogous transformation from dual receiver key encapsulation. With both dual receiver transformations we saw that while we only require IND-CPA security, soundness of the respective dual receiver schemes is vital.

Through 20 years of effort, we've successfully trained everyone to use passwords that are hard for humans to remember, but easy for computers to guess.

Randall Munroe, xkcd #936

CHAPTER 6
Theoretic Classification of Security Notions

I have presented the new notion of IND-SB-CPA for sender-binding encryption and sender-binding key encapsulation mechanism in Chapter 3 and Chapter 4, respectively, where I also gave some intuition on what this notion implies as well as shown that it is sufficiently strong to turn authenticated channels into secure communication. I furthermore broadened the intuitive understanding by generic example transformations in Chapter 5. What is still missing from the picture is a formal classification of how the IND-SB-CPA notions directly relate to other game-based security notions.

To fill this gap we firstly examine the sender-binding encryption notion of IND-SB-CPA and its connection to tag-based encryption as well as classical public-key encryption indistinguishability notions ranging from IND-CPA to IND-CCA2 in Section 6.1. I then provide similar considerations for sender-binding key encapsulation version of IND-SB-CPA security in Section 6.2 before I attempt a more holistic classification of game-based security notions in Section 6.3.

I have previously published insights from this chapter in [BGMOS22] and [SBB+23b], respectively.

6.1 Theoretic Classification of IND-SB-CPA for Sender-binding Encryption

Let me start with the sender-binding encryption version of IND-SB-CPA security. I want to compare this to security notions for the two types of encryption schemes that were previously used to turn authenticated channels into secure message transfer: Classic public-key encryption and tag-based encryption. As comparisons are a bit more straightforward there, we firstly look at tag-based encryption security in Section 6.1.1 before I discuss classic public-key encryption notions in Section 6.1.2.

6.1.1 Tag-based Notions Imply IND-SB-CPA

Remember that in Section 3.2, I developed IND-SB-CPA security by firstly weakening the tag-based notion of IND-stag-CCA to IND-gtag-CCA security and subsequently changing some details which are expedient for any sender-binding encryption security notion. I follow a similar path in this section. We firstly look at the formal proofs of when IND-gtag-CCA denotes a strictly weaker notion than IND-stag-CCA. Secondly, we convince ourselves that the changes from IND-gtag-CCA to IND-SB-CPA again constitute a weakening of the security notion.

IND-stag-CCA Implies IND-gtag-CCA

In this section we analyse the implicational relationship between IND-gtag-CCA and IND-stag-CCA security.

Lemma 7 (IND-stag-CCA ⇒ IND-gtag-CCA):
Any IND-stag-CCA secure tag-based encryption scheme automatically satisfies IND-gtag-CCA security.

Proof:
Let (gen, enc, dec) be a tag-based encryption scheme. Under assumption of an efficient adversary $\mathcal{A}_{\text{gtag-CCA}}$ with non-negligible probability to win the IND-gtag-CCA security game, it is very straight forward to construct an efficient adversary $\mathcal{A}_{\text{stag-CCA}}$ who has the same success probability in the IND-stag-CCA game. An overview of the construction can be found in Figure 23.

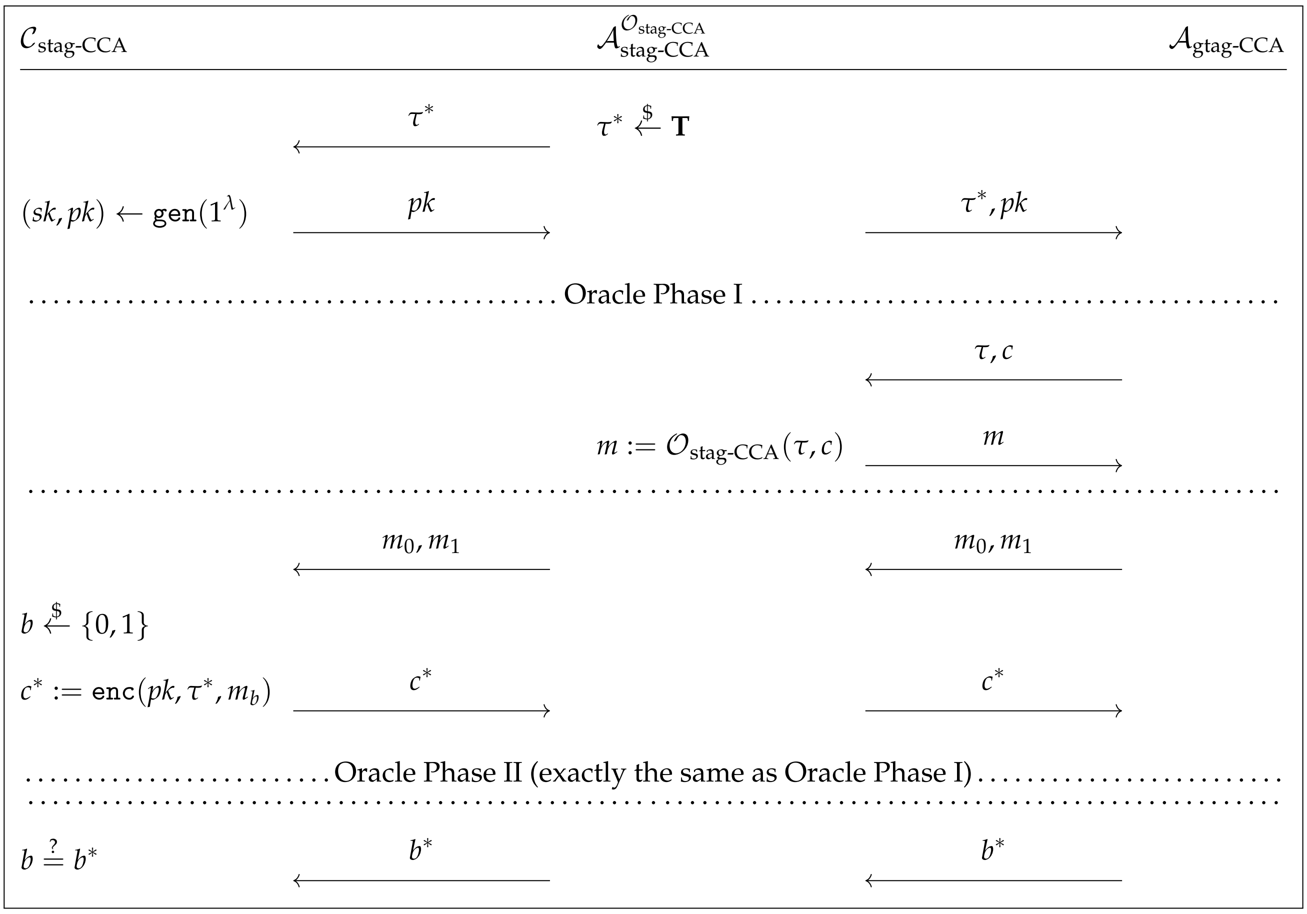

$\mathcal{C}_{\text{stag-CCA}}$
$\mathcal{A}_{\text{stag-CCA}}^{\mathcal{O}_{\text{stag-CCA}}}$
$\mathcal{A}_{\text{gtag-CCA}}$
τ^*
$\tau^* \xleftarrow{\$} \mathbf{T}$
$(sk, pk) \leftarrow \texttt{gen}(1^\lambda)$
pk
τ^*, pk
Oracle Phase I
τ, c
$m := \mathcal{O}_{\text{stag-CCA}}(\tau, c)$
m
m_0, m_1
m_0, m_1
$b \xleftarrow{\$} \{0,1\}$
$c^* := \texttt{enc}(pk, \tau^*, m_b)$
c^*
c^*
Oracle Phase II (exactly the same as Oracle Phase I)
$b \overset{?}{=} b^*$
b^*
b^*

Figure 23: Reduction from IND-gtag-CCA to IND-stag-CCA

The adversary $\mathcal{A}_{\text{stag-CCA}}$ firstly draws a random challenge tag τ^* (just like the IND-gtag-CCA challenger would do) and sends it to the challenger $\mathcal{C}_{\text{stag-CCA}}$. The challenge tag is also given to $\mathcal{A}_{\text{gtag-CCA}}$ together with the challenge key pk from $\mathcal{C}_{\text{stag-CCA}}$. Afterwards $\mathcal{A}_{\text{stag-CCA}}$ just forwards all messages between challenger, oracle and $\mathcal{A}_{\text{gtag-CCA}}$. If the adversary $\mathcal{A}_{\text{gtag-CCA}}$ wins, so does $\mathcal{A}_{\text{stag-CCA}}$. □

On the other hand we find that an implication from IND-gtag-CCA to IND-stag-CCA is not true in general. For polynomially sized tag spaces, i.e. when $\frac{1}{|\mathbf{T}|}$ is non-negligible, however, we find that IND-gtag-CCA and IND-stag-CCA are equivalent.

Lemma 8 (IND-gtag-CCA $\not\Rightarrow$ IND-stag-CCA):
There is an IND-gtag-CCA secure tag-based encryption scheme which does not satisfy IND-stag-CCA security.

Proof:
Let $(\texttt{gen}, \texttt{enc}, \texttt{dec})$ be an IND-stag-CCA secure tag-based encryption scheme with tag space $\mathbf{T} := \{0,1\}^\lambda$. Now consider the punctured scheme

$$\begin{aligned}
\texttt{Gen} &\equiv \texttt{gen} \\
\texttt{Enc}(pk, \tau, m) &:= \begin{cases} m, & \tau = 0^\lambda \\ \texttt{enc}(pk, \tau, m), & \text{else} \end{cases} \\
\texttt{Dec}(sk, \tau, c) &:= \begin{cases} c, & \tau = 0^\lambda \\ \texttt{dec}(sk, \tau, c), & \text{else.} \end{cases}
\end{aligned}$$

By assumption $(\texttt{gen}, \texttt{enc}, \texttt{dec})$ is IND-stag-CCA and by Lemma 7 in particular IND-gtag-CCA secure. Furthermore the probability

$$\mathbb{P}[\tau^* = 0^\lambda] = \frac{1}{2^\lambda}$$

is negligible, hence the resulting scheme $(\texttt{Gen}, \texttt{Enc}, \texttt{Dec})$ is still IND-gtag-CCA secure. It does not, however, satisfy IND-stag-CCA security anymore, as the adversary $\mathcal{A}_{\text{stag-CCA}}$ has the power to choose $\tau^* = 0^\lambda$ in this game and win with probability one. □

Lemma 9 (IND-gtag-CCA $\Rightarrow$ IND-stag-CCA):
Any IND-gtag-CCA secure tag-based encryption scheme with a tag space $\mathbf{T}$ *such that* $\frac{1}{|\mathbf{T}|}$ *is non-negligible automatically satisfies IND-stag-CCA security.*

Proof:
Let $\mathcal{A}_{\text{stag-CCA}}$ be an adversary with non-negligible success probability to win the IND-stag-CCA game. We construct an adversary $\mathcal{A}_{\text{gtag-CCA}}$ as follows: In the first step $\mathcal{A}_{\text{gtag-CCA}}$ receives challenge tags $\tau^*_{\mathcal{C}}$ and $\tau^*_{\mathcal{A}}$ from $\mathcal{C}_{\text{gtag-CCA}}$ and $\mathcal{A}_{\text{stag-CCA}}$ respectively. If $\tau^*_{\mathcal{C}} \neq \tau^*_{\mathcal{A}}$ the adversary $\mathcal{A}_{\text{gtag-CCA}}$ aborts. Otherwise it continues to just forward messages between challenger, oracle and $\mathcal{A}_{\text{gtag-CCA}}$. Hence $\mathcal{A}_{\text{gtag-CCA}}$ has success probability

$$\frac{1}{|\mathbf{T}|} \cdot \mathbb{P}[\mathcal{A}_{\text{stag-CCA}} \text{ wins}],$$

which is non-negligible. □

IND-gtag-CCA Implies IND-SB-CPA

In this section we concentrate on the relationship between IND-SB-CPA and IND-gtag-CCA. This lets us enqueue IND-SB-CPA security into the linear hierarchy of tag-based encryption notions. To compare the two notions we assume the tag space $\mathbf{T}$ considered for IND-gtag-CCA to be equal to the set $\mathbf{P}$ of party IDs. Of course a bijection between the two is sufficient as well, but we compare the notions for tag and ID spaces of the same size. An overview of the implications between tag-based encryption notions and IND-SB-CPA is shown in Figure 24.

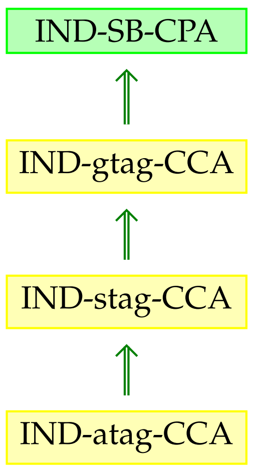


Figure 24: Relationship Between IND-SB-CPA and Tag-based Encryption Notions

Lemma 10 (IND-gtag-CCA $\Rightarrow$ IND-SB-CPA):
Any IND-gtag-CCA secure tag-based encryption scheme automatically satisfies IND-SB-CPA security.

Proof:
Let $(\texttt{gen}, \texttt{enc}, \texttt{dec})$ be a tag-based encryption scheme. Under assumption of an efficient adversary $\mathcal{A}_{\text{SB-CPA}}$ with non-negligible probability to win the IND-SB-CPA security game, we will construct an efficient adversary $\mathcal{A}_{\text{gtag-CCA}}$ who has the same success probability in the IND-gtag-CCA game. An overview of the construction can be found in Figure 25.

After being handed an ID S as the challenge tag and a public key pk, the adversary $\mathcal{A}_{\text{gtag-CCA}}$ determines an ID R matching the public key $pk = pk_R$ and generates a key pair (sk_S, pk_S) matching the ID S. Depending on the specific scheme, these might, e.g. involve some key registration or be completely independent of one another. The IDs and public keys (S, pk_S, R, pk_R) are handed on to $\mathcal{A}_{\text{SB-CPA}}$. Any valid oracle queries $(pk_{R'}, S', c)$ from $\mathcal{A}_{\text{SB-CPA}}$ (i.e. those with $S' \notin \{S, R\}$ and $pk_{R'} \in \{pk_S, pk_R\}$) are answered in one of two ways: If pk'_R is equal to the challenge key pk_R, the query (S', c) is forwarded to $\mathcal{A}_{\text{gtag-CCA}}$'s own oracle $\mathcal{O}_{\text{gtag-CCA}}$. Otherwise, $\mathcal{A}_{\text{gtag-CCA}}$ uses the secret key sk_S to perform the decryption itself. In both cases the challenge is answered exactly like oracle $\mathcal{O}_{\text{SB-CPA}}$ would. After forwarding the messages m_0, m_1 and the challenge ciphertext c^* between $\mathcal{A}_{\text{SB-CPA}}$ and $\mathcal{C}_{\text{gtag-CCA}}$, the oracle phase is repeated exactly as before. Finally, the bit b^* which $\mathcal{A}_{\text{SB-CPA}}$ outputs is forwarded as well. If the adversary $\mathcal{A}_{\text{SB-CPA}}$ wins, so does $\mathcal{A}_{\text{gtag-CCA}}$. $\square$

Lemma 11 (IND-SB-CPA $\not\Rightarrow$ IND-gtag-CCA):
There is an IND-SB-CPA secure sender-binding encryption scheme which does not satisfy IND-gtag-CCA security.

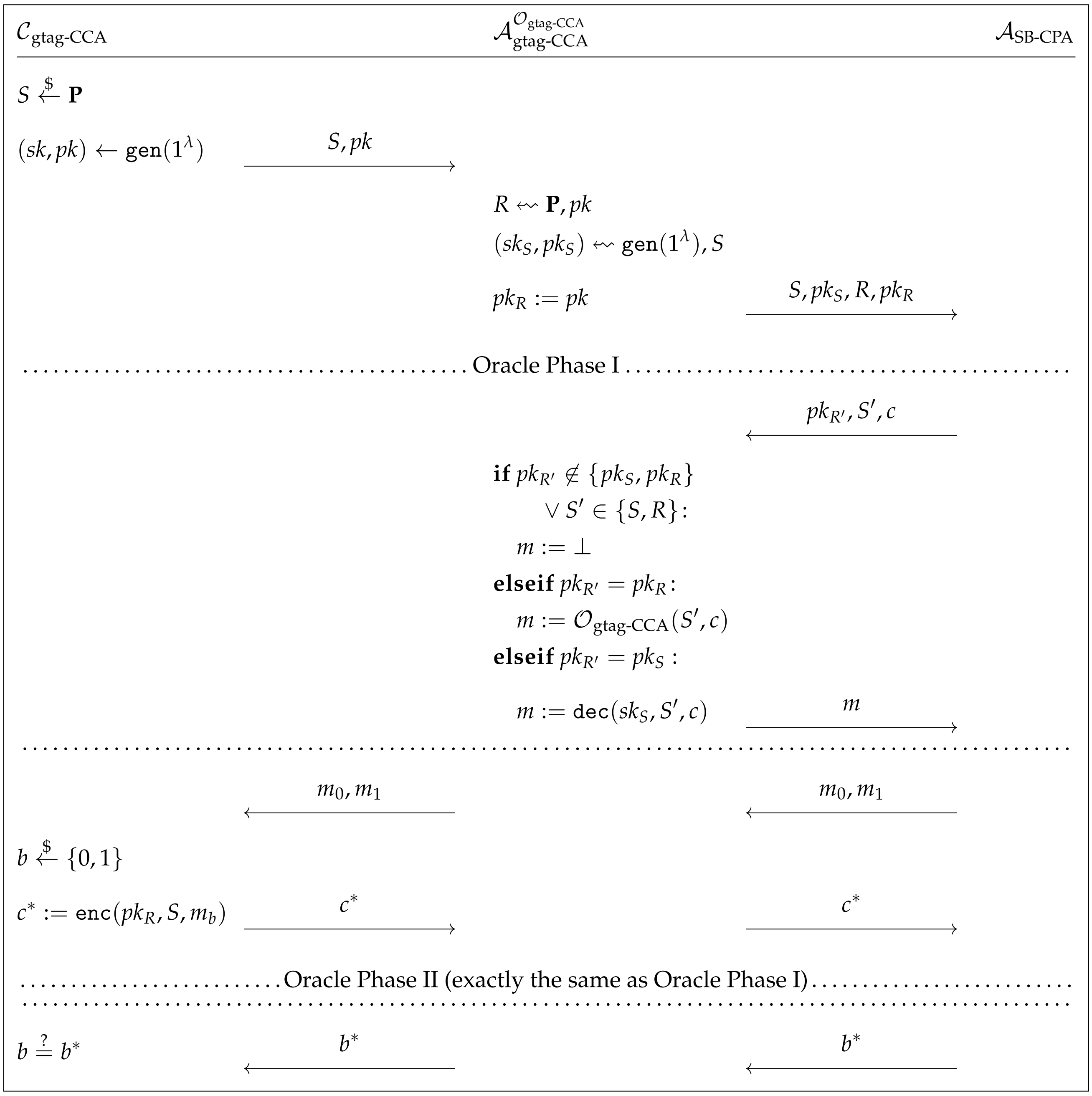

$\mathcal{C}_{\text{gtag-CCA}}$
$\mathcal{A}_{\text{gtag-CCA}}^{\mathcal{O}_{\text{gtag-CCA}}}$
$\mathcal{A}_{\text{SB-CPA}}$
$S \xleftarrow{\$} \mathbf{P}$
$(sk, pk) \leftarrow \texttt{gen}(1^\lambda)$
S, pk
$R \leftsquigarrow \mathbf{P}, pk$
$(sk_S, pk_S) \leftsquigarrow \texttt{gen}(1^\lambda), S$
$pk_R := pk$
S, pk_S, R, pk_R
Oracle Phase I
$pk_{R'}, S', c$
if $pk_{R'} \notin \{pk_S, pk_R\}$
$\vee S' \in \{S, R\}$:
$m := \perp$
elseif $pk_{R'} = pk_R$:
$m := \mathcal{O}_{\text{gtag-CCA}}(S', c)$
elseif $pk_{R'} = pk_S$:
$m := \texttt{dec}(sk_S, S', c)$
m
m_0, m_1
m_0, m_1
$b \xleftarrow{\$} \{0, 1\}$
$c^* := \texttt{enc}(pk_R, S, m_b)$
c^*
c^*
Oracle Phase II (exactly the same as Oracle Phase I)
$b \stackrel{?}{=} b^*$
b^*
b^*

Figure 25: Reduction from IND-SB-CPA for SBE to IND-gtag-CCA

Proof:
Let us consider the dual receiver encryption-based example $(\texttt{Gen}, \texttt{Enc}, \texttt{Dec})$ from section 5.1.2 again. In Lemma 4 we have already shown that this scheme is IND-SB-CPA secure. To prove our current claim it remains to be shown that $(\texttt{Gen}, \texttt{Enc}, \texttt{Dec})$ does not satisfy IND-gtag-CCA security. We do so by constructing an efficient adversary $\mathcal{A}_{\text{gtag-CCA}}$ which has non-negligible probability of winning the IND-gtag-CCA security game. Firstly the challenger $\mathcal{C}_{\text{gtag-CCA}}$ chooses a random party ID $S \in \mathbf{P}$, generates the challenge key pair (SK_R, PK_R) and registers it for some party R. On input of S, PK_R, the adversary $\mathcal{A}_{\text{gtag-CCA}}$ generates a fresh key pair (SK_S, PK_S), and registers this key pair with $\mathcal{F}_{\text{KRK}}$ in the name of S. Now the adversary chooses random messages $m_0 \neq m_1$

for the challenge and receives $c^* = \texttt{Enc}(PK_R, S, m_b)$. Due to dual receiver encryption soundness the adversary can now decrypt the challenge as

$$m_b = \texttt{Dec}(SK_S, R, c^*)$$

and win the IND-gtag-CCA game with probability one. □

Although this proof is instructing for the intuitive understanding of sender-binding encryption schemes, it relies on the fact that there is a connection between tags and party keys as well as on the party whose ID is randomly chosen as the challenge tag being corruptible by the adversary. I.e. the adversary needs to be able to register keys for this party, which might not always be realistic. Due to this caveat let me give a second proof of the lemma:

Proof (Alternative version):
Let $(\texttt{gen}, \texttt{enc}, \texttt{dec})$ be an IND-SB-CPA secure sender-binding encryption scheme. We use this to construct a sender-binding encryption/tag-based encryption scheme $(\texttt{Gen}, \texttt{Enc}, \texttt{Dec})$ which is still IND-SB-CPA secure but does not satisfy IND-gtag-CCA security:

$$\begin{aligned}\texttt{Gen} &\equiv \texttt{gen}\\ \texttt{Enc} &\equiv \texttt{enc}\\ \texttt{Dec}(sk, S, c) &:= \begin{cases}\texttt{dec}(sk, S, c)||sk, & sk = sk_S\\ \texttt{dec}(sk, S, c)||0\cdots 0, & \text{else.}\end{cases}\end{aligned}$$

It is obvious that this modified scheme does still satisfy IND-SB-CPA security, as we have $(\texttt{Gen}, \texttt{Enc}) \equiv (\texttt{gen}, \texttt{enc})$ everywhere and $\texttt{Dec} = \texttt{dec}$ on the domain where $\mathcal{O}_{\text{SB-CPA}}$ answers queries. It is not, however, IND-gtag-CCA secure, as any adversary can query the oracle $\mathcal{O}_{\text{gtag-CCA}}$ with input (R, c), where R is the party ID corresponding to challenge key pk_R and c is an arbitrary ciphertext. The oracle will hand back sk_R which can be used to decrypt the challenge ciphertext c^* and win the security game every time. □

6.1.2 Classic Public-key Notions Skew to IND-SB-CPA

In this section we look at the implications between IND-SB-CPA and indistinguishability notions for classic public-key encryption ranging from IND-CPA to IND-CCA2. This highlights a certain skewness between the new IND-SB-CPA and classical public-key encryption security notions which we try to understand a bit better in Section 6.3.

The standard games from IND-CPA to IND-CCA2 are commonly defined for public-key encryption schemes where encryption depends only on the keys of the decrypting/receiving party. Hence they only deal with one pair of keys which is used to encrypt and decrypt the challenge. For sender-binding encryption, another (encrypting/sending) party and their credentials need to be fixed. This gives us a degree of freedom of when as well as by whom this is chosen. In principle, there are (at least) four different options: The sender and its credentials are chosen…

(1) …randomly by the challenger, right before encrypting the challenge.

(2) …randomly by the challenger at the start of the game.

(3) …by the adversary at the start of the game.

(4) . . . by the adversary, together with the challenge messages.

We already know this degree of freedom from identity-based encryption schemes, where different notions exist according to when the receiver is chosen. The four above possibilities give rise to the obvious implications

$$(4) \Rightarrow (3) \Rightarrow (2) \Rightarrow (1)$$

for any notion. Note that by "choosing" we mean that all public credentials are handed from the choosing to the non-choosing party. Of course the values may actually be fixed earlier if the choosing party decides to do so. To intuitively compare other games to IND-SB-CPA we will stay within the same mode of choosing the sender which our definition of IND-SB-CPA in Section 3.2 presents. I.e. that the challenger randomly chooses the sender together with the receiver at the start of the game. This corresponds to item (2) in the list above.

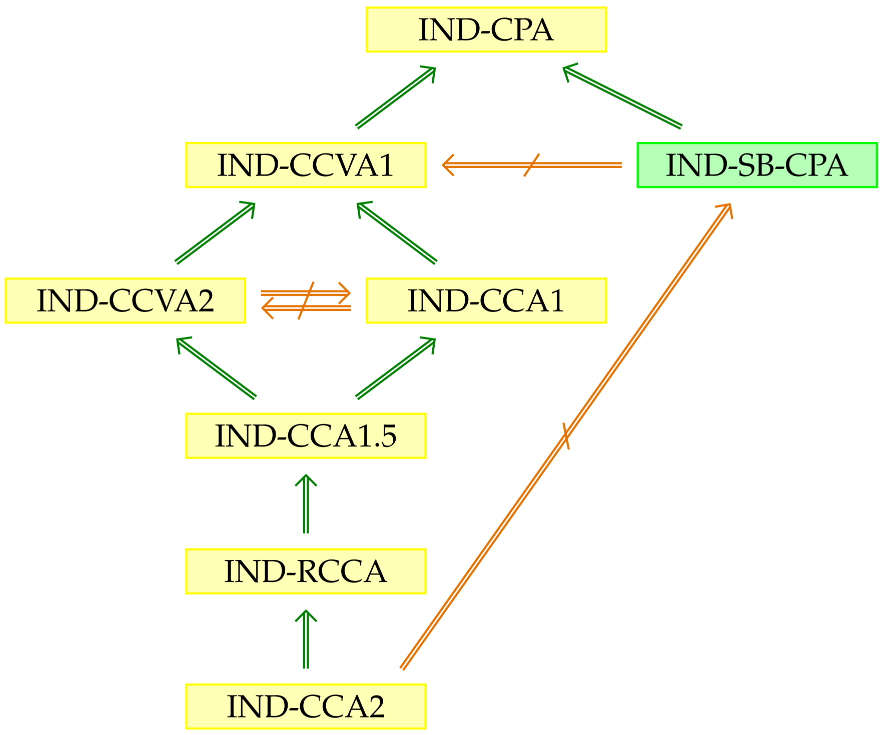


Figure 26: Relationship Between IND-SB-CPA and Classic Public-key Encryption Notions

Figure 26 gives an overview of the implications between IND-SB-CPA and classic game-based public-key encryption security notions. We will now examine these (non-)implications in more detail. Note that all (non-)implications work equally well in most of the other sender choosing modes listed above, as long as IND-SB-CPA is also adapted to this mode.

Lemma 12 (IND-SB-CPA $\Rightarrow$ IND-CPA):
Any IND-SB-CPA secure sender-binding encryption scheme automatically satisfies IND-CPA security.

Since it is obvious that IND-CPA is just IND-SB-CPA without any access to oracles, the adversary is strictly less powerful in this case and the implication rather trivial. We will therefore refrain from a formal proof at this point and move on to more interesting cases:

Lemma 13 (IND-SB-CPA $\not\Rightarrow$ IND-CCVA1):
There is an IND-SB-CPA secure sender-binding encryption scheme which does not satisfy IND-CCVA1 security.

In this case we conduct a proof by example: We construct a sender-binding encryption scheme that satisfies IND-SB-CPA security but which is not IND-CCVA1 secure. Consider the identity-based encryption example $(\texttt{Gen}, \texttt{Enc}, \texttt{Dec})$ we saw as a generic way to gain IND-SB-CPA secure sender-binding encryption in Section 5.1.3:

$$\begin{aligned}
&\texttt{Gen} \equiv \texttt{gen} \text{ (i.e. } (SK, PK) := (msk, mpk))\\
&\texttt{Enc:}\quad (PK_R, S, m) = (mpk_R, S, m) \mapsto \texttt{enc}\Big(mpk_R, \texttt{f}_{\text{ID}}(S), m\Big)\\
&\texttt{Dec:}\qquad (SK_R, S, c) = (msk_R, S, c) \mapsto \texttt{dec}\Big(\texttt{ext}(msk_R, \texttt{f}_{\text{ID}}(S)), c\Big).
\end{aligned}$$

We now modify its encryption and decryption algorithm such that the scheme $(\texttt{Gen}, \texttt{Enc}^*, \texttt{Dec}^*)$ still satisfies IND-SB-CPA security but is not IND-CCVA1 secure. The intuition behind this construction is the following: Each ciphertext is concatenated with a string of bits. For the generic use of honestly encrypting and decrypting messages this string is just zeros and completely uninteresting. The decryption algorithm is defined in such a way, however, that it still succeeds when part of this string of zeros is replaced by an (equally long) prefix of the secret key used for decryption. Given a verification oracle it is now fairly easy to extract this secret key by testing if a ciphertext is still valid when the string of zeros is modified bit by bit. Since the secret key used for decryption is specific to the receiver *and* the sender of the message in this particular identity-based encryption construction, the IND-SB-CPA oracles can not be used to extract any secrets, as they do not pertain to communication between the sender and receiver used for the challenge ciphertext.

More formally let k be the length of user secret keys usk when they are binary encoded and let $pr : \{0,1\}^k \times \{0,1\}^k \to \{\mathsf{true}, \mathsf{false}\}$ be a prefix checking function defined by:

$$\begin{aligned}
&pr\Big((x_1 \cdots x_k), (y_1 \cdots y_k)\Big) = \mathsf{true}\\
&:\Leftrightarrow \exists k_0 \in \{0, \dots, k\} \colon y_i = \begin{cases} x_i, & i \leqslant k_0\\ 0, & \text{otherwise.}\end{cases}
\end{aligned}$$

With this function we can now (remembering $\texttt{Gen} = \texttt{gen}$, i.e. $(SK, PK) = (msk, mpk)$) define

$$\begin{aligned}
&\texttt{Enc}^*\colon (PK_R, S, m) = (mpk_R, S, m) \mapsto \Big(\texttt{enc}(mpk_R, \texttt{f}_{\text{ID}}(S), m) \| 0^k\Big)\\
&\texttt{Dec}^*\colon (SK_R, S, C) = \Big(msk_R, S, (c\|x)\Big) \mapsto \begin{cases} \texttt{dec}(usk_S, c), & pr(usk_S, x)\\ \bot, & \text{otherwise}\end{cases}
\end{aligned}$$

where $usk_S := \texttt{ext}(msk_R, \texttt{f}_{\text{ID}}(S))$ is defined as before and $x = (x_1 \cdots x_k) \in \{0,1\}^k$.

Lemma 14:
If the identity-based encryption scheme $(\texttt{gen}, \texttt{ext}, \texttt{enc}, \texttt{dec})$ *is IND-sID-CPA secure, then the sender-binding encryption scheme* $(\texttt{Gen}, \texttt{Enc}^*, \texttt{Dec}^*)$ *is IND-SB-CPA secure.*

Proofsketch:
The reduction to the IND-sID-CPA security of the underlying identity-based encryption scheme is largely identical to the proof in Section 5.1.3. The only difference is that for any oracle query $(mpk_{R'}, S', C)$ with $C = (c \| x)$, the adversary $\mathcal{A}_{\text{sID-CPA}}$ only decrypts c if $pr(usk_{S'}, x)$ is true. Otherwise it hands back $\perp$. $\triangledown$

Lemma 15:
If the identity-based encryption scheme $(\texttt{gen}, \texttt{ext}, \texttt{enc}, \texttt{dec})$ *is only IND-sID-CPA secure, then the sender-binding encryption scheme* $(\texttt{Gen}, \texttt{Enc}^*, \texttt{Dec}^*)$ *is not IND-CCVA1 secure.*

Proof:
Let $\mathcal{A}_{\text{CCVA1}}$ be an adversary which queries the oracle in the following manner: Set

$$c \leftarrow \texttt{Enc}(PK_R, S, m) = \texttt{enc}\Big(mpk_R, \texttt{f}_{\text{ID}}(S), m\Big)$$

for some arbitrary message m and define queries

$$\forall i \in \{1, \ldots, k\} \colon Q_i := (mpk_R, S, c \| x_1 \cdots x_{i-1} \| 1 \| 0^{k-i})$$

where $x_j := 1$ if the oracles response to query Q_j was true and $x_j := 0$ otherwise for all $j \in \{1, \ldots, i\}$. The user secret key usk_S is thus obtained as

$$usk_S = x_1 \cdots x_k$$

from the oracle's responses and can be used to successfully decrypt any challenge ciphertext c^*. $\square$

The proof of Lemma 13 follows directly from Lemmas 14 and 15. Note that by transitivity of implication this yields in particular, that IND-SB-CPA does not imply any of the stronger security notions either.

Lemma 16 (IND-CCA2 $\not\Rightarrow$ IND-SB-CPA):
There is an IND-CCA2 secure public-key encryption scheme which does not satisfy IND-SB-CPA security.

Proof:
This again is rather easy to see by example: Let $(\texttt{gen}, \texttt{enc}, \texttt{dec})$ be any standard IND-CCA2 secure encryption scheme, where ciphertexts are independent of the sender S. The adversary $\mathcal{A}_{\text{SB-CPA}}$ can query the oracle $\mathcal{O}_{\text{SB-CPA}}$ in the second oracle phase with input (pk_R, P, c^*) for any $P \neq S$ and obtain the content of the challenge ciphertext c^*. $\square$

Note, again, that by transitivity of implication this means IND-SB-CPA is not implied by any of the weaker classic public-key encryption security notions either.

6.2 Theoretic Classification of IND-SB-CPA for Sender-binding Key Encapsulation

We now change the focus from the sender-binding encryption to the sender-binding key encapsulation notion of IND-SB-CPA security. Before we discuss how IND-SB-CPA fits into the landscape of other key encapsulation security notions, note that an IND-SB-CPA secure sender-binding encryption obviously yields an IND-SB-CPA secure sender-binding key encapsulation mechanism by the standard public-key encryption to key encapsulation construction of randomly drawing and then encrypting a symmetric key.

For key encapsulation security notions, classifying IND-SB-CPA for sender-binding key encapsulation with respect to classic key encapsulation security is unfortunately rather infeasible. While a classic key encapsulation mechanism takes no input and requires secrecy and various forms of integrity about the internally determined key, IND-SB-CPA security for sender-binding key encapsulation asserts only secrecy (no integrity) of the key, but additionally provides integrity (without secrecy) of some additional user input—the identity S. Since those two settings are even more incompatible than comparing sender-binding encryption to classic public-key encryption notions, I will only consider IND-SB-CPA for sender-binding key encapsulation in relation to the similar setting of tag-based key encapsulation mechanisms.

6.2.1 Relation to tag-KEM Security Notions

Let $(\texttt{gen}, \texttt{key}, \texttt{enc}, \texttt{dec})$ be a tag-based key encapsulation mechanism. From this we construct a sender-binding key encapsulation mechanism $(\texttt{Gen}, \texttt{Enc}, \texttt{Dec})$ in the natural way: Using sender IDs as tags, and combining `key` and `enc` into a single encryption algorithm. I.e.

```
$\texttt{Gen} \equiv \texttt{gen}$
$\texttt{Enc}(pk_R, S)$ :
  • $(aux, K) \leftarrow \texttt{key}(pk_R)$
  • $C \leftarrow \texttt{enc}(aux, S)$
  $\hookrightarrow$ Return $(K, C)$
$\texttt{Dec} \equiv \texttt{dec}$.
```

It could be argued that this change to the underlying tag-based key encapsulation mechanism means we are dealing with a generic transformation instead of directly comparing security notions. This is an absolutely valid point of view, and we could have discussed the following details in Chapter 5. However, I think that the tag-based key encapsulation division into key generation and encapsulation is very specific to the needs of [AGKS05] where the tag depends on the generated key. It does not reflect in other tag-based notions like tag-based encryption and becomes rather meaningless when tags are chosen to be sender IDs, which is why I have chosen to present the relations in this chapter, instead.

To the best of my knowledge, no weaker security notion than IND-CCA2 has been proposed for tag-based key encapsulation so far. Hence we start the comparison there:

Lemma 17 (IND-CCA2 $\Rightarrow$ IND-SB-CPA):
If $(\texttt{gen}, \texttt{key}, \texttt{enc}, \texttt{dec})$ *is an IND-CCA2 secure tag-based key encapsulation mechanism then* $(\texttt{Gen}, \texttt{Enc}, \texttt{Dec})$ *is an IND-SB-CPA secure sender-binding key encapsulation mechanism.*

Proof:
Assume on the contrary that $\mathcal{A}_{\text{SB-CPA}}$ is an adversary with non-negligible success probability in winning the sender-binding key encapsulation version of the IND-SB-CPA game. We use $\mathcal{A}_{\text{SB-CPA}}$ to construct an equally successful adversary $\mathcal{A}_{\text{CCA2}}$ playing the tag-based key encapsulation mechanism version of the IND-CCA2 security game. This adversary mainly forwards

messages between $\mathcal{A}_{\text{SB-CPA}}$ and the challenger and oracle. Additionally it creates credentials for S and uses them to decrypt respective oracle queries. The detailed reduction can be found in Figure 27. □

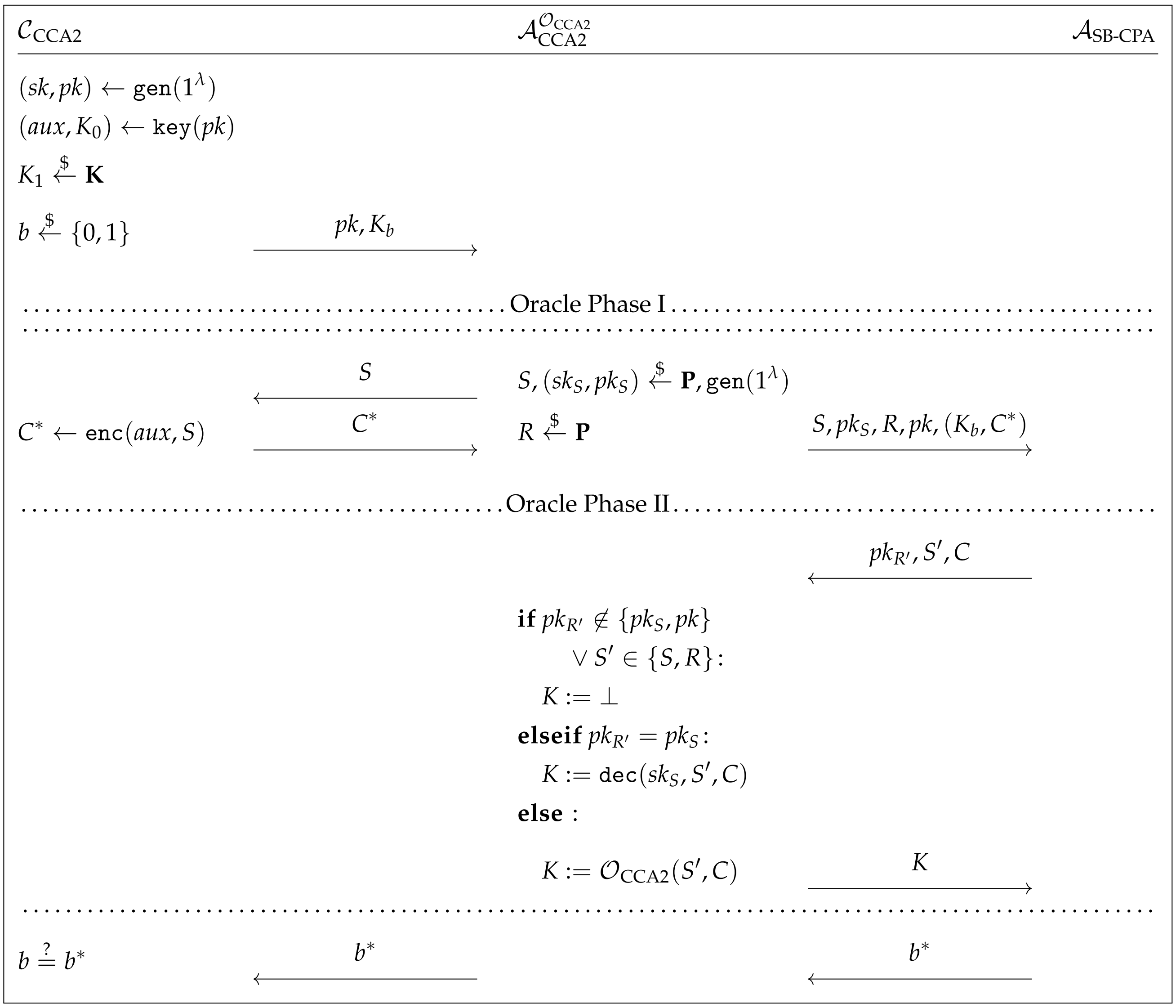

$\mathcal{C}_{\text{CCA2}}$
$\mathcal{A}_{\text{CCA2}}^{\mathcal{O}_{\text{CCA2}}}$
$\mathcal{A}_{\text{SB-CPA}}$
$(sk, pk) \leftarrow \texttt{gen}(1^\lambda)$
$(aux, K_0) \leftarrow \texttt{key}(pk)$
$K_1 \xleftarrow{\$} \mathbf{K}$
$b \xleftarrow{\$} \{0,1\}$
pk, K_b
Oracle Phase I
S
$S, (sk_S, pk_S) \xleftarrow{\$} \mathbf{P}, \texttt{gen}(1^\lambda)$
$C^* \leftarrow \texttt{enc}(aux, S)$
C^*
$R \xleftarrow{\$} \mathbf{P}$
$S, pk_S, R, pk, (K_b, C^*)$
Oracle Phase II
$pk_{R'}, S', C$
if $pk_{R'} \notin \{pk_S, pk\}$
$\vee\, S' \in \{S, R\}$:
$K := \perp$
elseif $pk_{R'} = pk_S$:
$K := \texttt{dec}(sk_S, S', C)$
else :
$K := \mathcal{O}_{\text{CCA2}}(S', C)$
K
$b \overset{?}{=} b^*$
b^*
b^*

Figure 27: Reduction from IND-SB-CPA for SB-KEMs to IND-CCA2 for tag-KEMs

It is easy to see from the reduction that the IND-CCA2 game grants a lot more oracle access than we need for the proof. This indicates that IND-SB-CPA security is actually a lot weaker than IND-CCA2 in the sender-binding key encapsulation setting. To further substantiate this claim I take the weakest known security notion for tag-based encryption, adapt it to the key encapsulation setting and show that it still implies the sender-binding key encapsulation version of IND-SB-CPA security via the above construction.

The tag-based encryption notion in question is IND-gtag-CCA, which I introduced in Section 3.2. The difference to IND-CCA2 lies in the oracle access as well as when and by whom the challenge tag τ^* is chosen: Both oracle phases grant access to a decryption oracle punctuated at $\tau = \tau^*$, i.e. the complete challenge tag is excluded from decryption rather than just the challenge tuple (τ^*, C^*). The challenge tag itself is not chosen adaptively and not even by the adversary at all anymore, but randomly drawn by the challenger. The adaptive interface of a tag-based key encapsulation mechanism—where encapsulation is divided into key and enc so that the tag may depend on the output of key—does not seem quite fitting anymore when in the security game the tag is drawn at random and hence independent of the output of key. Nevertheless we cannot rule out that such a security notion may still be meaningful for a tag-based key encapsulation mechanism with separate key and enc and therefore keep the division.

Definition (IND-gtag-CCA for tag-KEMs):
A tag-based key encapsulation mechanism $\Sigma = (\texttt{gen}, \texttt{key}, \texttt{enc}, \texttt{dec})$ with key space $\mathbf{K}$ satisfies *indistinguishability under given-tag weakly chosen ciphertext attack (IND-gtag-CCA)*, if and only if for any PPT adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ the advantage

$$\begin{aligned}\mathsf{Adv}^{\mathrm{gtag}}_{\mathcal{A},\Sigma}(\lambda) := \Big|\mathbb{P}\Big[b \leftarrow \mathcal{A}_2^{\mathcal{O}^*}(K_b, C^*, aux_{\mathcal{A}}) \;\Big|\; &\tau^* \xleftarrow{\$} \mathbf{T}; (sk, pk) \leftarrow \texttt{gen}(1^\lambda);\\ &(aux_{\mathcal{A}}) \leftarrow \mathcal{A}_1^{\mathcal{O}^*}(pk, \tau^*); (aux, K_0) \leftarrow \texttt{key}(pk);\\ &C^* \leftarrow \texttt{enc}(aux, \tau^*); K_1 \xleftarrow{\$} \mathbf{K}; b \xleftarrow{\$} \{0,1\}\Big] - \tfrac{1}{2}\Big|\end{aligned}$$

is negligible in the security parameter λ, where $\mathcal{O}^*$ returns $\perp$ for $\tau = \tau^*$ and $\texttt{dec}(sk, \tau, C)$ otherwise. ○

Let us go on to see that this weaker notion is still sufficient to imply IND-SB-CPA security for sender-binding key encapsulation mechanisms.

Lemma 18 (IND-gtag-CCA $\Rightarrow$ IND-SB-CPA):
If $(\texttt{gen}, \texttt{key}, \texttt{enc}, \texttt{dec})$ *is an IND-gtag-CCA secure tag-based key encapsulation mechanism then* $(\texttt{Gen}, \texttt{Enc}, \texttt{Dec})$ *is an IND-SB-CPA secure sender-binding key encapsulation mechanism.*

Proof:
The proof of Lemma 18 works almost exactly the same as the proof of Lemma 17. The sole difference is that the identity S is randomly provided by the challenger $\mathcal{C}_{\text{gtag-CCA}}$ rather than randomly drawn by $\mathcal{A}_{\text{CCA2}}$ as before. Note that the provision $S' \notin \{S, R\}$ guarantees that oracle queries forwarded to $\mathcal{O}_{\text{gtag-CCA}}$ get decrypted correctly. □

Analogous to Section 6.1.1 we can use the generic dual receiver-based construction from Section 5.2.1 to separate the two security notions and show that IND-SB-CPA security does not imply IND-gtag-CCA for key encapsulation mechanisms either.

Lemma 19 (IND-SB-CPA $\not\Rightarrow$ IND-gtag-CCA):
There is an IND-SB-CPA secure sender-binding key encapsulation mechanism scheme which does not satisfy IND-gtag-CCA security.

Proof:
Consider the dual receiver-based sender-binding key encapsulation mechanism scheme from Section 5.2.1, which by Lemma 6 is IND-SB-CPA secure. We prove that this particular scheme does not satisfy IND-gtag-CCA security by constructing an adversary $\mathcal{A}_{\text{gtag-CCA}}$ which has non-negligible probability of winning the IND-gtag-CCA game. The challenger $\mathcal{C}_{\text{gtag-CCA}}$ provides $\mathcal{A}_{\text{gtag-CCA}}$ with input (pk_R, S) where public key pk_R has been registered with $\mathcal{F}_{\text{KRK}}$ for some party R. $\mathcal{A}_{\text{gtag-CCA}}$ goes on to generate keys (sk_S, pk_S) for party S and registers them with $\mathcal{F}_{\text{KRK}}$ as well before the challenge is created. Now when the challenger hands challenge (K_b, C^*) to $\mathcal{A}_{\text{gtag-CCA}}$ it can use sk_S to decrypt the challenge ciphertext and win the security game. □

6.3 Attempting a More Holistic Classification

As seen in Section 6.1.2, there seems to be a certain skewness between the new IND-SB-CPA and classical public-key encryption security notions which we would like to understand a bit better. In this section I discuss the possibility of a more holistic classification of game-based security notions for encryption schemes. I will then go on to indicate how IND-SB-CPA fits into this classification and that it is rather related to symmetric IND-CPA security. As throughout the rest of this dissertation I will keep calling an encrypting party "sender" and a decrypting party "receiver" for this section. I choose these terms not only because secure communication is the obvious use case of encryption but borrowing terms from this setting also enhances readability of the explanations.

We consider game-based notions to generally be of the form

$$\mathsf{goal}\text{-}\mathsf{set}\text{-}\mathsf{scp}\text{-}\mathsf{pow},$$

where $\mathsf{goal} \in \{\text{IND, NM, RoR,}\dots\}$ denotes the adversary's goal and

$$\mathsf{pow} \in \{\text{CPA, CCVA1, RCCA, CCA2,}\dots\}$$

denotes the adversary's power (restrictions), which—together with scp—determine the provision of oracles. In addition to these classically present parameters, we need setting set and scope scp to provide more degrees of freedom to accurately capture the wide range of notions.

The parameter set captures the *setting* in which the game is conducted. This includes all parties (implying their public information) and other parameters which need to be fixed as well as who (challenger or adversary) chooses them and when they are chosen. The classic public-key notions like IND-CPA use

$$\mathsf{set} = (\mathcal{C}, \text{Start}, \text{Receiver})$$

indicating that the only fixed party is the receiver who is chosen at the start of the game by the challenger. Note that, again, we mean this to imply that all public knowledge about these parties (e.g. the receiver's public key) is handed to the non-choosing party (e.g. the adversary) at the indicated time (e.g. "Start").

The variable scp indicates which *scope* of communication the adversary's knowledge restriction (indicated by pow) pertains to. This will definitely include the concrete configuration (S, R) of the challenge ciphertext, but may also include related configurations, like ciphertexts encrypted for the sender rather than receiver or for the same receiver but encrypted by different senders. For scp we will exemplarily introduce the terms *symmetric*, *directed*, *omnidirected* and *undique* for now:

(1) *Directed* scope means the adversary's power is only restricted with respect to the challenge configuration $(S \rightsquigarrow R)$.

(2) *Symmetric* scope indicates restriction regarding the challenge configuration $(S \rightsquigarrow R)$ as well as the reverse configuration $(R \rightsquigarrow S)$. This is the scope used in IND-SB-CPA.

(3) *Omnidirected* scope includes all configurations going out from the chosen sender, i.e. $(S \rightsquigarrow P)$ for arbitrary parties P. While this is not commonly considered for encryption schemes, it is the default setting for signatures.

(4) *Undique* scope can be viewed as the reverse of omnidirection: It includes all configurations incoming to the chosen receiver, i.e. $(P \rightsquigarrow R)$ for arbitrary P. This scope is commonly used for public-key encryption schemes where encryption and decryption are independent of the sender.

Note that there are many other instantiations of scp which might result in useful security notions. Outside of the indicated scope, we assume the adversary to have perfect power/knowledge. Within the scope scp, the adversary's power is determined by the parameter pow. We do not differentiate whether the adversary naturally has the required power—e.g. because they choose and therefore know a secret key or because only public knowledge is necessary—or whether it is provided in the form of an oracle.

For the relationship between the different scopes it is rather obvious that any scheme which is goal-set-scp-pow secure for $\mathsf{scp} \in \{$symmetric, omnidirected, undique$\}$ also satisfies goal-set-directed-pow. But any implications between the three stronger scopes are not trivially given.

From this interpretation of game-based security notions the skewness between IND-SB-CPA and commonly considered public-key encryption games like IND-CCA2 becomes intuitively obvious: We compare a symmetrically scoped notion with undique ones. I.e. we can not expect any implications between the security notions if there are no implications between the scopes. We also see that within this model IND-SB-CPA security is equivalent to classical symmetric key IND-CPA security. They are both of the form

$$\mathsf{goal\text{-}set\text{-}scp\text{-}pow} = \text{IND-}(\mathcal{C}, \text{Start}, (S, R))\text{-symmetric-CPA}.$$

Analysing this further might be an interesting question for future research.

In this chapter we attempted to comprehensively classify IND-SB-CPA security with respect to other game-based security notions. While we were able to show that IND-SB-CPA provides a strictly weaker security with respect to all previously proposed tag-based security notions, comparing it to classic public-key encryption as well as key encapsulation notions proved more complicated. Nevertheless we thoroughly investigated the skewness between the sender-binding encryption version of IND-SB-CPA security and classic public-key encryption notions and tried to understand it further by attempting a holistic classification of all game-based encryption security notions.

It's like someone took a transcript of a couple arguing at IKEA and made random edits until it compiled without errors.

Randall Munroe, xkcd #1513

CHAPTER 7
A Note on Efficient Constructions

Although I fully endorse theoretic advancement for its own sake, it undoubtedly becomes even more valuable if it results in practical real world benefits as well. In this chapter I discuss where potential real world benefits lie with regards to the theoretic insights of this dissertation and how far we are on the way to realising them.

The main theoretic advancement of this dissertation are the two new IND-SB-CPA encryption security notions which are sufficiently strong to facilitate secure communication in conjunction with authentication, while simultaneously being weaker than the security notions which are currently employed for this purpose. The obvious potential for practical benefits here is efficiency: The requirement to only satisfy IND-SB-CPA instead of, e.g. IND-CCA2 security could mean there are significantly more efficient protocols possible. This would save resources like time, computational power and energy. Furthermore, easier to satisfy security notions can also lead to higher security. While this sounds counter intuitive at first, those security notions can be applicable in scenarios where scarce resources precluded the implementation of previously available solutions. This again boils down to the question of efficient protocols.

I will use the remainder of this chapter to give an overview of the current efforts to devise more efficient encryption protocols by aiming for IND-SB-CPA security instead of other security notions.

None of the following constructions are my own contribution. They were originally published in [BGMOS22; SBB+23b] and can be attributed to Wasilij Beskorovajnov, Roland Gröll and Sarai Eilebrecht.

7.1 Sender-binding Encryption

For sender-binding encryption, the concrete efficient protocols which have been proposed thus far all take the detour via the generic dual receiver encryption transformation I have explained in Section 5.1.2. This means instead of directly providing an IND-SB-CPA secure sender-binding encryption scheme, IND-CPA secure and sound dual receiver encryption schemes were developed which each result in efficient sender-binding encryption schemes by Section 5.1.2.

The three dual receiver encryption schemes in question are based on McEliece as well as LPN and LWE assumptions and were proven secure in the standard model. I will only give details on the most competitive of the three protocols, the plain McEliece-based version, which was presented in [BGMOS22]. Details on the other two constructions, a 2-repetition McEliece variant and one construction based on LWE, can be found in the full version [BGMOS21].

The plain McEliece-based construction of Beskorovajnov et al. can be construed as an augmentation of a low-noise and IND-CPA secure LPN-based tag-based encryption construction which Kiltz, Masny and Pietrzak presented in [KMP14]. It is based on McEliece as well as LPN assumptions. While the IND-CPA security translates from the tag-based encryption to the dual receiver encryption scheme without many problems, the required dual receiver soundness needed extra work. For this purpose Beskorovajnov et al. added a second encryption of the randomness to the ciphertext of their dual receiver encryption scheme. This means the randomness recovery mechanism of the scheme can be exploited to perform the consistency check necessary to guarantee soundness. Another change from the protocol in [KMP14] is an exchange of the trapdoor mechanism to McEliece over Goppa codes.

Although the resulting dual receiver encryption scheme from [BGMOS22] can not improve efficiency by orders of magnitude, the efficiency discussion Beskorovajnov et al. provide on their new dual receiver encryption scheme shows it to improve public key and ciphertext sizes by constant factors compared to previous solutions in the standard model. Already gaining this much gives a good indication that it could indeed be possible to greatly improve the efficiency of encryption schemes used for secure message transfer if more effort is made on protocols that aim for the weaker but sufficient security notion of IND-SB-CPA.

7.2 Sender-binding Key Encapsulation

In contrast to sender-binding encryption, where we needed to take a detour over dual receiver encryption, two efficient protocols have directly been proposed for sender-binding key encapsulation mechanism so far. They are based on the hardness of LWE and again McEliece combined with LPN, and were published in [SBB+23b] and its full version [SBB+23a], respectively. Since the second sender-binding key encapsulation protocol, based on McEliece and LPN, is an adaptation of the dual receiver encryption scheme I explained in Section 7.1, I will not give any further details here.

The LWE-based sender-binding key encapsulation presented in [SBB+23b] is a tweaked version of the key encapsulation mechanism proposed in [MP12; BIL21]. The hash of the scheme is exchanged for sender IDs and the goal of IND-SB-CPA security leads to a message authentication code (MAC) becoming superfluous. This simplification means some efficiency gain can be expected—although, again, probably not by orders of magnitude—compared to the key encapsulation mechanism from [BIL21], which is considered to be the previously most efficient solution in the standard model. By staying in the standard model and using presumed post-quantum secure primitives like LWE only, the resulting sender-binding key encapsulation mechanism also manages to provide post-quantum security.

Science got way easier when we realised you were allowed to do studies just to rule stuff out.

Randall Munroe, xkcd #2783

CHAPTER 8
Conclusion

Secure communication remains one of the most important cryptographic problem in our modern digital society. In this dissertation I have investigated the following research question:

> *What is the weakest encryption security definition in order to establish secure communication from existing authenticated channels?*

I have done so for the settings of classic public-key encryption as well as hybrid encryption via the KEM-DEM framework. For classic public-key encryption the research question has remained open for two decades while for hybrid encryption, unfortunately, it has not received much attention, yet—even though it is vital for most of our modern digital communication.

I have addressed the research question by developing the concepts of sender-binding encryption and sender-binding key encapsulation, each with a corresponding new security notion of indistinguishability under sender-binding chosen plaintext attack (IND-SB-CPA). This new IND-SB-CPA security notion manages to unify all prior approaches to construct secure communication from some form of encryption and ideal authenticated channels in the universal composability (UC) framework.

I have given formal proofs that IND-SB-CPA secure sender-binding encryption and sender-binding key encapsulation (in conjunction with a suitably secure data encapsulation mechanism) are strong enough to UC-realise secure communication in the form of secure message transfer and secure channels from ideal authenticated channels. The proofs hold under static corruption and for a polynomial set of potential communication parties.

Furthermore, and for the first time regarding this research question, I was able to obtain an optimality result as well: Any generalised public-key encryption scheme which in conjunction with authenticated channels provides secure message transfer in the UC framework via the de facto standard encrypt-then-authenticate principle, necessarily provides the functionality of a sender-binding encryption scheme which satisfies IND-SB-CPA security. This means that under the mentioned reasonable restrictions, IND-SB-CPA secure sender-binding encryption fully answers above research question for the public-key encryption setting. Deriving a similar optimality result for hybrid encryption remains an interesting topic for future research.

In addition to these realisations and the optimality result, I have thoroughly analysed the respective IND-SB-CPA security notions for sender-binding encryption and sender-binding key encapsulation mechanisms by providing generic constructions from other types of encryption schemes as well as theoretic classifications with respect to established game-based security notions.

I have furthermore argued the potential for real world impact of my theoretical work by discussing concrete efficient constructions which were already proposed for IND-SB-CPA secure sender-binding encryption as well as sender-binding key encapsulation. The in my opinion most worthwhile goal for future research would be to thoroughly investigate how much more efficiency can be gained via encryption and key encapsulation protocols which aim for IND-SB-CPA security instead of currently employed stronger security notions.

Book people hate seeing books sorted by color, but it turns out they get *way* more angry if you sort the pages by number.

Randall Munroe, xkcd #2791

APPENDIX A
Acronyms

DEM/data encapsulation data encapsulation mechanism

DR-KEM/dual receiver key encapsulation dual receiver key encapsulation mechanism

DRE dual receiver encryption

ETA encrypt-then-authenticate

IBE identity-based encryption

IND indistinguishability

IND-atag-CCA/atag-CCA indistinguishability under adaptive-tag weakly chosen ciphertext attack

IND-CCA2/CCA2 indistinguishability under adaptive chosen ciphertext attack

IND-CCVA1/CCVA1 indistinguishability under non-adaptive chosen ciphertext verification attack

IND-CPA/CPA indistinguishability under chosen plaintext attack

IND-gtag-CCA/gtag-CCA indistinguishability under given-tag weakly chosen ciphertext attack

IND-OT/OT indistinguishability under one-time attack

IND-RCCA/RCCA indistinguishability under replayable chosen ciphertext attack

IND-SB-CPA/SB-CPA indistinguishability under sender-binding chosen plaintext attack

IND-stag-CCA/stag-CCA indistinguishability under selective-tag weakly chosen ciphertext attack

IND-sID-CPA/sID-CPA indistinguishability under selective identity chosen plaintext attack

ITI interactive Turing machine instance

KEM/key encapsulation key encapsulation mechanism

KRK key registration with knowledge

LPN learning parity with noise

LWE learning with errors

MAC message authentication code

PKE public-key encryption

PPT probabilistic polynomial time

SB-KEM/sender-binding key encapsulation sender-binding key encapsulation mechanism

SBE sender-binding encryption

SMT secure message transfer

tag-KEM/tag-based key encapsulation tag-based key encapsulation mechanism

TBE tag-based encryption

UC universal composability

Honk iff you love formal logic.

Randall Munroe, xkcd #1033

APPENDIX B
Variables

$\xleftarrow{\$}$	left side uniformly randomly drawn from right side
$\geqslant_{\mathrm{UC}}$	left side UC-emulates right side
$\mathcal{A}$	adversary
a_n	element of a sequence
Adv	advantage
aux	auxiliary in- our output
c	ciphertext
$\mathcal{C}$	challenger
C	encapsulation of a key K
$\mathcal{D}$	dummy adversary
$\mathtt{dec}/\mathtt{Dec}$	decryption algorithm
$\mathtt{decap}$	decapsulation algorithm
ε	negligible function
$\mathtt{enc}/\mathtt{Enc}$	encryption algorithm
$\mathtt{encap}$	encapsulation algorithm
EXEC	execution of a system of ITIs
$\mathtt{ext}$	user key extraction algorithm
$\mathcal{F}$	ideal functionality
$\mathtt{f}_{\mathrm{act}}$	function indicating active sessions via true/false/init
$\mathtt{f}_{\mathrm{ID}}$	function of parties' identity-based encryption IDs

$\mathtt{f}_{\mathbf{K}}$	function storing known symmetric session keys
$\mathtt{f}_{\mathrm{Key}}$	function checking whether public and secret key belong together
$\mathtt{f}_{\mathbf{PK}}$	function storing known public keys
$\mathtt{f}_{\mathbf{SK}\to\mathbf{PK}}$	function generating public from secret key
$\mathtt{gen}/\mathtt{Gen}$	key generation algorithm
goal	adversary's goal
H	hybrid execution
i	natural number, possibly zero
id	identity
$\mathbf{ID}$	set of identities
j	natural number, possibly zero
k	natural number, possibly zero
K	symmetric key
$\mathbf{K}$	set of symmetric keys
$\mathtt{key}$	symmetric key generation algorithm
λ	security parameter
m	message
$\mathbf{M}$	set of messages
mid	message ID
$\mathbf{MID}$	set of message IDs
mpk	master public key
msk	master secret key
n	natural number, possibly zero
$\mathcal{O}$	oracle
π	protocol
p	polynomial
P	protocol party
$\mathbf{P}$	set of protocol parties

$\mathbb{P}$	probability
P^{π}	party P running protocol π
$\mathsf{p}_{\mathrm{msg}}$	function storing pending messages
$\mathsf{p}_{\mathrm{reg}}$	function storing pending registrations
$\mathsf{p}_{\mathrm{ret}}$	function storing pending retrieval requests
pid	party ID
pk/PK	public key
$\mathbf{PK}$	set of public keys
pow	adversary's power (restrictions)
pr	prefix checking function
q	polynomial
ρ	non-negligible function
R	protocol party "receiver"
$\mathbf{R}$	set of registered parties
Σ	concrete cryptographic scheme
S	protocol party "sender"
$\mathcal{S}$	UC simulator
sid	session ID
scp	scope of communication that pow pertains to
set	setting of security game
sk/SK	secret key
$\mathbf{SK}$	set of secret keys
τ	tag
$\mathbf{T}$	set of tags
usk	user secret key
Ψ	system of ITIs
$\mathcal{Z}$	UC environment

Bibliography

[AGKS05] Masayuki Abe, Rosario Gennaro, Kaoru Kurosawa and Victor Shoup. *Tag-KEM/DEM: A New Framework for Hybrid Encryption and A New Analysis of Kurosawa-Desmedt KEM*. Advances in Cryptology – EUROCRYPT 2005. Ed. by Ronald Cramer. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 128–146. ISBN: 978-3-540-32055-5.

[BMPR20] Christian Badertscher, Ueli Maurer, Christopher Portmann and Guilherme Rito. *Revisiting (R)CCA Security and Replay Protection*. Cryptology ePrint Archive, Report 2020/177. 2020.

[BCNP04] B. Barak, R. Canetti, J. B. Nielsen and R. Pass. *Universally composable protocols with relaxed set-up assumptions*. 45th Annual IEEE Symposium on Foundations of Computer Science. 2004, pp. 186–195.

[BDPR98] Mihir Bellare, Anand Desai, David Pointcheval and Phillip Rogaway. *Relations Among Notions of Security for Public-Key Encryption Schemes.* Vol. 1462. Aug. 1998, pp. 26–45.

[BP04] Mihir Bellare and Adriana Palacio. *Towards Plaintext-Aware Public-Key Encryption Without Random Oracles*. Advances in Cryptology - ASIACRYPT 2004. Ed. by Pil Joong Lee. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, pp. 48–62. ISBN: 978-3-540-30539-2.

[BR95] Mihir Bellare and Phillip Rogaway. *Optimal asymmetric encryption*. Advances in Cryptology — EUROCRYPT'94. Ed. by Alfredo De Santis. Berlin, Heidelberg: Springer Berlin Heidelberg, 1995, pp. 92–111. ISBN: 978-3-540-44717-7.

[BGMOS21] Wasilij Beskorovajnov, Roland Gröll, Jörn Müller-Quade, Astrid Ottenhues and Rebecca Schwerdt. *A New Security Notion for PKC in the Standard Model: Weaker, Simpler, and Still Realizing Secure Channels*. Cryptology ePrint Archive, Paper 2021/1649. 2021.

[BGMOS22] Wasilij Beskorovajnov, Roland Gröll, Jörn Müller-Quade, Astrid Ottenhues and Rebecca Schwerdt. *A New Security Notion for PKC in the Standard Model: Weaker, Simpler, and Still Realizing Secure Channels*. Public-Key Cryptography – PKC 2022. Ed. by Goichiro Hanaoka, Junji Shikata and Yohei Watanabe. Cham: Springer International Publishing, 2022, pp. 316–344. ISBN: 978-3-030-97131-1.

[BCHK07] Dan Boneh, Ran Canetti, Shai Halevi and Jonathan Katz. *Chosen-ciphertext security from identity-based encryption*. SIAM Journal on Computing 36.5 (2007), pp. 1301–1328.

[BF01] Dan Boneh and Matthew K. Franklin. *Identity-Based Encryption from the Weil Pairing*. CRYPTO 2001. Ed. by Joe Kilian. Vol. 2139. LNCS. Springer, Heidelberg, 2001, pp. 213–229.

[BDK+18] Joppe Bos, Leo Ducas, Eike Kiltz, T Lepoint, Vadim Lyubashevsky, John M. Schanck, Peter Schwabe, Gregor Seiler and Damien Stehle. *CRYSTALS - Kyber: A CCA-Secure Module-Lattice-Based KEM*. 2018 IEEE European Symposium on Security and Privacy. 2018, pp. 353–367.

[BIL21] Xavier Boyen, Malika Izabachène and Qinyi Li. *Secure Hybrid Encryption in the Standard Model from Hard Learning Problems*. Post-Quantum Cryptography. Ed. by Jung Hee Cheon and Jean-Pierre Tillich. Cham: Springer International Publishing, 2021, pp. 399–418. ISBN: 978-3-030-81293-5.

[Bre16] Joachim Breitner. *Lazy Evaluation: From natural semantics to a machine-checked compiler transformation*. Ph.D. thesis. Karlsruher Institut für Technologie, Fakultät für Informatik, Apr. 2016.

[Can00] Ran Canetti. *Security and composition of multiparty cryptographic protocols*. Journal of CRYPTOLOGY 13.1 (2000), pp. 143–202.

[Can01] Ran Canetti. *Universally composable security: A new paradigm for cryptographic protocols*. Proceedings 42nd IEEE Symposium on Foundations of Computer Science. IEEE. 2001, pp. 136–145.

[Can04] Ran Canetti. *Universally composable signature, certification, and authentication*. Proceedings. 17th IEEE Computer Security Foundations Workshop, 2004. IEEE. 2004, pp. 219–233.

[CHK04] Ran Canetti, Shai Halevi and Jonathan Katz. *Chosen-Ciphertext Security from Identity-Based Encryption*. Advances in Cryptology - EUROCRYPT 2004. Ed. by Christian Cachin and Jan L. Camenisch. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, pp. 207–222. ISBN: 978-3-540-24676-3.

[CK01] Ran Canetti and Hugo Krawczyk. *Analysis of key-exchange protocols and their use for building secure channels*. International conference on the theory and applications of cryptographic techniques. Springer. 2001, pp. 453–474.

[CK02] Ran Canetti and Hugo Krawczyk. *Universally composable notions of key exchange and secure channels*. International Conference on the Theory and Applications of Cryptographic Techniques. Springer. 2002, pp. 337–351.

[CKN03] Ran Canetti, Hugo Krawczyk and Jesper B. Nielsen. *Relaxing Chosen-Ciphertext Security*. Advances in Cryptology - CRYPTO 2003. Ed. by Dan Boneh. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, pp. 565–582. ISBN: 978-3-540-45146-4.

[CR02] Ran Canetti and Tal Rabin. *Universal Composition with Joint State*. Cryptology ePrint Archive, Report 2002/047. 2002.

[CHH+09] Seung Geol Choi, Javier Herranz, Dennis Hofheinz, Jung Yeon Hwang, Eike Kiltz, Dong Hoon Lee and Moti Yung. *The Kurosawa–Desmedt key encapsulation is not chosen-ciphertext secure*. Information processing letters 109.16 (2009), pp. 897–901.

[CFZ14] Sherman S. M. Chow, Matthew Franklin and Haibin Zhang. *Practical Dual-Receiver Encryption*. Topics in Cryptology – CT-RSA 2014. Ed. by Josh Benaloh. Cham: Springer International Publishing, 2014, pp. 85–105. ISBN: 978-3-319-04852-9.

[CS02] Ronald Cramer and Victor Shoup. *Design and Analysis of Practical Public-Key Encryption Schemes Secure against Adaptive Chosen Ciphertext Attack*. SIAM Journal on Computing 33 (Jan. 2002).

[DDA13] Angsuman Das, Sabyasachi Dutta and Avishek Adhikari. *Indistinguishability against Chosen Ciphertext Verification Attack Revisited: The Complete Picture*. ProvSec 2013. Ed. by Willy Susilo and Reza Reyhanitabar. Vol. 8209. LNCS. Springer, Heidelberg, 2013, pp. 104–120.

[DLKY04] Theodore Diament, Homin K. Lee, Angelos D. Keromytis and Moti Yung. *The dual receiver cryptosystem and its applications*. Proceedings of the 11th ACM Conference on Computer and Communications Security, CCS 2004, Washington, DC, USA, October 25-29, 2004. Ed. by Vijayalakshmi Atluri, Birgit Pfitzmann and Patrick D. McDaniel. ACM, 2004, pp. 330–343.

[DH76] W. Diffie and M. Hellman. *New directions in cryptography*. IEEE Transactions on Information Theory 22.6 (1976), pp. 644–654.

[DY83] D. Dolev and A. Yao. *On the security of public key protocols*. IEEE Transactions on Information Theory 29.2 (1983), pp. 198–208.

[GM84] Shafi Goldwasser and Silvio Micali. *Probabilistic encryption*. Journal of Computer and System Sciences 28.2 (1984), pp. 270–299. ISSN: 0022-0000.

[HHK10] Javier Herranz, Dennis Hofheinz and Eike Kiltz. *Some (in)sufficient conditions for secure hybrid encryption*. Inf. Comput. 208 (Nov. 2010), pp. 1243–1257.

[HLM03] Jonathan Herzog, Moses Liskov and Silvio Micali. *Plaintext Awareness via Key Registration*. Advances in Cryptology - CRYPTO 2003. Ed. by Dan Boneh. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, pp. 548–564. ISBN: 978-3-540-45146-4.

[ISO06] *Information technology — Security techniques — Encryption algorithms — Part 2: Asymmetric ciphers*. Standard. Geneva, CH: International Organization for Standardization, May 2006.

[KY06] Jonathan Katz and Moti Yung. *Characterization of Security Notions for Probabilistic Private-Key Encryption*. Journal of Cryptology 19 (May 2006), pp. 67–95.

[Kil06] Eike Kiltz. *Chosen-ciphertext security from tag-based encryption*. Theory of Cryptography Conference. Springer. 2006, pp. 581–600.

[KMP14] Eike Kiltz, Daniel Masny and Krzysztof Pietrzak. *Simple Chosen-Ciphertext Security from Low-Noise LPN*. PKC 2014. Ed. by Hugo Krawczyk. Vol. 8383. LNCS. Springer Berlin Heidelberg, 2014, pp. 1–18.

[KD04] Kaoru Kurosawa and Yvo Desmedt. *A New Paradigm of Hybrid Encryption Scheme*. Advances in Cryptology – CRYPTO 2004. Ed. by Matt Franklin. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, pp. 426–442. ISBN: 978-3-540-28628-8.

[KT14] Kaoru Kurosawa and Le Trieu Phong. *Kurosawa-Desmedt Key Encapsulation Mechanism, Revisited*. Progress in Cryptology – AFRICACRYPT 2014. Ed. by David Pointcheval and Damien Vergnaud. Cham: Springer International Publishing, 2014, pp. 51–68. ISBN: 978-3-319-06734-6.

[MRY04] Philip D. MacKenzie, Michael K. Reiter and Ke Yang. *Alternatives to Non-malleability: Definitions, Constructions, and Applications (Extended Abstract)*. TCC 2004. Ed. by Moni Naor. Vol. 2951. LNCS. Springer Berlin Heidelberg, 2004, pp. 171–190.

[MP12] Daniele Micciancio and Chris Peikert. *Trapdoors for Lattices: Simpler, Tighter, Faster, Smaller*. Advances in Cryptology – EUROCRYPT 2012. Ed. by David Pointcheval and Thomas Johansson. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 700–718. ISBN: 978-3-642-29011-4.

[MF21] Arno Mittelbach and Marc Fischlin. *The Theory of Hash Functions and Random Oracles - An Approach to Modern Cryptography*. Information Security and Cryptography. Springer, 2021.

[NMO06] Waka Nagao, Yoshifumi Manabe and Tatsuaki Okamoto. *A Universally Composable Secure Channel Based on the KEM-DEM Framework*. Vol. 89-A. Jan. 2006, pp. 28–38. ISBN: 978-3-540-24573-5.

[NHKJ12] Geontae Noh, Dowon Hong, Jeong Ok Kwon and Ik Rae Jeong. *A Strong Binding Encryption Scheme from Lattices for Secret Broadcast*. IEEE Communications Letters 16.6 (2012), pp. 781–784.

[PSJ12] Sumit Kumar Pandey, Santanu Sarkar and Mahabir Prasad Jhanwar. *Relaxing IND-CCA: Indistinguishability against Chosen Ciphertext Verification Attack*. SPACE 2012. Ed. by Andrey Bogdanov and Somitra Kumar Sanadhya. Vol. 7644. Lecture Notes in Computer Science. Springer, 2012, pp. 63–76.

[Sch24] Rebecca Schwerdt. *On the Optimal Weakness of IND-SB-CPA for Realizing Secure Message Transfer*. Submitted to PKC 2024.

[SBB+23a] Rebecca Schwerdt, Laurin Benz, Wasilij Beskorovajnov, Sarai Eilebrecht, Jörn Müller-Quade and Astrid Ottenhues. *Sender-binding Key Encapsulation*. Cryptology ePrint Archive, Paper 2023/127. 2023.

[SBB+23b] Rebecca Schwerdt, Laurin Benz, Wasilij Beskorovajnov, Sarai Eilebrecht, Jörn Müller-Quade and Astrid Ottenhues. *Sender-binding Key Encapsulation*. Public-Key Cryptography – PKC 2023. Ed. by Alexandra Boldyreva and Vladimir Kolesnikov. Vol. 13940. Lecture Notes in Computer Science. Springer, 2023, pp. 744–773.

[Sho01] Victor Shoup. *A Proposal for an ISO Standard for Public Key Encryption*. Cryptology ePrint Archive, Paper 2001/112. 2001.

[WM11] M.E. Whitman and H.J. Mattord. *Principles of Information Security*. Cengage Learning, 2011. ISBN: 9781133172932.