RUHR UNIVERSITÄT BOCHUM

On the Selective Opening Security of Public-Key Encryption

Dissertation Thesis

Felix Heuer

RUHR UNIVERSITÄT BOCHUM

On the Selective Opening Security of Public-Key Encryption

Dissertation Thesis

Felix Heuer

Reviewer: Prof. Dr. Kiltz
Prof. Dr. Jager

Date of oral examination: 13.06.2017

Acknowledgments

There are many people I am deeply grateful to and who I'd like to thank for supporting me in lots of different ways throughout the last three years.

First and foremost I would like to thank my advisor Eike Kiltz. He offered plenty of time for many meetings and never grew tired of explaining things (over and over again). I appreciate his honest and helpful feedback as well as giving me confidence in the quality of my talks. I'm thankful that Selective Opening emerged as a topic I gladly worked on for the last years. Beyond, Eike let me attend numerous workshops and conferences, allowing me to get to know many folks from our community and have worthwhile discussions. I particularly enjoyed CRYPTO'15 and TCC'16 in Tel Aviv where I fell in love with hummus.

Also thanks to Tibor who instantly agreed to review my dissertation and be part of the committee.

Further, I'd like to thank my coauthors Eike Kiltz, Tibor Jager, Sven Schäge, Krzysztof Pietrzak and Bertram Poettering for collaborating.

While I'm certain that every crypto group out there claims to be the most terrific, there is no conceivable way it's not you. It is astonishing how well our groups get along and the work environment with all of you is second to none. Please keep being magnificent.

The above holds especially for Eike's group, in particular for the seniorish or former members Saqib, Bertram, Federico, Manuel and Benedikt with whom I shared much of my time here. You. are. awesome!

Particular thanks go to Marion and Anja, the good souls of our groups. They do not only help us deal with administrative barriers but are the very foundation for our good work environment.

Special thanks got to Elena Kirshanova with whom I had the pleasure of sharing an office for three great years. I enjoyed all our discussions, mocking (Panda is still around, protesting silently) and fun as well as mental and dietary support whenever there was a submission deadline or notification coming up. Also, thanks for proofreading every single submission of mine and staring me to death on essentially *any* occasion.

Thanks to Saqib for lots of help and 'guadiance' in my early days as well as still ongoing discussions about everything including cryptography, news, TV series and the English language. (No, I won't stop calling it an 'n-tipede', cow $\neq$ cowl, and I still don't understand how your definition of 'my people' changes.)

Thanks to Bertram for his patience, endless scientific discussions and making me spot every spacing issue in a text, 'thanks'. I enjoyed debating hilarious and—let's say—provocative concepts (Safari!) and am confident that, eventually, one of your business ideas will make it.

Thanks to my new fantastic roommates Tinka and Ralph for providing chamomile tea when needed. I'm looking forward to our time in the office and more discussions about crypto and grooming.

Thanks to Lisa for wondering how I do during the more stressful period of writing my thesis. I guess back then it didn't seem like it but I really appreciated it.

Thanks to Federico for sending me the latest *Rick and Morty* - Season 3 trailer whose appearance went unnoticed by *me*.

Thanks to Saqib, Benenickt, Federico, Ralph, Bertram, Giorgia and Virginie for proofreading my dissertation on such short notice. Special thanks to Elena and Manuel (he ended up reading in the Bahnhofsmission Frankfurt (Main (station))) for checking my whole thesis in no time.

It means a great deal to me that some of my mates spared no effort to join for my defense: Thanks to Elena, Saqib and Ilya. Also, thanks to Sebastian for attending...

While getting along well at work shouldn't be taken for granted, I was even more fortunate to have splendid holidays in the course of the last years.

Thanks to Ilya, Robert and Jiaxin for our great trip along the East Coast and learning that preparing pancakes in a microwave is possible (yet not advisable).

Thanks to Federico and Manuel for our amazing West Coast trip (it doesn't help getting rid of prejudices if you witness a theft right after arriving in Tenderloin. "Huh, this is actually salt. Federico, can I borrow your hat?". "Where's Federico?!"), as well as our incredible trip through Canada, the USA and National Parks (Maple syrup cookies! "Why would you go to Melvindale?" isn't really assuring if it comes from a Border Patrol Agent. "Where's Federico?!". Waiting for the Grand Geyser to erupt is a pain ("surely just another 20 minutes..."), creepy music at night on a deserted road in Yellowstone is fun and Navajo time zones are as confusing as it is to stumble across an old car wreck in a canyon. Anyway, all I hear is *la la*...).

Thanks to Federico, Manuel and Benedikt for the trip to Cardiff (and London) (*Now* I'd actually like to visit the Doctor Who Experience. 'Thanks', Manuel.)

Further thanks to Manuel and the other giraffe for our trip through UK. Our shoes shall remember the patches of slightly moist moss until their last day!

Also thanks to Bertram for having a stunning time in Angkor and Vịnh Hạ Long and bravely enduring my shoppingmania spending six-digit figures on the last day.

All of these trips created memories for a lifetime and I hope you enjoyed them as much as I did. I look forward to new holidays with you.

I'd like to thank all my friends from Wetter for many fun evenings and celebrations throughout all these years since leaving high school. Specific thanks to Patrick who is a guarantee of great fun and a marvelous friend to have.

Finally but most importantly I'd like to thank my family and in particular my parents for their relentless support extending way beyond the last couple of years of studies but especially throughout my grumpy days of stressful thesis-writing.

Bochum, June 2017

Contents

Introduction

Public-Key Cryptography Through the Ages Since its discovery a mere 40 years ago public-key cryptography has come a long way from existing as a proof of concept to highly efficient constructions resisting strong attacks.

The origin of public-key cryptography can be traced back to the seminal works of Merkle [Mer78] as well as Diffie and Hellman [DH76] at the end of the 1970s. Merkle proposed a first protocol for two parties to exchange a key. Thereby the time of an attacker to obtain the key would rise quadratic in the time of the two parties. Diffie and Hellman introduced the first key exchange protocol later proven secure under a computational complexity assumption. The first public-key scheme was published in 1977 by Rivest, Shamir and Adleman [RSA78]. While all protocols mentioned so far are not considered secure given today's understanding, they served as a proof of feasibility for public-key encryption.

During the next decade attention was shifted towards achieving security against passive adversaries. Initially, it was captured by the notion of *semantic security*, demanding that an attacker shall not be able to derive any 'meaningful' information given a ciphertext. Semantic security turned out to be equivalent to the easier to handle notion of indistinguishability under chosen-plaintext attacks (IND-CPA) [GM84]. In a nutshell, a public-key scheme is IND-CPA secure if no attacker, given the public key, can tell apart encryptions of plaintexts of their choosing. Clearly, IND-CPA security has to be a minimal confidentiality requirement for public-key encryption (PKE) as the public key is openly available. Goldwasser and Micali gave the first IND-CPA secure public-key scheme [GM82]. Though, their scheme is of little use in practice as merely encrypts individual bits. The first practical scheme was proposed two years later by Elgamal [Gam84] whose scheme is closely related to the Diffie-Hellman key exchange. Later, Goldreich and Levin showed that IND-CPA public-key encryption can be constructed from any trapdoor one-way permutation [GL89].

Only in the 1990s schemes secure under chosen-ciphertext attacks (IND-CCA) were devised. The notion of IND-CCA security is a natural extension of IND-CPA security where an attacker may request decryptions of ciphertexts of their choice[1] when asked to tell apart encryptions of plaintexts under their control [RS92]. As a first step, Naor and Yung [NY90] constructed public-key encryption secure against so-called lunch time attacks (IND-CCA1). Building on this work Dolev, Dwork and Naor presented the first—yet impractical—scheme secure under adaptive chosen-ciphertext attacks (IND-CCA2) [DDN91]. Towards the end of the 1990s Cramer and Shoup presented an improved IND-CCA secure version of the Elgamal encryption scheme [CS98] that was later generalized and captured in the concept of Hash Proof Systems [CS02]. IND-CCA security quickly became—and still is—the de facto confidentiality notion for encryption.

A key example for the practical relevance of IND-CCA secure PKE was given by Bleichenbacher [Ble98]. He performed an attack on RSA as standardized in PKCS#1 v1.5 (RFC 2313) exploiting that when submitting a ciphertext to a server for decryption one would learn whether the decryption of a ciphertext results in a plaintext with syntactically correct padding or not. Fortunately, research on IND-CCA secure public-key encryption was initiated early enough such that a replacement was already at hand. RSA-OAEP, having an infamous history of (in)security (see Section 2.3) on its own, took over and was standardized in PKCS#1 v2.0 (RFC 2437).

Many efficient PKE schemes employed in practice have security proofs in the *random oracle model* (ROM) where we assume hash functions to behave like a truly random function. Formally introduced by Bellare and Rogaway [BR93], the idea of 'random looking functions' was already present in the work of Fiat and Shamir [FS87]. As a truly random hash function does not have a compact description, the hash function is provided as a *oracle* implemented by the environment. The power of the random oracle methodology in proofs stems from its oracle nature: a) The environment can observe any query made to the random oracle (in particular by an attacker) and b) the environment can *program* values into the random oracle (as long as their correctly distributed). Clearly, random oracles cannot exist in practice and conclusions drawn from proofs in the random oracle model were questioned due to [CGH98]. They gave contrived schemes that are provably secure in the ROM but become insecure if the random oracle is instantiated with *any* concrete hash function. Fortunately, this behavior has not been observed in practice. Today, proofs in the random oracle model are still indispensable as they strengthen the belief that a cryptographic construction is sound. Prime examples for IND-CCA secure schemes with security proofs in the ROM are the Optimal Asymmetric Encryption Padding (OAEP) [BR95, Sho02, KP09], Diffie-Hellman Integrated Encryption Scheme (DHIES) [BR97, ABR01, SBZ02] or the Fujisaki-Okamoto transform [FO99], e.g., instantiated with Elgamal encryption [ElG84].

[1] up to technical restrictions

Another idealized model of computation is known as *ideal cipher model* dating back to the early works of Shannon [Sha49]. Here we assume that a blockcipher, i.e., a family of keyed permutations, behaves ideally in the sense that for each key the permutation is truly random. Uninstantiability results reminiscent to those obtained in the random oracle model exist for the ideal cipher model, too [Bla06]. In fact, the random oracle and ideal cipher model are equivalent [CPS08, DKT16, DS16].

The Need for Stronger Security Models In 1996 Kocher introduced a new class of practical attacks nowadays referred to as side-channel attacks [Koc96a]. These attacks exploit weaknesses in the *implementation* of cryptography functionality rather than the (abstract) cryptographic algorithms itself. Security under such attacks had never been thoroughly analyzed as they are taking place beyond the standard attacker model. In principle, a side-channel attack can be conducted from many different angles. For instance: time [Koc96b], power consumption [KJJ99] or acoustic noise [GST14]. While an attacker remains passive in aforementioned attacks, an intervening adversary may find even more ways to extract information. For instance, cold boot attacks [HSH+08] exploiting data remanence or fault injection as introduced by Boneh *et al.* in the 'Bellcore-Attacks' [BDL97].

The community was aware of the potential threat posed by side-channel attacks since the early works on public-key encryption. While practical countermeasures, like blinding were devised early [Koc96a], theoretical models of computation that leaks information and provable security against side-channel attacks picked up pace only in the last decade [NS09].

While there is some progress in eliminating some side-channels (e.g. threshold implementations [GP99] as an instance of masking [NRR06]) practitioners are in a dire situation as they can only react, once a side-channel condition has been discovered [MPL+11].

We conclude that there *are* attacks that lie outside of our standard attacker model and we ought to strengthen our notions of security.

Selective Opening Attacks

Public-Key Encryption in Practice Let us consider how public-key encryption is used in practice and what kind of attacks might exist. Clearly, we have to consider more than two parties sending and receiving encrypted plaintexts potentially depending upon each other, for instance as they send emails, use messengers or connect to servers. As for the adversary's capabilities we have to assume that they have means to compromise the integrity of users' machines, possibly due to malware or even hardware installed on users' systems. For instance, as we came to learn in 2013, the NSA regularly intercepted deliveries of computer hardware in order to install malicious soft- and hardware (for instance, see the NSA ANT catalog [Sta13]). Less invasively, an attacker might as well simulate users participating in large protocols or come in form of colluding users deliberately sharing their secrets to gain additional information. A similar scenario naturally arises in multi-party computation where we assume secure channels between parties. Since a party might become corrupted, we would need the encryption on the channels to have stronger security guarantees than implied by standard security notions.

Independently of how such an attack is launched, a *corruption* would potentially leak all the information a user possesses, in particular, its secret key or sent plaintexts as well as the random coins that were used when encrypting the plaintexts.

Let us discuss why it is reasonable to assume that random coins of a user might leak. Why would they store their random coins after encryption in the first place? The immediate answer is that we aim at being as conservative as possible: Securely erasing data is expensive and thus, the random coins might exist even after being 'deleted' on the user's machine. Beyond that, keeping the random coins might serve a desired functionality. Encryptions under a public-key scheme may be seen as a commitment to a plaintext, while the random coins serve as opening information.[2]

Even more justification can be found in practice where the randomness might be deleted after use but may be easily recoverable for an attacker. As generating 'true' randomness is expensive one might rely on pseudorandom generators to derive randomness for encryption. Alternatively, one might use a pseudorandom function evaluated on a plaintext to obtain random coins. Now, if the employed random number generator is weak, or worse: backdoored [BLN16], or the adversary holds the pseudorandom functions's key we have to assume that the randomness of users leaks.

We end up with the setting of *selective opening* attacks (under sender corruption) that can be modeled as follows (see Figure 1): One user, the receiver, generates a key pair and many users encrypt plaintexts using the receiver's public key (of course using fresh and independent random coins). Again the adversary controls the plaintext distribution—and may have arbitrary ciphertexts decrypted in the case of a SO-CCA attack. On top of that the adversary is allowed to corrupt users thereby revealing the plaintexts they encrypted and the random coins they used. Again, the adversary's goal is to derive new information about the plaintexts.

Clearly, we cannot expect any confidentiality for plaintexts sent by a corrupted user but what about the confidentiality of plaintexts sent by users that remain uncorrupted?

[2]E.g., any correct, IND-CPA secure public-key encryption scheme is a computationally hiding, perfectly binding commitment scheme.

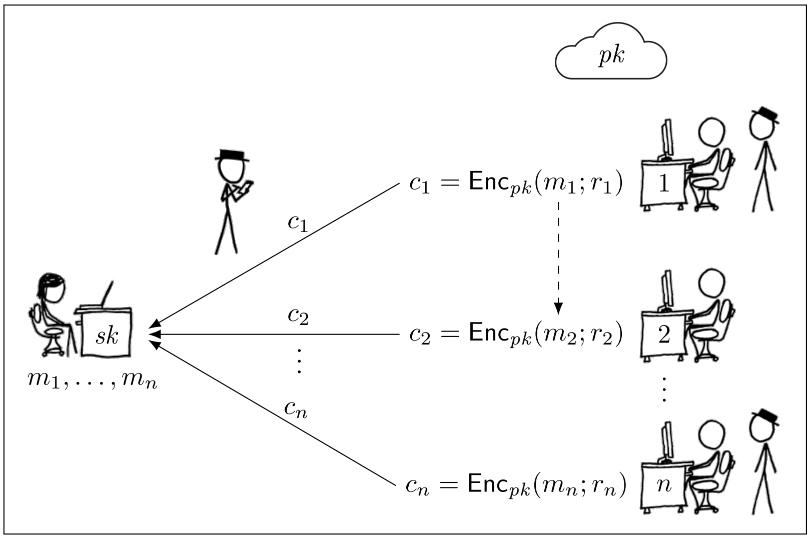

Figure 1: Depiction [Mun] of a selective opening attack with sender corruption. Senders $1, \ldots n$ sitting on the right. They encrypt possibly related plaintexts (e.g., m_2 shall depend on m_1). Ciphertexts are sent to the receiver (on the left). The adversary (black hat) knows pk, observes ciphertexts $c_1, \ldots, c_n$ and corrupts users 1 and n thereby learning $(m_1, r_1), (m_n, r_n)$. The adversary seeks to obtain non-trivial (i.e., not given by knowledge of m_1 and m_n) information on plaintexts encrypted by uncorrupted users, e.g., m_2.

Can we have any security guarantees for plaintexts sent by them? Selective opening (SO) security is provided if in this situation the confidentiality of plaintexts in 'unopened' ciphertexts is still ensured. Intuitively, as all the encryptions occur independently of each other, standard indistinguishability (i.e., IND-CPA or IND-CCA) notions should imply their SO counterpart. Unfortunately, formal analysis reveals that this is not the case.

Hardness of Proving Selective Opening Security The hardness of establishing results on selective opening security is well-studied and does *not* stem from the multi-user setting. Here a standard hybrid argument ensures that standard IND-{CPA,CCA} security implies IND-{CPA, CCA} security. Generally, the reduction loses the number of encryptions per user and the number of users as a factor in tightness [BBM00]. However, for certain PKE schemes based on assumptions allowing for challenge rerandomization, the reduction can be tightened [BBM00].

In contrast, the infeasibility of proving IND-CPA secure PKE schemes selective opening secure can be traced back to two peculiarities of selective opening attacks: the occurrence of related plaintexts and revealing the randomness of corrupted senders. In the SO setting every sender uses fresh randomness (in particular) independently of other users. Hence, one might be surprised that a hybrid argument to obtain SO security from IND-CPA security seems impossible to push through. As we will sketch in Section 1.2.3 the dependency of plaintexts seemingly only allows for reductions with an exponential loss. As for the disclosure of random coins it has been observed that standard security entails some notion of 'SO' security if *only* the plaintexts leak under corruption [Yil10].

Nevertheless, the central theme of this thesis is to establish results

On the Selective Opening Security of Public-Key Encryption Schemes.

Main Research Areas in Selective Opening Security

We proceed by discussing the major research topics in selective opening security before covering our contributions.

Notions of Selective Opening Security The study of selective opening attacks dates back to the work of Dwork *et al.* [DNRS99] introducing the "selective decommitment problem". Dwork *et al.* investigated the selective opening security of commitments schemes: Given commitments and allowing an attacker to *open* some of the commitments, what security guarantees can we expect the unopened commitments to have?

Formalizing suitable notions of SO security has proven to be highly non-trivial. Since the occurring plaintexts may depend on each other, opening ciphertexts usually leaks information on plaintexts still encrypted in (unopened) ciphertexts. Thus, it is not obvious how to capture that plaintexts in unopened ciphertexts shall remain *as confidential as possible.*

Two flavors of SO security have been introduced and studied in prior work: notions based on indistinguishability (IND-SO) and notions based on simulatability (SIM-SO).

For IND based notions [BHY09] an adversary may open arbitrary ciphertexts and is challenged to tell apart the originally encrypted plaintexts from fresh plaintexts that are as likely to occur as the original plaintexts. As computing these fresh plaintexts involves resampling from a distribution conditioned on the plaintexts revealed to an attacker, one usually restricts the distribution to be *efficiently conditionally resampleable* to ensure an efficient security experiment. Nevertheless, Böhl *et al.* [BHK12] adopted a notion from the commitment scheme setting [BHY09] where the resampleability restriction on the distribution is dropped. [BHK12] renamed IND-SO to *weak* IND-SO and introduced a strictly stronger notion, called *full* IND-SO, where an attacker may choose an arbitrary distribution.[3] On the one hand *full* IND-SO might be motivated from the application of encryption in practice where arbitrary distributions may arise, on the other hand *full* IND-SO security did neither lead to any meaningful results in understanding selective opening attacks nor is there any instantiation of a *fully* IND-SO-CPA secure PKE.

In contrast, SIM based notions (capturing semantic security in the SO setting) do not suffer from a restriction on the distribution. Boiled down, a scheme is SIM-SO secure if for every SO adversary there exists a simulator that can compute the same output without seeing any ciphertexts. Importantly, such simulators may corrupt senders to learn the plaintexts they (virtually) encrypted.

Both flavors, IND-SO and SIM-SO, may be considered under CPA and CCA attacks. We consider the naming '*weak* IND-SO' unfortunate and simply refer to the security notion as IND-SO security while mostly glossing over the existence of *full* IND-SO notions.

While almost all results consider SO security under sender corruption, similar notions modeling selective opening security under receiver corruption are defined in [BDWY11].

In this work we will mainly focus on two notions of selective opening security, indistinguishability-based selective opening security under chosen-plaintext attacks (IND-SO-CPA) in Part I and simulation-based selective opening security under chosen-ciphertext attacks (SIM-SO-CCA) in Part II.

As just discussed there is a whole zoo of security notions under selective opening attacks: (For the matter of this section) *weak* and *full* IND-SO, as well as SIM-SO, under CPA and CCA attacks. Many results cover the relations amongst notions of SO security as well as between standard and SO security.

Relations Amongst Notions of SO Security The notions of SIM-SO-CPA and *full* IND-SO-CPA security are incomparable (i.e., none implies the other) and constitute the strongest notions of security [BHK12, BDWY12, HR14], as both imply *weak* IND-SO-CPA (Figure 2). Any notion of SO-CPA security implies IND-CPA [BHK12]. The notion *weak* IND-SO-CPA is strictly weaker than *full* IND-SO-CPA ([BHK12]).

SIM-SO and *full* IND-SO seemingly differ in terms of achievability. There exist constructions of (even) SIM-SO-CCA secure public-key encryption schemes [FHKW10, Hof12], while until now there is not even a *full* IND-SO-CPA secure PKE.

Independently, notions for non-malleability under selective opening attacks were introduced and studied in [HLMC15].

[3] Assuming the existence of one-way functions one can easily construct distributions that do not support efficient conditional resampling.

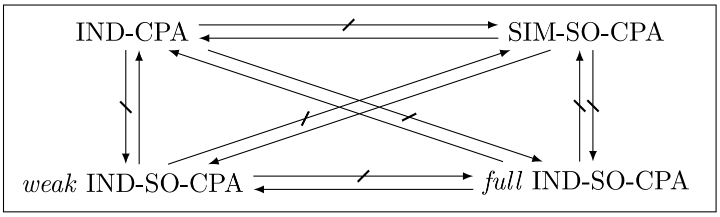

Figure 2: Relations amongst notions of IND-CPA and SO-CPA security [BHK12, HPW15]. Arrows show black-box implications. For striked arrows there exists a PKE scheme separating the notions.

Relations Amongst Standard and SO Security We summarize positive and negative results relating the notions of standard and selective opening security.

Separations A negative result was given by Bellare *et al.* [BDWY11]. They showed that IND-CPA security does not imply SIM-SO-CPA security, independently of the underlying distribution. They exploited that no 'committing' PKE scheme can be SIM-SO-CPA secure. Shortly afterwards, [BHK12] proved that IND-CPA is strictly weaker than *full* IND-SO-CPA. Hofheinz *et al.* separated IND-CCA and *weak* IND-SO-CCA [HR14] by constructing a PKE scheme that is IND-CCA secure but vulnerable under *weak* IND-SO-CCA attacks. Later, Hofheinz *et al.* could adopt the result to passive attacks [HRW16] as they gave a scheme that is (even) IND-CCA secure but breaks under *weak* IND-SO-CPA attacks. Their scheme relies on the existence of public-coin differing-inputs obfuscation and correlation-intractable hash functions that are "plausible" [HRW16] to exist. As to the state of existence of (at least) indistinguishability obfuscation, there have been as least as many attacks (most recently the widely applicable 'annihilation attacks' [MSZ16]) as proposed candidates. Currently, it is unclear whether the separation scheme of [HRW16] can be instantiated.

Implications The first positive result relating IND-CPA and IND-SO-CPA was given by [BY09] who transferred a result for commitment schemes [DNRS99] to the PKE setting: IND-CPA implies *weak* IND-SO-CPA when plaintexts come from a product distribution. Another result was contributed by [HR14]: IND-CPA implies *weak* IND-SO-CPA security in the generic group model [Sho97], at least for a certain class of PKE schemes including Elgamal and Cramer-Shoup.

Constructing Selective Opening Secure Public-Key Encryption The problem of constructing selective opening secure public-key encryption in the standard model has been solved by Bellare, Hofheinz, and Yilek [BHY09]. The authors show that *lossy encryption* [PW08] implies IND-SO-CPA security, while lossy encryption with *efficient opening* allows for SIM-SO-CPA security. Interestingly, it turns out that even the Golwasser-Micali scheme [GM82], the first 'provably secure' public-key encryption scheme, constitutes a lossy public-key scheme with efficient opening. This line of research is continued in [HLOV11] by Hemenway *et al.*, who show that re-randomizable encryption and statistically hiding two-round oblivious transfer imply lossy encryption. From a cryptographic point of view the above works solve the problem of finding SO-CPA secure encryption schemes, as there are several constructions of efficient lossy or re-randomizable encryption schemes, e.g., [PW08, BHY09, HLOV11]. Further *deniable encryption* [CDNO97] and techniques from *non-committing encryption* [CFGN96, HOR15] already allow for constructing SO secure PKE [Dac14].

SIM-SO secure constructions in the standard model usually suffer in efficiency from bit-wise encryption to ensure *efficient openability* or employing expensive building blocks [LP15, HJR16]. Recently, somewhat more efficient standard model SIM-SO-CPA secure PKE schemes have been proposed [HJR16].[4] An identity-based encryption scheme with selective-opening security under passive attacks was proposed by [BWY11].

As for security under active attacks, Hemenway *et al.* [HLOV11], Fehr *et al.* [FHKW10], Hofheinz [Hof12] describe SO-CCA secure encryption schemes. Whereas [HLOV11] can only handle adversaries that specify the set of users to be corrupted in one shot, [FHKW10, Hof12] also applies to adversaries with fully adaptive corruption capabilities. The work of [LP15] identifies special properties of a KEM, allowing to construct SIM-SO-CCA secure PKE.

Little attention has been payed to SO-CCA security under receiver corruption. [HPW15] shows how to obtain it from *non-committing encryption for receiver*.

Lately, SIM-SO-CCA security for identity-based encryption has been achieved in [LDL$^+$14].

Main Results of This Thesis

We proceed by summarizing the main results presented in this work.

Part I - Results in the Standard Model

Since notions of selective opening security were introduced [DNRS99] it was an open question whether they constitute a strictly stronger notion than standard security notions in either, the CPA or CCA setting. Only recently both questions were settled as Hofheinz *et al.* [HR14, HRW16] constructed contrived schemes that are IND-CCA secure but break under an IND-SO-CCA and even an IND-SO-CPA selective opening

[4] Encryption still processes the plaintext bit-wise though.

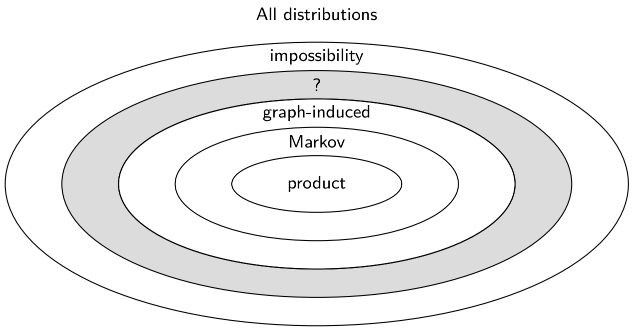

Figure 3: Overview on results addressing 'IND-CPA $\Rightarrow$ IND-SO-CPA' depending on the underlying plaintext distribution. A positive result for product distributions was know prior to our work [DNRS99, BY09]. We contribute positive results for Markov and certain graph-induced distributions. The impossibility result of [HR14] applies to certain 'secret-sharing distributions' leaving a gray area where we do not know whether IND-CPA implies IND-SO-CPA security.

attack (Figure 3). Thus, a general implication of IND-SO-CPA from IND-CPA security is impossible.

On the other hand, the only positive result holds when the plaintexts are independent of each other, i.e., stem from a product distribution [DNRS99, BY09], see Figure 3. For arbitrary distributions the only known reduction from IND-SO-CPA to IND-CPA security loses an exponential factor, thus, leaving the following question unanswered:

Does standard security imply selective opening security
for any non-trivial distribution?

We answer the question in the positive by giving a new black-box reduction whose loss depends on the dependency-structure of the plaintext distribution. In particular, for some concrete distributions our reduction is polynomial and constitutes the first positive results in almost 20 years:

- IND-CPA security implies IND-SO-CPA security for any public-key scheme if the plaintexts come from certain memoryless distributions including Markov distributions (Section 1.3).

- We give a hybrid argument when the plaintexts come from a distributions that can be *decomposed* into multiple independent distributions. Our result entails the positive result of [DNRS99, BY09] (Section 1.4).

- All results in the CPA setting can be lifted to the relation between IND-CCA and IND-SO-CCA security (Section 1.5).

The results presented in Chapter 1 are based on joint work with Eike Kiltz, Krzysztof Pietrzak and Georg Fuchsbauer published in [FHKP16].

Part II - Results in Idealized Models of Computation

For our first contribution in this part we move the focus from results applying to all public-key schemes to concrete schemes. More precisely, we study three well-known generic transformations that turn relatively weak primitives with security in the sense of one-wayness to IND-CCA secure public-key encryption schemes in the random oracle model. We show that all three transformations actually give rise to SIM-SO-CCA secure schemes under exactly the same assumptions employed to obtain IND-CCA security. More precisely, for the following transformations selective opening security comes 'for free' in the random oracle model:

- A transformation from any one-way PCA key encapsulation mechanism. (Section 2.2)
- The OAEP padding followed by a trapdoor permutation. (Section 2.3)
- The Fujisaki-Okamoto transformation for any one-way secure public-key scheme. (Section 2.4)

Most notably, the first transformation covers an instantiation of the well-known DHIES (Hashed Elgamal) scheme [BR97], while the second covers the already mentioned RSA-OAEP scheme [BR95]. Both DHIES and RSA-OAEP are important building blocks in several standards for public-key encryption and key exchange protocols. We also show a similar result for the well-known Fujisaki-Okamoto transformation that can generically turn a one-way secure public-key encryption system into a IND-CCA secure public-key encryption system.

Further, all cryptographic primitives beyond the one-way primitives by any of the three transformations exist information-theoretically. Hence, only assuming the existence of any of the cryptographic primitives listed above implies the existence of SIM-SO-CCA secure PKE in the random oracle model.

These results presented in Chapter 2 are joint work with Tibor Jager, Eike Kiltz and Sven Schäge and are published in [HJKS15, HJSK16].

For the last contribution we investigate the selective opening security of highly efficient public-key hybrid encryptions schemes as employed in practice. Considering that users in practice are exposed to the threats modeled in selective opening attacks, and given that the classical confidentiality notions are formally weaker than notions of SO security, the following question is immediate:

Are users 'safe' if they trust in a public-key scheme designed towards the goal of 'only' standard confidentiality?

The question calls for a thorough SO analysis of all encryption schemes covered by international standards. The facts that all PKE schemes that so far were formally confirmed to be SO secure require heavy building blocks and that practitioners systematically avoid these for reasons of efficiency suggest that likely most practical schemes would not withstand SO attacks. Fortunately, as we can show, virtually all practical PKE constructions provably do meet security under selective opening attacks.

Our approach is complementary to that of prior works [FHKW10, LP15]. Instead of analyzing the asymmetric building blocks of constructions, we observe that selective opening security is tightly linked to the security of the symmetric building blocks. We introduce a specific property called *simulatability* for blockcipher-based data encapsulation mechanisms (DEMs) that is met by virtually all DEMs used in practice. Simulatability guarantees that if a corresponding DEM is combined with any IND-CCA secure key encapsulation mechanism (KEM), then the overall hybrid PKE scheme achieves SIM-SO-CCA security in the ideal cipher model.

Our results are:

- The IND-CCA security of hybrid encryption employing any IND-CCA secure KEM can be lifted to SIM-SO-CCA security if the DEM is simulatable. (Sections 3.2 and 3.4)
- The popular modes of operation CTR, CBC, CCM, and GCM are simulatable. (Section 3.3)

The results of Chapter 3 are joined work with Bertram Poettering and published in [HP16].

Part I

Results in the Standard Model

Chapter 1

Standard Security implies Selective Opening Security[1]

In this chapter we present the first non-trivial positive results on selective opening security in the standard model. We show that IND-CPA security implies IND-SO-CPA security for a class of 'relatively memoryless' distributions. We consider *graph-induced* distributions where dependencies among plaintexts correspond to edges in a graph and show that IND-CPA implies IND-SO-CPA security for all graph-induced distributions that satisfy a certain *low connectivity* property.

In particular, our result holds for the class of Markov distributions, i.e., distributions on vectors $(m_1, \ldots, m_n)$ where all information relevant for the distribution of m_i is present in m_{i-1}. For instance, our results cover distributions where plaintext m_i contains all previous plaintexts $m_1, \ldots, m_{i-1}$ (e.g. email conversations) or distributions where plaintexts are increasing integers, i.e., $m_1 \leq m_2 \leq \ldots \leq m_n$.

Note that a positive result on IND-SO-CPA (rather than SIM-SO-CPA) security for all IND-CPA secure public-key encryption schemes for certain distributions is the best we can hope for: The negative result by Bellare *et al.* [BDWY12] rules out any such implication for SIM-SO-CPA security.

Further, recall the separation result 'IND-CPA $\not\Rightarrow$ IND-SO-CPA' by Hofheinz *et al.* [HRW16] which exploits distributions that arise from secret sharing. We note that there are still uncharted grounds in terms of distributions between our positive and the negative result by [HRW16] where it is unknown whether standard security implies SO security (see Figure 3).

[1] SOMETIMES

1.1 Notational Conventions and Experiments

Notation We distinguish the following operators for assigning values to variables: We use symbol '$\leftarrow$' when the assigned value results from a constant expression (including the output of a deterministic algorithm) and we write '$\leftarrow_\$$' when the value is sampled uniformly at random from a finite set, is the output of a randomized algorithm, or is sampled according to some distribution.

If f is a function or a deterministic algorithm that maps elements from a set A to a set B we use the notation $f\colon A \to B$. If f is a randomized algorithm from A to B we correspondingly write $f\colon A \to_\$ B$; in case the algorithm takes no input we write $f\colon \to_\$ B$. If R denotes the randomness space of an algorithm $f\colon A \to_\$ B$, we may write $f\colon A \times R \to B$ for its deterministic version. If $f\colon A \times B \to C$ is a function then for any $a \in A$ we write $f_a(\cdot)$ for the partially applied function $f_a\colon B \to C$, $b \mapsto f(a,b)$. If $f\colon A \to B$ is a function or a deterministic algorithm we let $[f] := f(A) \subseteq B$ denote the image of A under f; if $f\colon A \to_\$ B$ has randomness space R, we correspondingly let $[f] := f(A \times R) \subseteq B$ denote the set of all its possible outputs. We denote the disjoint union of two sets A, B with $A \cupdot B$.

For $a, b \in \mathbb{N}$, $a \leq b$, let $[a,b] := \{a, a+1, \ldots, b\}$ and $[a] := [1,a]$. For $n \in \mathbb{N}$ and $\mathcal{I} \subseteq [n]$ let $\overline{\mathcal{I}} := [n] \setminus \mathcal{I}$. For two bitstrings x, y we denote the concatenation of x and y by $x \,\|\, y$.

We use boldface letters to denote vectors which are of dimension $n \in \mathbb{N}$ if not specified otherwise. We let $|\mathbf{v}|$ denote the number of entries in $\mathbf{v}$. For $i \in [n]$ let v_i denote the i^{th} entry of $\mathbf{v}$. We write '$v \in \mathbf{v}$' if there exists an i such that $v = v_i$. For a set $\mathcal{I} = \{i_1, \ldots, i_{|\mathcal{I}|}\} \subseteq [n]$, $i_1 < \ldots < i_{|\mathcal{I}|}$ let $\mathbf{v}_{\mathcal{I}} := (v_{i_1}, \ldots, v_{i_{|\mathcal{I}|}}) \in S^{|\mathcal{I}|}$. For $\mathbf{v} = ((v_{1,1}, \ldots, v_{1,j}), \ldots, (v_{n,1}, \ldots, v_{n,j})) \in (S^j)^n$ let $\mathbf{v}_{\mathcal{I},k} := (v_{i_1,k}, \ldots, v_{i_{|\mathcal{I}|},k}) \in S^{|\mathcal{I}|}$.

We employ the Big O notation $\mathcal{O}/\ \Omega/\ \Theta$ for asymptotic behavior denoting upper/ lower/ upper and lower asymptotic bounds. We write $f(n) = \mathsf{poly}(n)$ for $f(n) = \mathcal{O}(n^c)$ for constant c and $f(n) = \mathsf{const}$ for $f(n) = \mathcal{O}(1)$.

Experiments Our security definitions are given in terms of *experiments* written in pseudocode. We write '$L \leftarrow \emptyset$' to initiate L as empty, independently of L's data type.

Within an experiment a (possibly) stateful *adversary* is explicitly invoked. An experiment may contain oracle procedures. We write $\mathcal{A}^{\mathcal{O}}$, to indicate that algorithm $\mathcal{A}$ has oracle access to $\mathcal{O}$. An oracle ends with a 'Return X', returning X to the algorithm that called the oracle. We syntactically distinguish between an oracle FUNC granting access to a functionality and the implementation of the functionality func itself.

If an algorithm $\mathcal{A}$ halts without any output we write $() \leftarrow_\$ \mathcal{A}$. An experiment terminates when a 'Stop with X' command is executed; X then serves as the output of the experiment. We write 'Abort' as an abbreviation for 'Stop with 0'. For a boolean expression B we may write 'Return B' and 'Stop with B'. Then expression B is evaluated and its truth value (encoded as a bit) is returned (resp. stopped with). We write $(a =_? b)$ for the boolean variable that is *true* iff $a = b$. For an event E let $\overline{\mathrm{E}}$ denote the complementary event. For a boolean expression X let $\neg X$ denote its negation.

We write $\Pr[\mathsf{Exp} \Rightarrow 1]$ for the probability of the event that experiment Exp terminates by running into a 'Stop with 1' instruction. An adversary *wins* an experiment if it stops with 1.

Our proofs in Part II employ sequences of experiments (see [BR06, Sho04b]). For better comparability we usually depict multiple experiments in the same figure. Thereby, certain instructions may only be executed in some experiments. A line ending with a comment of the form '$/\!/ \mathsf{Exp}_i - \mathsf{Exp}_j$' (resp. '$/\!/ \mathsf{Exp}_i$') is only executed when an experiment in $\mathsf{Exp}_i - \mathsf{Exp}_j$ (resp. Exp_i) is run. For an instruction within an oracle $\mathcal{O}$, we write '$/\!/ \mathcal{A}_i$' to indicate that the respective line is only executed when $\mathcal{O}$ was queried by $\mathcal{A}_i$.

1.2 Public-Key Encryption

In this section we recall the syntax of public-key encryption as well as the confidentiality of IND-CPA security and extend them to a setting where multiple plaintexts shall be protected simultaneously. We continue by defining confidentiality under *selective opening* attacks. To conclude, we sketch a naïve reduction with exponential loss from IND-SP-CPA to IND-CPA security that will allow for some early insights as to what restrictions have to be imposed to ensure an at most polynomial loss.

Definition 1.2.1 (public-key encryption scheme). A *public-key encryption* (PKE) scheme for a plaintext space $\mathcal{M}$ consists of a public-key space $\mathcal{PK}$, a secret-key space $\mathcal{SK}$, a ciphertext space $\mathcal{C}$, and a triple of efficient algorithms denoted with $\mathsf{PKE} = (\mathsf{PKE.Gen}, \mathsf{PKE.Enc}, \mathsf{PKE.Dec})$ of the form

$$\mathsf{PKE.Gen}\colon \to_\$ \mathcal{PK} \times \mathcal{SK} \quad \mathsf{PKE.Enc}\colon \mathcal{PK} \times \mathcal{M} \to_\$ \mathcal{C} \quad \mathsf{PKE.Dec}\colon \mathcal{SK} \times \mathcal{C} \to \mathcal{M} \cup \{\bot\} \ ,$$

where symbol '$\bot$' may be used to indicate errors. We assume that for all $(pk, sk) \in [\mathsf{PKE.Gen}]$, pk contains sk. The finite randomness space of $\mathsf{PKE.Enc}$ is typically denoted with $\mathcal{R}$. Correctness requires that for all $(pk, sk) \in [\mathsf{PKE.Gen}]$ and $m \in \mathcal{M}$, if $c \in [\mathsf{PKE.Enc}_{pk}(m)]$ then $\mathsf{PKE.Dec}_{sk}(c) = m$. For fixed sk we say a ciphertext c is *valid* if $\mathsf{PKE.Dec}_{sk}(c) \neq \bot$.

In the following, PKE will always denote a public-key encryption scheme for plaintext space $\mathcal{M}$ if not otherwise indicated.

We extend the application of encryption to vectors of plaintexts. For $pk \in \mathcal{PK}$, $n \in \mathbb{N}$, $\mathbf{m} \in \mathcal{M}^n$ and $\mathbf{r} \in \mathcal{R}^n$ let $\mathbf{c} = \mathsf{PKE.Enc}_{pk}(\mathbf{m}; \mathbf{r})$ where

$$\mathsf{PKE.Enc}_{pk}(\mathbf{m}; \mathbf{r}) := (\mathsf{PKE.Enc}_{pk}(m_1; r_1), \ldots, \mathsf{PKE.Enc}_{pk}(m_n; r_n)) \ .$$

1.2.1 Standard Security Notions under Passive Attacks

Definition 1.2.2 (IND-CPA/mult-IND-CPA secure PKE). Let $\varepsilon \colon \mathbb{N} \to \mathbb{R}^{\geq 0}$. We say that PKE is (τ, ε)*-mult-IND-CPA secure* if for all τ-time adversaries $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ that interact in the $\mathsf{mult\text{-}IND\text{-}CPA}_b$ experiments as given in Figure 1.1 and for all $n_{m\text{-}cpa} \in \mathbb{N}$ we have

$$\left| \Pr[\mathsf{mult\text{-}IND\text{-}CPA}_0^{\mathcal{A}}(n_{m\text{-}cpa}) \Rightarrow 1] - \Pr[\mathsf{mult\text{-}IND\text{-}CPA}_1^{\mathcal{A}}(n_{m\text{-}cpa}) \Rightarrow 1] \right| \leq \varepsilon(n_{m\text{-}cpa}) \ .$$

Exp $\mathsf{mult\text{-}IND\text{-}CPA}_b^{\mathcal{A}}(n_{m\text{-}cpa})$
$(pk, sk) \leftarrow_\$ \mathsf{PKE.Gen}$
$(\mathbf{m}^0, \mathbf{m}^1, st) \leftarrow_\$ \mathcal{A}_1(pk, n_{m\text{-}cpa})$
$\mathbf{c} \leftarrow_\$ \mathsf{PKE.Enc}_{pk}(\mathbf{m}^b)$
$b' \leftarrow_\$ \mathcal{A}_2(st, \mathbf{c})$
Return b'

Figure 1.1: The $\mathsf{mult\text{-}IND\text{-}CPA}_b$ experiments as used in Definition 1.2.2. We require $\mathcal{A}_1$ to output $\mathbf{m}^0, \mathbf{m}^1$ such that $|\mathbf{m}^0| = |\mathbf{m}^1| = n_{m\text{-}cpa}$.

For $n_{m\text{-}cpa} := 1$ and $\varepsilon := \varepsilon(1)$, PKE is (τ, ε)*-IND-CPA secure* if for all τ-time adversaries that interact in the $\mathsf{mult\text{-}IND\text{-}CPA}$ experiment from Figure 1.1 we have

$$\left| \Pr\left[\mathsf{mult\text{-}IND\text{-}CPA}_0^{\mathcal{A}}(1) \Rightarrow 1\right] - \Pr\left[\mathsf{mult\text{-}IND\text{-}CPA}_1^{\mathcal{A}}(1) \Rightarrow 1\right] \right| \leq \varepsilon \ .$$

In informal discussions we say that a scheme is *IND-CPA* (resp. *mult-IND-CPA*) *secure* if for all efficient adversaries and (for all $n \in \mathbb{N}$, respectively) ε is small. We recap a folklore result showing that IND-CPA security entails mult-IND-CPA security.

Lemma 1.2.3 *Let* PKE *be* $(\tau_{cpa}, \varepsilon_{cpa})$*-IND-CPA secure. Then* PKE *is* $(\tau_{m\text{-}cpa}, \varepsilon_{m\text{-}cpa})$*-mult-IND-CPA secure where*

$$\tau_{m\text{-}cpa} \approx \tau_{cpa} \qquad \varepsilon_{m\text{-}cpa}(n) \leq n_{m\text{-}cpa} \cdot \varepsilon_{cpa} \ .$$

The proof readily follows from a hybrid argument. We refer the reader to [KL07].

Next, we define selective opening security under passive attacks.

1.2.2 Selective Opening Security under Passive Attacks

Definition 1.2.4 (efficiently resampleable distribution). For $n \in \mathbb{N}$, let $2^{[n]}$ denote the power set of $\{1, \ldots, n\}$ and $\mathcal{M}$ be a set. Let $\{\mathfrak{D}_n\}_{n \in \mathbb{N}}$ be a sequence of distributions such that for all $n \in \mathbb{N}$, $\mathfrak{D}_n$ is an efficiently sampleable distribution over $\mathcal{M}^n$. We say $\{\mathfrak{D}_n\}_{n \in \mathbb{N}}$ is *efficiently resampleable* if for all $n \in \mathbb{N}$ there exists an efficient resampling algorithm $\mathsf{Resamp}_{\mathfrak{D}_n} : \mathcal{M}^n \times 2^{[n]} \rightarrow_{\$} \mathcal{M}^n$, such that for all $\mathbf{m} \leftarrow_{\$} \mathfrak{D}$ and $\mathcal{I} \in 2^{[n]}$, $\mathbf{m}' \leftarrow_{\$} \mathsf{Resamp}_{\mathfrak{D}_n}(\mathbf{m}, \mathcal{I})$, $\mathbf{m}'$ is $\mathfrak{D}_n$-distributed conditioned on $\mathbf{m}_{\mathcal{I}} = \mathbf{m}'_{\mathcal{I}}$.

A class $\mathcal{D}$ of sequences of distributions is efficiently resampleable if every sequence in $\mathcal{D}$ is efficiently resampleable.

For the remainder of this work $\{\mathfrak{D}_n\}_{n \in \mathbb{N}}$ denotes a sequence of efficiently resampleable distributions such that for all $n \in \mathbb{N}$, $\mathfrak{D}_n$ is a distribution over $\mathcal{M}^n$ if not indicated otherwise. As $n \in \mathbb{N}$ uniquely specifies a distribution from $\{\mathfrak{D}_n\}_{n \in \mathbb{N}}$ we may write $\mathfrak{D}$ whenever n is fixed. Further, we assume that an efficiently resampleable distribution implicitly comes with an efficient resampling algorithm.

Definition 1.2.5 (IND-SO-CPA secure PKE). Let $\mathcal{D}$ be a subset of the class of sequences of efficiently resampleable distributions. For a function $\varepsilon \colon \mathbb{N} \rightarrow \mathbb{R}^{\geq 0}$ we say that PKE is (τ, ε)-*IND-SO-CPA secure with respect to* $\mathcal{D}$ if for all τ-time adversaries $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3)$ that interact in the $\mathsf{IND\text{-}SO\text{-}CPA}_b$ experiments as given in Figure 1.2 and all $n \in \mathbb{N}$ we have

$$\left| \Pr\left[\mathsf{IND\text{-}SO\text{-}CPA}_0^{\mathcal{A}}(n) \Rightarrow 1\right] - \Pr\left[\mathsf{IND\text{-}SO\text{-}CPA}_1^{\mathcal{A}}(n) \Rightarrow 1\right] \right| \leq \varepsilon(n) \ .$$

If $\mathcal{D}$ is the class of *all* sequences of efficiently resampleable distributions, we say that PKE is (τ, ε)-*IND-SO-CPA secure.* In informal statements we say that a PKE scheme is *IND-SO-CPA secure*, if for all efficient adversaries and all $n \in \mathbb{N}$, ε is small.

Definition 1.2.5 is in the spirit of [BHY09] but we allow for adaptive corruptions and let the adversary choose the distribution, as was done by Böhl *et al.* [BHK12]. “In fact, otherwise it is not even clear that the resulting definition implies IND-CPA security.” [BHK12]

Exp $\mathsf{IND\text{-}SO\text{-}CPA}_b^{\mathcal{A}}(n)$
$\mathcal{I} \leftarrow \emptyset$
$(pk, sk) \leftarrow_\$ \mathsf{PKE.Gen}$
$(\mathfrak{D}, st) \leftarrow_\$ \mathcal{A}_1(pk, n)$
$\mathbf{m}^0 \leftarrow_\$ \mathfrak{D}$
$\mathbf{r} \leftarrow_\$ \mathcal{R}^n$
$\mathbf{c} \leftarrow \mathsf{PKE.Enc}_{pk}(\mathbf{m}^0; \mathbf{r})$
$st' \leftarrow_\$ \mathcal{A}_2^{\text{OPEN}}(st, \mathbf{c})$
$\mathbf{m}^1 \leftarrow_\$ \mathsf{Resamp}_{\mathfrak{D}}(\mathbf{m}^0, \mathcal{I})$
$b' \leftarrow_\$ \mathcal{A}_3(st', \mathbf{m}^b)$
Stop with b'

Oracle OPEN(i)
$\mathcal{I} \leftarrow \mathcal{I} \cup \{i\}$
Return (m_i^0, r_i)

Figure 1.2: Security experiments $\mathsf{IND\text{-}SO\text{-}CPA}_b$ used in Definition 1.2.5. We require $\mathcal{A}_1$ to output $\mathfrak{D}$ such that $\mathfrak{D} \in \mathcal{D}$ and $\mathfrak{D}$ is a distribution over $\mathcal{M}^n$. $\mathcal{A}_2$ may query OPEN(i) for $i \in [n]$.

1.2.3 A Failing Reduction and First Insights

At first sight one might claim that a straight-forward reduction shows that mult-IND-CPA security (and thus IND-CPA security (see Lemma 1.2.3)) already implies IND-SO-CPA security since every party samples fresh randomness independently. Let us try to devise a reduction that constructs a mult-IND-CPA attacker $\mathcal{A}_{m\text{-}cpa}$ from an IND-SO-CPA attacker $\mathcal{A}_{so}$.

Attacker $\mathcal{A}_{m\text{-}cpa}$ will relay pk to $\mathcal{A}_{so}$. Then $\mathcal{A}_{so}$ outputs (in particular) a distribution $\mathfrak{D}$ and expects ciphertexts $\mathbf{c}$. Recall that $\mathcal{A}_{m\text{-}cpa}$ sends two plaintext vectors $\mathbf{m}^0$ and $\mathbf{m}^1$ to its experiment in order to receive an encryption of either $\mathbf{m}^0$ or $\mathbf{m}^1$ as challenge. To use $\mathcal{A}_{so}$ to its advantage, $\mathcal{A}_{m\text{-}cpa}$ shall embed its own challenge in $\mathbf{c}$.

To simulate the IND-SO-CPA experiment for $\mathcal{A}_{so}$ correctly, one of the vectors, say, $\mathbf{m}^0$ shall consist of plaintexts *sampled* from $\mathfrak{D}$, while the other one, $\mathbf{m}^1$, shall consist of *resampled* plaintexts, sampled *after* $\mathcal{A}_{so}$ made its opening queries. However, $\mathcal{A}_{so}$ will only issue opening queries *after* receiving $\mathbf{c}$, while $\mathcal{A}_{m\text{-}cpa}$ —generally[2]— requires knowledge of all opening queries in order to resample $\mathbf{m}^1$ correctly and hence, obtain $\mathbf{c}$.

As the distribution of any plaintext in $\mathbf{m}^1$ may depend on all other plaintexts, the reduction would have to guess all opening queries going to be made by $\mathcal{A}_{so}$. Now $\mathcal{I}$ might be any subset of $\{1, \ldots, n\}$, for instance of size $n/2$ in the worst case. Hence $\mathcal{A}_{m\text{-}cpa}$'s winning probability would be exponentially smaller than $\mathcal{A}_{so}$'s, thus not leading to any meaningful security implications.

[2]This is the very reason why a standard hybrid argument *does* work for independent plaintexts: The resampling becomes sampling and $\mathcal{A}_{m\text{-}cpa}$ does not have to guess opening queries.

The main observation leading to our positive result is as follows: For certain classes of distributions it suffices for $\mathcal{A}_{m\text{-}cpa}$ to *locally* guess the position of only a few opening queries in order to resample (parts of) $\mathbf{m}^1$ correctly. Clearly, guessing the positions of fewer opening queries has a significantly higher probability than guessing the position of all opening queries. Eventually, we employ a hybrid argument and will repeatedly make use of local guessing.

1.3 Results for Graph-Induced Distributions

We continue by fixing the notation for graphs and *graph-induced* distributions.

1.3.1 Graphs and Distributions

Graphs A *directed graph* G consists of a set of vertices V identified with $[n]$ for $n > 0$ and a set of edges $E \subseteq V^2 \setminus \{(v, v) : v \in V\}$, i.e., we do not allow for loops or multiple edges between two vertices. G is *undirected* if $(v_2, v_1) \in E$ for each $(v_1, v_2) \in E$. $\{(v_1, v_2), (v_2, v_1)\} \subseteq E$ is called *undirected edge* between v_1 and v_2. For $V' \subseteq V$ let $G_{V'} := (V', E')$ denote the *induced subgraph* of G where $E' := E \cap V'^2$. For $G = (V, E)$ we obtain its *undirected version*, $G^{\leftrightarrow} = (V, E^{\leftrightarrow})$ where $E^{\leftrightarrow} \supseteq E$ is obtained by adding the minimum number of edges to E so that the graph becomes undirected. For $V' \subseteq V$ let $N(V') := \{v \in V \setminus V' : \exists v' \in V' \textit{ s.t. } (v, v') \in E^{\leftrightarrow}\}$ denote the *(open) neighborhood* of V' in G. For a vertex v, we denote by $\mathrm{P}(v) = \{j : (j, v) \in E\}$ the set of its *parents*.

A *path* from v_1 to v_ℓ in G is a list of at least two vertices $(v_1, \ldots, v_\ell)$ where $v_i \in V$ for $i \in [\ell]$ and $(v_i, v_{i+1}) \in E$ for all $i \in [\ell - 1]$. The *length* of a path $(v_1, \ldots, v_\ell)$ is $\ell - 1$. For two distinct vertices u, v the *distance* $d(u, v)$ between u and v is given by the length of a shortest path between u und v. If there is a path from u to v then u is a *predecessor* of v. Let $\mathrm{pred}(v)$ denote the set of all predecessors of v. A *cycle* is a path where $v_\ell = v_1$. G is *acyclic* if it contains no cycle.

A subset $V' \subseteq V$ is *connected* in G if for every pair of distinct vertices $(v_1, v_2) \in V'$ there exists a path from v_1 to v_2 in $G^{\leftrightarrow}$. If V' is connected we call it *connected subgraph*. *G is connected* if V is connected in G. Graph G is *disconnected* if G is not connected. We assume G to be connected if not stated otherwise. A (set-)maximal connected set of vertices of G is called *connected component*.

In the following $\{G_n\}_{n \in \mathbb{N}}$ will denote a sequence of directed, acyclic graphs such that for all $n \in \mathbb{N}$, G_n is a graph on n vertices if not indicated otherwise.

Notational convention For the sake of notational brevity, we use the same notation for both the i^{th} plaintext of an n-plaintext vector and vertex i in a graph on n vertices.

We now define Markov distributions, which are distributions on vectors of random variables reflecting processes. That is to say variables with higher indices depend on preceding variables. A distribution is Markov if it is *memoryless* in the sense that all relevant information for the distribution of m_i is already present in m_{i-1}, although the latter itself may depend on its predecessor.

Definition 1.3.1 For $n \in \mathbb{N}$ let $\mathbf{M} = (M_1, \ldots, M_n)$ denote a vector of $\mathcal{M}$-valued random variables. We say $\{\mathfrak{D}_n\}_{n\in\mathbb{N}}$ is *Markov* if the following holds for all $n \in \mathbb{N}$ and all $\mathbf{m} \in \mathcal{M}^n$:

$$\Pr_{\mathbf{M}\leftarrow\mathfrak{D}}\left[M_j = m_j \,\middle|\, \bigwedge_{i=1}^{j-1} M_i = m_i \right] = \Pr_{\mathbf{M}\leftarrow\mathfrak{D}}\left[M_i = m_i \,\middle|\, M_{i-1} = m_{i-1} \right] .$$

We note that Markov distributions can be seen as 'induced' by a chain graph $M_1 \to M_2 \to \ldots \to M_n$, where for all $j \in [n]$ the distributions of any M_j given its predecessors, or solely M_{j-1}, respectively, are identical.

We will now generalize this to arbitrary graphs and still require (a generalization of) 'memorylessness'. We say that a graph G induces a distribution $\mathfrak{D}$ if whenever the distribution of M_j depends on M_i, then there is a path from i to j in $G^{\leftrightarrow}$. As for Markov distributions, we require that the information about the distribution of a plaintext is present in its parents; in particular, for all $j \in [n]$ and $\mathbf{M} = (M_1, \ldots, M_n) \leftarrow \mathfrak{D}$ the distribution of M_j shall only depend on its parents in G, i.e., the set $\mathrm{P}(j)$, rather than all its predecessors $\mathrm{pred}(j)$.

Definition 1.3.2 (graph-induced distribution). We say that $\{\mathfrak{D}_n\}_{n\in\mathbb{N}}$ is $\{G_n\}_{n\in\mathbb{N}}$-*induced* if the following holds for all $n \in \mathbb{N}$:

- For all $i, j \in [n]$, $i \neq j$: If for $\mathfrak{D}_n$ the distribution of M_j depends on M_i then there is a path from i to j in $G_n^{\leftrightarrow}$.

- For all $j \in [n]$ and all $\mathbf{m} \in \mathcal{M}^n$ we have

$$\Pr_{\mathbf{M}\leftarrow\mathfrak{D}}\left[M_j = m_j \,\middle|\, \bigwedge_{i\in\mathrm{pred}(j)} M_i = m_i \right] = \Pr_{\mathbf{M}\leftarrow\mathfrak{D}}\left[M_j = m_j \,\middle|\, \bigwedge_{i\in\mathrm{P}(j)} M_i = m_i \right] .$$

We assume that for any $n \in \mathbb{N}$ one can efficiently reconstruct a graph G_n with the above properties given $\mathfrak{D}_n$.

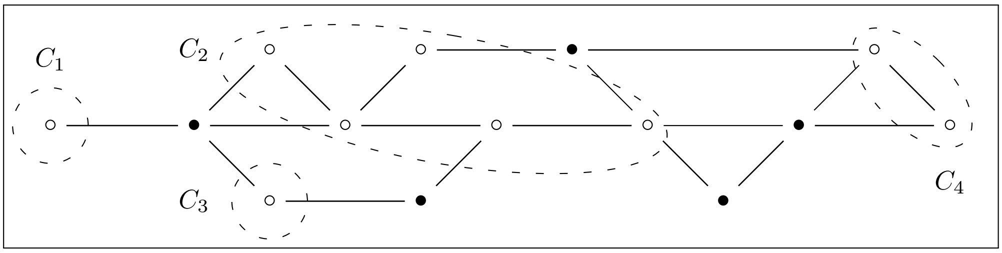

Figure 1.3: Example graph. Vertices are given by • and ◦. Set $\mathcal{I}$ shall contain all vertices marked with •. Connected components C_i, are enclosed in dashed lines.

As with a family of distributions, we say that $\mathfrak{D}$ *is* G*-induced* whenever n is already fixed.

Although our proof ideas can be applied to disconnected graphs directly, Sections 1.3.3−1.3.4 consider *connected graphs* for simplicity. A hybrid argument over the connected components of a graph as given in Section 1.4 extends all our results to disconnected graphs.

Our Approach in Terms of Graphs As a warm-up we sketch our novel approach in terms of graphs. For fixed n let G be a graph inducing a distribution over $\mathcal{M}^n$. Fix any subset $\mathcal{I} \subseteq [n]$ of opening queries made by some adversary.

The main observation is that removing $\mathcal{I}$ and all incident edges, G decomposes into connected components $C_1, \ldots, C_{n'}$. These can be resampled independently as to resample from the distribution on C_k it suffices to know the *neighborhood* of C_k and $\mathfrak{D}$. See Figure 1.3 for a toy example.

To argue that there is no efficient adversary $\mathcal{A}_{so}$ that distinguishes sampled and resampled plaintexts in the selective opening experiment, we proceed in a sequence of hybrid experiments. We start in an experiment where after receiving encryptions of sampled plaintexts and replies to opening queries, $\mathcal{A}_{so}$ obtains sampled plaintexts. In the k^{th} hybrid step we use mult-IND-CPA security to replace *sampled* plaintexts on a connected component C_k with *resampled* plaintexts without $\mathcal{A}_{so}$ noticing. To this end, the reduction from the indistinguishability of two consecutive hybrids to mult-IND-CPA has to identify, i.e. guess, C_k to embed its own challenge before $\mathcal{A}_{so}$ makes any opening query.

Note that there are multiple approaches if one wishes to specify a connected component C_k of $G_{\overline{\mathcal{I}}}$ in G. For instance, 1) clearly, C_k can be represented by the vertices contained in it. 2) Secondly, one may try to identify C_k by its neighborhood in G. For instance, if G is a chain graph and for any $\mathcal{I} \subseteq [n]$ it suffices to give *two* vertices to characterize a connected component C_k of $G_{\overline{\mathcal{I}}}$, for instance the neighborhood $N(C_k)$

of C_k. Thus, in each hybrid step, a reduction would merely lose a factor of $\mathcal{O}(n^2)$ to (implicitly) guess C_k.

More generally, we are interested in all graph structures that allow a reduction to identify some C_k with an at most polynomial (in n) loss in each hybrid step. We provide definitions to formally capture the two approaches sketched above.

Definition 1.3.3 (maximum border). Let $G = (V, E)$ be a graph. We define the *maximum border* of G as the maximal size of the neighborhood of any connected subgraph, taken over all connected subsets of V.

$$B(G) := \max_{V' \subseteq V} \{|N(V')| : G_{V'} \text{ is connected}\} \quad .$$

For example, if G is an n-path for $n \geq 3$ then $B(G) = 2$. For the complete graph or star graph on n vertices we have $B(G) = n - 1$. Clearly, $B(G) < n$ for any graph.

Definition 1.3.4 (number of connected subgraphs). Let $G = (V, E)$. We define the *number of connected subgraphs* of G:

$$S(G) := |\{V' \subseteq V : G_{V'} \text{ connected}\}| \quad .$$

For example, a chain graph on n vertices has $\frac{1}{2} \cdot n \cdot (n + 1)$ connected subsets of its vertices while for the complete graph C_n on n vertices we have $S(C_n) = 2^n - 1$.

Both approaches, graph sequences $\{G_n\}_{n \in \mathbb{N}}$ where $S(G) = \mathsf{poly}(n)$ or where $B(G) = \mathsf{const}$, seem promising. However, a connected component might not be uniquely specified by its neighborhood. For instance, consider a graph on n vertices, consisting of a chordless circle of n vertices. Clearly, its maximum border is of size 2. Removing any two non-adjacent vertices, the graph decomposes into two connected components, whereby both of them had the *same* neighborhood. Hence, if a reduction aims at identifying a connected component C_k in $G_{\overline{\mathcal{I}}}$ it might have to guess the 'correct' subgraph, too.

1.3.2 Relating the Maximum Border and Number of Connected Subgraphs

We continue with the following theorem bounding the number of connected subgraphs in terms of the maximum border of a graph.

Theorem 1.3.5 *Let G be a connected graph. Then the following bound on $S(G)$ holds:*

$$S(G) \leq \frac{2}{(B(G)-1)!} \cdot n^{B(G)} \quad \textit{for all} \quad 0 < B(G) \leq \frac{n-2}{3} \ .$$

In particular for a sequence of graphs $\{G_n\}_{n\in\mathbb{N}}$, for all $n \in \mathbb{N}$, $B(G_n) = \mathsf{const}$ implies $S(G_n) = \mathsf{poly}(n)$.

Before proving the theorem, we give an easy lemma: If two connected subsets of the vertices of G share the same neighborhood and are distinct, they have to be disjoint. We will apply Lemma 1.3.6 in the proof of Theorem 1.3.5 to obtain an upper bound on the number of connected subsets in a graph that share the same neighborhood.

Lemma 1.3.6 *Let $G = (V, E)$ and $V_1, V_2 \subseteq V$, such that $V_1 \neq V_2$ each of them connected in G, such that $N(V_1) = N(V_2)$. Then $V_1 \cap V_2 = \emptyset$.*

Proof of Lemma 1.3.6. Assume $V_1 \cap V_2 \neq \emptyset$. As $V_1 \neq V_2$ we have $V_1 \setminus V_2 \neq \emptyset$ without loss of generality. Because V_1 is connected, there exist vertices $v_\cap \in V_1 \cap V_2$ and $v_1 \in V_1 \setminus V_2$ such that $(v_1, v_\cap) \in E^{\leftrightarrow}$. Since $v_1 \notin V_2$, $v_\cap \in V_2$ and $(v_1, v_\cap) \in E^{\leftrightarrow}$, we see that $v_1 \in N(V_2)$. As $N(V_2) = N(V_1)$ it follows that $v_1 \in N(V_1)$; a contradiction since $v_1 \in V_1$. ■

Proof of Theorem 1.3.5. Let $B := B(G)$. We have

$$\begin{aligned} S(G) &= |\{V' \subseteq V : G_{V'} \text{ connected}\}| \\ &= \sum_{i=0}^{B} |\{V' \subseteq V : G_{V'} \text{ connected} \wedge |N(V')| = i\}| \ . \end{aligned}$$

For $i = 0$ we count the connected components of G. Since G is connected it follows

$$\begin{aligned} S(G) &= 1 + \sum_{i=1}^{B} \left|\{V' \subseteq V : G_{V'} \text{ connected} \wedge |N(V')| = i\}\right| \\ &= 1 + \sum_{i=1}^{B} \sum_{\substack{V_i \subseteq V \\ |V_i| = i}} \left|\{V' \subseteq V : G_{V'} \text{ connected} \wedge N(V') = V_i\}\right| \ . \end{aligned}$$

Let $V_i \subseteq V$ be non-empty and $\{V'_1, \ldots, V'_k\} := \{V' \subseteq V : G_{V'} \text{ connected} \wedge N(V') = V_i\}$ for appropriate k. By applying Lemma 1.3.6 to $V'_1, \ldots, V'_k$, we see that those sets V'_j are pairwise disjoint. Fix any vertex $v_i \in V_i$. Since $N(V'_j) = V_i$ for $j \in [k]$ and all V'_j are pairwise disjoint, there exists at least one vertex v'_j in each V'_j such that $(v'_j, v_i) \in E$ for all $j \in [k]$. Thus, $N(v_i) \geq k$, i.e. $B \geq k$. Hence, for given B and we obtain an upper bound for the number of possible sets V' for each fixed V_i. It follows

$$S(G) \leq 1 + \sum_{i=1}^{B} \sum_{\substack{V_i \subseteq V \\ |V_i|=i}} B = 1 + B \cdot \sum_{i=1}^{B} \binom{n}{i} \leq B \cdot \sum_{i=0}^{B} \binom{n}{i} \quad . \tag{1.1}$$

To bound the sum in (1.1) we use the geometric series and upper-bound the quotient of two consecutive binomial coefficients by $\frac{1}{2}$:

$$\frac{\binom{n}{i}}{\binom{n}{i+1}} = \frac{i+1}{n-i} \leq \frac{1}{2} \Leftrightarrow i \leq \frac{n-2}{3} \quad .$$

Hence

$$B \cdot \sum_{i=0}^{B} \binom{n}{i} \leq B \cdot \sum_{i=0}^{B} \frac{1}{2^i} \binom{n}{B} \leq B \cdot \binom{n}{B} \cdot \sum_{i=0}^{\infty} \frac{1}{2^i} \leq 2 \cdot B \cdot \frac{n^B}{B!} = \frac{2}{(B-1)!} \cdot n^B$$

for $B(G) \leq \frac{n-2}{3}$, which concludes the proof. ■

The Quality of the Bound from Theorem 1.3.5 Even though some of the approximations in the proof of Theorem 1.3.5 appear rather rough, there are graph sequences where the established bound is asymptotically tight. To this end, let $\{G_n\}_{n\in\mathbb{N}}$ be the sequence of chain graphs. We have $S(G_n) = n \cdot (n+1)/2$ while we recall that $B(G_n) = 2$ for all n and obtain an upper bound of the form $S(G_n) \leq 2 \cdot n^2$ via Theorem 1.3.5.

Further, the bound of Theorem 1.3.5 only holds for $n \geq 3B(G) + 2$. However, as we establish results for sequences of graphs $\{G_n\}_{n\in\mathbb{N}}$ where the maximum border $B(G_n)$ is constant for all n, there are only finitely many elements in the sequence where the bound does not apply. If need be, one can easily obtain a bound similarly to Theorem 1.3.5 that is weaker by a factor of roughly $B(G)$ but holds for all $B(G) < n$. To this end, one bounds the sum of binomial coefficients in (1.1) in terms of the incomplete upper gamma function Γ to obtain

$$\sum_{i=1}^{B} \binom{n}{i} \leq \sum_{i=1}^{B} \frac{n^i}{i!} = \frac{e^n \cdot \Gamma(B+1, n)}{B!} - 1 \quad .$$

Using a nice bound [NP00] on Γ we obtain a bound for $B(G) < n$.

1.3.3 Main Result for Graph-Induced Distributions

We state our main results relating IND-CPA and IND-SO-CPA security for certain distributions.

Theorem 1.3.7 *Let $\mathcal{D}$ be the class of efficiently resampleable sequences of distributions induced by sequences of connected graphs $\{G_n\}_{n\in\mathbb{N}}$.*

If PKE *is $(\tau_{cpa}, \varepsilon_{cpa})$-IND-CPA secure, then* PKE *is $(\tau_{so}, \varepsilon_{so})$-IND-SO-CPA secure where*

$$\tau_{so} \le \tau_{cpa} - 2 \cdot \tau_{resamp} \qquad \varepsilon_{so}(n) \le n \cdot (n-1) \cdot S(G_n) \cdot \varepsilon_{cpa}$$

where τ_{resamp} is the time of one execution of the resampling algorithm.

Note that, in particular, we have an (at most) polynomial loss in n if for all $n \in \mathbb{N}$ we have $S(G_n) = \mathsf{poly}(n)$.

Proof Sketch Recall the $\mathsf{IND\text{-}SO\text{-}CPA}_b$ experiment given in Figure 1.2. As a challenge the experiment sends $\mathbf{m}^b$, where $\mathbf{m}^0_{\overline{\mathcal{I}}}$ consists of plaintexts sampled at the beginning, while $\mathbf{m}^1_{\overline{\mathcal{I}}}$ is resampled (conditioned on $\mathbf{m}^1_{\mathcal{I}} = \mathbf{m}^0_{\mathcal{I}}$). We define hybrid experiments $\mathsf{H}_0, \mathsf{H}_1, \ldots, \mathsf{H}_n$. For this, let $\mathcal{S} \subseteq 2^V$ denote the set of all connected subsets of vertices of G. We have $|\mathcal{S}| = S(G)$.

Note that the vertices in $G_{\overline{\mathcal{I}}}$ consist of connected vertex sets $C_1, \ldots, C_{n'} \subseteq \mathcal{S}$ for some $n' \le n-1$. (This upper bound is attained by the star graph when $\mathcal{I}$ consists of the internal vertex.) We assume those components to be ordered, e.g., by the smallest vertex contained in each.

Thus, if $b = 1$ the IND-SO-CPA experiment can resample $\mathbf{m}^1_{\overline{\mathcal{I}}}$ in n' batches $\mathbf{m}^1_{C_1}, \ldots, \mathbf{m}^1_{C_{n'}}$ (as $\overline{\mathcal{I}} = \bigcup_{i=1}^{n'} C_i$). Moreover, each batch $\mathbf{m}^1_{C_i}$ can be resampled *independently*, i.e., as a function of $\mathbf{m}^0_{\mathcal{I}}$ and $\mathfrak{D}$, but not $\mathbf{m}^1_{C_j}$, $j \ne i$.

Proof of Theorem 1.3.7. Let $\mathcal{A}_{so} = (\mathcal{A}_{so,1}, \mathcal{A}_{so,2}, \mathcal{A}_{so,3})$ be an adversary that breaks the $(\tau_{so}, \varepsilon_{so})$-IND-SO-CPA security of PKE for some $n \in \mathbb{N}$. We define hybrid experiments H_k as a modification of the $\mathsf{IND\text{-}SO\text{-}CPA}_b$ experiment, in which the plaintexts of the first k batches $C_1, \ldots, C_k$ are resampled while the remaining batches stay sampled (Figure 1.4).

To this end line 09 is added and $\mathcal{A}_{so,3}$ is invoked on a partially sampled, partially resampled hybrid vector in line 10. Besides, the experiment remains as in Definition 1.2.5.

Clearly, H_0 is the (real) $\mathsf{IND\text{-}SO\text{-}CPA}_0$ experiment and $\mathsf{H}_{n'}$ for some $n' \le n-1$ is the (random) experiment $\mathsf{IND\text{-}SO\text{-}CPA}_1$. Note that for $j, k \in [n', n]$ hybrids H_j and H_k are identical. We have

Exp $\mathsf{H}_k^{\mathcal{A}_{so}}(n)$

01 $\mathcal{I} \leftarrow \emptyset$
02 $(pk, sk) \leftarrow_{\$} \mathsf{PKE.Gen}$
03 $(\mathfrak{D}, st_1) \leftarrow_{\$} \mathcal{A}_{so,1}(pk, n)$
04 $\mathbf{m}^0 \leftarrow_{\$} \mathfrak{D}$
05 $\mathbf{r} \leftarrow_{\$} \mathcal{R}^n$
06 $\mathbf{c} \leftarrow \mathsf{PKE.Enc}_{pk}(\mathbf{m}^0; \mathbf{r})$
07 $st_2 \leftarrow_{\$} \mathcal{A}_{so,2}^{\textsc{Open}}(st_1, \mathbf{c})$
08 $\mathbf{m}^1 \leftarrow_{\$} \mathsf{Resamp}_{\mathfrak{D}}(\mathbf{m}^0, \mathcal{I})$
09 $m_i \leftarrow \begin{cases} m_i^1 & \text{for } i \in \bigcup_{j=1}^{k} C_j \\ m_i^0 & \text{else} \end{cases}$
10 $b' \leftarrow_{\$} \mathcal{A}_{so,3}(st_2, \mathbf{m})$
11 Stop with b'

Oracle $\textsc{Open}(i)$

12 $\mathcal{I} \leftarrow \mathcal{I} \cup \{i\}$
13 Return (m_i^0, r_i)

Figure 1.4: Hybrid experiments $\mathsf{H}_k(n)$ used in the proof of Theorem 1.3.7. Line 09 was added to assemble the hybrid challenge vector given to $\mathcal{A}_3$ in line 10. C_i denotes the i^{th} component in $G_{\overline{\mathcal{I}}}$.

$$\begin{aligned}\Big|\Pr\left[\mathsf{IND\text{-}SO\text{-}CPA}_0^{\mathcal{A}_{so}}(n) \Rightarrow 1\right] &- \Pr\left[\mathsf{IND\text{-}SO\text{-}CPA}_1^{\mathcal{A}_{so}}(n) \Rightarrow 1\right]\Big| \\ &= \Big|\Pr\left[\mathsf{H}_0^{\mathcal{A}_{so}}(n) \Rightarrow 1\right] - \Pr\left[\mathsf{H}_{n'}^{\mathcal{A}_{so}}(n) \Rightarrow 1\right]\Big| \\ &\leq \sum_{k=0}^{n'-1} \Big|\Pr\left[\mathsf{H}_k^{\mathcal{A}_{so}}(n) \Rightarrow 1\right] - \Pr\left[\mathsf{H}_{k+1}^{\mathcal{A}_{so}}(n) \Rightarrow 1\right]\Big| \quad .\end{aligned}$$

We now upper-bound the distance between two consecutive hybrids with the following lemma.

Lemma 1.3.8 *There exists* $\mathcal{A}_{m\text{-}cpa} = (\mathcal{A}_{m\text{-}cpa,1}, \mathcal{A}_{m\text{-}cpa,2})$ *and* $n_{m\text{-}cpa} \in \mathbb{N}$ *such that* $\mathcal{A}_{m\text{-}cpa}$ *breaks the* $(\tau_{m\text{-}cpa}, \varepsilon_{m\text{-}cpa})$*-mult-IND-CPA security for* $n_{m\text{-}cpa}$ *of* PKE *where*

$$\tau_{m\text{-}cpa} \approx \tau_{so} + 2\cdot\tau_{resamp} \;\;,\;\; \varepsilon_{m\text{-}cpa} \geq \frac{1}{S(G_n)} \cdot \Big|\Pr\left[\mathsf{H}_k^{\mathcal{A}_{so}}(n) \Rightarrow 1\right] - \Pr\left[\mathsf{H}_{k+1}^{\mathcal{A}_{so}}(n) \Rightarrow 1\right]\Big| \;\;,$$

and τ_{resamp} *is the time of one execution of the resampling algorithm.*

Proof of Lemma 1.3.8. We construct adversary $\mathcal{A}_{m\text{-}cpa}$ as follows (see Figure 1.5). The value of $n_{m\text{-}cpa} \in \mathbb{N}$ follows from the proof.

$\mathcal{A}_{m\text{-}cpa,1}$ sends (pk, n) to $\mathcal{A}_{so}$ and picks $C_{k+1}^* \leftarrow_{\$} \mathcal{S}$ uniformly at random (trying to guess C_{k+1}) after receiving $(\mathfrak{D}, \mathsf{Resamp}_{\mathfrak{D}})$ (lines 01, 02). $\mathcal{A}_{m\text{-}cpa,1}$ samples $\mathbf{m}^0 \leftarrow \mathfrak{D}$ and resamples $\mathbf{m}^1$ conditioned on the neighborhood of C_{k+1}^* (lines 03, 04). It submits $(\mathbf{m}^0_{C_{k+1}^*}, \mathbf{m}^1_{C_{k+1}^*})$ to its mult-IND-CPA challenger and terminates in line 05. Then

Adversary $\mathcal{A}_{m\text{-}cpa,1}(pk, n_{m\text{-}cpa})$

01 $(\mathfrak{D}) \leftarrow_{\$} \mathcal{A}_{so,1}(pk, n)$
02 $C^*_{k+1} \leftarrow_{\$} \mathcal{S}$
03 $\mathbf{m}^0 \leftarrow_{\$} \mathfrak{D}$
04 $\mathbf{m}^1 \leftarrow \mathsf{Resamp}_{\mathfrak{D}}(\mathbf{m}^0, N(C^*_{k+1}))$
05 Output $(\mathbf{m}^0_{C^*_{k+1}}, \mathbf{m}^1_{C^*_{k+1}})$

Adversary $\mathcal{A}_{m\text{-}cpa,2}(\mathbf{c}_{C^*_{k+1}})$

06 $\mathbf{r} \leftarrow_{\$} \mathcal{R}^n$
07 For $i \leftarrow 1$ to n:
08 $\quad c_i \leftarrow \begin{cases} c_i & \text{for } i \in C^*_{k+1} \\ \mathsf{PKE.Enc}_{pk}(m^0_i; r_i) & \text{else} \end{cases}$
09 $\mathbf{c} \leftarrow (c_1, \ldots, c_n)$
10 $\mathcal{I} \leftarrow \emptyset$
11 $() \leftarrow_{\$} \mathcal{A}^{\textsc{Open}}_{so,2}(\mathbf{c})$
12 If $C^*_{k+1} \neq C_{k+1}$: Abort
13 $\widetilde{\mathbf{m}}^1 \leftarrow_{\$} \mathsf{Resamp}_{\mathfrak{D}}(\mathbf{m}^0, \mathcal{I})$
14 $m_i \leftarrow \begin{cases} \widetilde{m}^1_i & \text{for } i \in \bigcup_{j=1}^{k} C_j \\ m^0_i & \text{else} \end{cases}$
15 $\mathbf{m} \leftarrow (m_1, \ldots, m_n)$
16 $b' \leftarrow_{\$} \mathcal{A}_{so,3}(\mathbf{m})$
17 Output b'

Oracle $\textsc{Open}(i)$

18 If $i \in C^*_{k+1}$: Abort
19 $\mathcal{I} \leftarrow \mathcal{I} \cup \{i\}$
20 Return (m^0_i, r_i)

Figure 1.5: Pseudocode of adversary $\mathcal{A}_{m\text{-}cpa} = (\mathcal{A}_{m\text{-}cpa,1}, \mathcal{A}_{m\text{-}cpa,2})$. $\mathcal{A}_{m\text{-}cpa}$ interpolates between hybrids H_k, H_{k+1} for $\mathcal{A}_{so}$. For clarity we abstain from making the states output by and returned to $\mathcal{A}_{so}$ and $\mathcal{A}_{m\text{-}cpa}$ explicit.

$\mathcal{A}_{m\text{-}cpa,2}$ is started on ciphertexts for positions in C^*_{k+1}, picks fresh randomness and encrypts each plaintext in $\overline{C^*_{k+1}}$ (lines 06 – 08). Then it starts $\mathcal{A}_{so,2}$ on $(\mathbf{c}_1, \ldots, \mathbf{c}_n)$, embedding its challenge at positions C^*_{k+1} (see line 08 again) and answers opening queries honestly if they do not occur on C^*_{k+1}. If $\mathcal{A}_{so,2}$ issues such a query, $\mathcal{A}_{m\text{-}cpa,2}$ cannot answer and aborts (line 18). Once $\mathcal{A}_{so,2}$ terminates, $\mathcal{A}_{m\text{-}cpa,2}$ verifies that it guessed C_{k+1} correctly and aborts if it failed to do so (line 12).

$\mathcal{A}_{m\text{-}cpa,2}$ resamples plaintexts $\widetilde{\mathbf{m}}^1$ conditioned on all plaintexts from opened ciphertexts (see line 13). These are then sent in the first k batches while plaintexts from $\mathbf{m}^0$ are sent in every other position (see lines 14, 15). $\mathcal{A}_{m\text{-}cpa,2}$ relays $\mathcal{A}_{so,3}$'s output to its mult-IND-CPA challenger (line 16).

As $\mathcal{A}_{m\text{-}cpa}$ submits vectors of length $|C^*_{k+1}|$ to its mult-IND-CPA experiment, we choose $n_{m\text{-}cpa} := |C^*_{k+1}|$.

Analysis In the following we write $\mathbf{m} \equiv \mathbf{m}'$ if $\mathbf{m}$ and $\mathbf{m}'$, interpreted as random variables, are identically distributed where the probability is taken over all choices in the computation of $\mathbf{m}$ and $\mathbf{m}'$.

Assume, $\mathcal{A}_{m\text{-}cpa}$ guessed correctly, i.e. $C^*_{k+1} = C_{k+1}$, then $\mathcal{A}_{m\text{-}cpa}$ perfectly simulates hybrids H_k and H_{k+1} for plaintexts and ciphertexts at positions in $\overline{C_{k+1}}$. Further, run in $\mathsf{mult\text{-}IND\text{-}CPA}_0$, $\mathcal{A}_{m\text{-}cpa}$ obtains $\mathsf{PKE.Enc}_{pk}(\mathbf{m}^0_{C_{k+1}})$. Hence, $\mathcal{A}_{so}$ receives encryptions of sampled plaintexts. As for $\mathcal{A}_{so}$'s challenge, the $(k+1)^{th}$ batch contains sampled plaintexts $\mathbf{m}^0_{C_{k+1}}$, thus $\mathcal{B}_{\mathsf{mult}}$ perfectly simulates hybrid H_k.

When $\mathcal{A}_{m\text{-}cpa}$ is run in the $\mathsf{mult\text{-}IND\text{-}CPA}_1$ experiment, $\mathcal{A}_{so}$ obtains encryptions of *resampled* plaintexts $\mathsf{PKE.Enc}_{pk}(\mathbf{m}^1_{C_{k+1}})$ while it expects encrypted *sampled* plaintexts: $\mathsf{PKE.Enc}_{pk}(\mathbf{m}^0_{C_{k+1}})$. As a challenge $\mathcal{A}_{so}$ expects *resampled* plaintexts $\widetilde{\mathbf{m}}^1_{C_{k+1}}$ but obtains *sampled* $\mathbf{m}^0_{C_{k+1}}$. Thus, the *sampled* and *resampled* plaintexts change roles on positions C_{k+1}. However, they are equally distributed, i.e., $\mathbf{m}^0_{C_{k+1}} \equiv \mathbf{m}^1_{C_{k+1}}$ since $N(C_{k+1})$ was fixed when resampling $\mathbf{m}^1$ and the distribution of C_{k+1} depends on $\mathfrak{D}$ and plaintexts in positions $N(C_{k+1})$ only. Likewise, $\mathbf{m}^1_{C_{k+1}} \equiv \widetilde{\mathbf{m}}^1_{C_{k+1}}$ for $\mathbf{m}^1 \leftarrow \mathsf{Resamp}_{\mathfrak{D}}(\mathbf{m}^0, N(C_{k+1}))$ and $\widetilde{\mathbf{m}}^1 \leftarrow \mathsf{Resamp}_{\mathfrak{D}}(\mathbf{m}^0, \mathcal{I})$ since the distribution of plaintexts in C_{k+1} solely depends on $\mathfrak{D}$ and plaintexts in $N(C_{k+1}) \subseteq \mathcal{I}$.[3] Thus, $\mathcal{A}_{so}$'s view is identical to its view in hybrid H_{k+1}. Let Abort denote the event that $\mathcal{A}_{m\text{-}cpa}$ aborts during its execution. We have

$$\begin{aligned} &\Pr\left[\mathsf{mult\text{-}IND\text{-}CPA}_0^{\mathcal{A}_{m\text{-}cpa}} \Rightarrow 1\right] = \Pr\left[\mathsf{H}_k^{\mathcal{A}_{so}} \Rightarrow 1 \wedge \overline{\textsc{Abort}}\right] \\ \text{and} \quad &\Pr\left[\mathsf{mult\text{-}IND\text{-}CPA}_1^{\mathcal{A}_{m\text{-}cpa}} \Rightarrow 1\right] = \Pr\left[\mathsf{H}_{k+1}^{\mathcal{A}_{so}} \Rightarrow 1 \wedge \overline{\textsc{Abort}}\right] \quad . \end{aligned}$$

Observe that Abort does not happen iff $\mathcal{A}_{m\text{-}cpa}$ guessed C_{k+1} correctly. Since $\overline{\textsc{Abort}}$ is independent of $\mathcal{A}_{so}$'s output in a hybrid and $|\mathcal{S}| = S(G)$, we have

$$\varepsilon_{m\text{-}cpa} \geq \frac{1}{S(G_n)} \cdot \left| \Pr\left[\mathsf{H}_k^{\mathcal{A}_{so}}(n) \Rightarrow 1\right] - \Pr\left[\mathsf{H}_{k+1}^{\mathcal{A}_{so}}(n) \Rightarrow 1\right] \right| \quad .$$

One easily verifies that $\mathcal{A}_{m\text{-}cpa}$'s running time is essentially the running time of $\mathcal{A}_{so}$ except for two invocations of Resamp performed by $\mathcal{A}_{m\text{-}cpa}$. ■

We proceed with the proof of Theorem 1.3.7. Using Lemma 1.3.8 we have

[3] Note that further $C_{k+1} \cap \mathcal{I} = \emptyset$ as we assumed that $\mathcal{A}_{m\text{-}cpa}$ guessed C_{k+1} correctly.

$$\begin{aligned}\varepsilon_{so}(n) &= \left|\Pr\left[\mathsf{H}_0^{\mathcal{A}_{so}}(n) \Rightarrow 1\right] - \Pr\left[\mathsf{H}_{n'}^{\mathcal{A}_{so}}(n) \Rightarrow 1\right]\right| \\ &\leq \sum_{k=0}^{n'-1} \left|\Pr\left[\mathsf{H}_k^{\mathcal{A}_{so}}(n) \Rightarrow 1\right] - \Pr\left[\mathsf{H}_{k+1}^{\mathcal{A}_{so}}(n) \Rightarrow 1\right]\right| \\ &\leq \sum_{k=0}^{n'-1} S(G_n) \cdot \varepsilon_{m\text{-}cpa} \ .\end{aligned}$$

Now observe that $\mathcal{A}_{m\text{-}cpa}$ sends vectors of length $|C_{k+1}^*| = n_{m\text{-}cpa}$ to its mult-IND-CPA challenger. Eventually, we reduce the mult-IND-CPA security of PKE to its IND-CPA security (see Lemma 1.2.3).

$$\sum_{k=0}^{n'-1} S(G_n) \cdot \varepsilon_{m\text{-}cpa} \overset{(L.\ 1.2.3)}{\leq} \sum_{k=0}^{n'-1} S(G_n) \cdot n_{m\text{-}cpa} \cdot \varepsilon_{cpa} \overset{n_{m\text{-}cpa} \leq n}{\leq} n \cdot (n-1) \cdot S(G_n) \cdot \varepsilon_{cpa}$$

for an $(\tau_{cpa}, \varepsilon_{cpa})$-IND-CPA attacker which completes the proof of Theorem 1.3.7. ■

Markov Distributions. Markov distributions (Definition 1.3.1) are induced by the chain graph $(V = [n], E = \{(i, i+1) : i \in [n-1]\})$, for which $S(G) = \frac{1}{2} \cdot n \cdot (n+1)$.

Corollary 1.3.9 *If* PKE *is* $(\tau_{cpa}, \varepsilon_{cpa})$*-IND-CPA secure, then* PKE *is* $(\tau_{so}, \varepsilon_{so})$*-IND-SO-CPA secure w.r.t efficiently resampleable Markov distributions where*

$$\tau_{so} \leq \tau_{cpa} - 2 \cdot \tau_{resamp} \ , \qquad \varepsilon_{so}(n) \leq \frac{1}{2} \cdot n^2 \cdot (n-1)^2 \cdot \varepsilon_{cpa}$$

and τ_{resamp} *is the time of one execution of the resampling algorithm.*

The proof follows from Theorem 1.3.7.

Theorems 1.3.7 and 1.3.5 together now yield the following corollary.

Corollary 1.3.10 *Let* $\mathcal{D}$ *be the class of efficiently resampleable sequences of distributions induced by sequences of connected graphs* $\{G_n\}_{n\in\mathbb{N}}$*.*

If PKE *is* $(\tau_{cpa}, \varepsilon_{cpa})$*-IND-CPA secure, then* PKE *is* $(\tau_{so}, \varepsilon_{so})$*-IND-SO-CPA secure where*

$$\tau_{so} \leq \tau_{cpa} - 2 \cdot \tau_{resamp} \ , \qquad \varepsilon_{so}(n) \leq \frac{2 \cdot (n-1)}{(B(G_n) - 1)!} \cdot n^{B(G_n)+1} \cdot \varepsilon_{cpa}$$

where τ_{resamp} *is the time of one execution of the resampling algorithm.*

In particular, we obtain obtain an (at most) polynomial loss in n if $\{G_n\}_{n\in\mathbb{N}}$ is such that $B(G_n) = \mathsf{const}$ for all $n \in \mathbb{N}$.

To sum up, we showed that IND-CPA security implies IND-SO-CPA security for efficiently resampleable and $\{G_n\}_{n\in\mathbb{N}}$-induced distributions where $B(G_n) = \mathsf{const}$ or $S(G_n) = \mathsf{poly}(n)$. However, Corollary 1.3.10 cannot cover a larger class of graphs than Theorem 1.3.7, as $B(G_n) = \mathsf{const}$ implies $S(G_n) = \mathsf{poly}(n)$ (see Theorem 1.3.5). Actually, it is easy to see that Theorem 1.3.7 ensures a polynomial (in n) reduction for a strictly larger class of graph-induced distributions than Corollary 1.3.10. To this end, let $\{G_n\}_{n\in\mathbb{N}}$ be the sequence of graphs obtained by attaching a star graph on $\log n$ vertices to a chain of $n - \log n$ vertices. Then $S(G) = \mathsf{poly}(n)$ while $B(G) = \log n > \mathsf{const}$.

Recall the hybrid structure of our proofs. We had (roughly) n hybrid steps as G might decompose into roughly n connected components by removing the vertices corresponding to opened indices. On the other hand, when covering a hybrid step, in the worst case, a connected component could contain (roughly n) vertices. Clearly, these two worst cases are mutually exclusive but our given hybrid approach was too rigid to exploit that. In the next section we devise a tighter reduction.

1.3.4 A Tighter Reduction for Directed Graphs

Directed Graphs We slightly refine our definition of directed graphs. In this section we say a graph $G = (V, E)$ is *directed*, if it does not contain an undirected edge. That is, for all $(u, v) \in V^2$ we have $\{(u, v), (v, u)\} \cap E \leq 1$. We refer to an directed, acyclic graph as *DAG*. For a *DAG G* we require that the vertices are ordered in such a way that there is *no* directed path from i to j for $i < j$. Such an ordering always exists as G has neither cycles nor undirected edges.

In our proofs we always traverse dependencies *backwards*. For instance, the distribution of m_i solely depends on m_j then m_i is switched from sampled to resampled *before* m_j is replaced. As in the previous proofs, we perform n hybrid steps. Thereby, $m_1, \ldots, m_i$ will be resampled in the i^{th} hybrid.

Theorem 1.3.11 *Let $\mathcal{D}$ be the class of efficiently resampleable sequences of distributions induced by sequences of connected DAGs $\{G_n\}_{n\in\mathbb{N}}$.*

If PKE *is $(\tau_{cpa}, \varepsilon_{cpa})$-IND-CPA secure, then* PKE *is $(\tau_{so}, \varepsilon_{so})$-IND-SO-CPA secure where*

$$\tau_{so} \leq \tau_{cpa} - 3 \cdot \tau_{resamp} \ , \qquad \varepsilon_{so}(n) \leq 3 \cdot n^{B(G_n)+1} \cdot \varepsilon_{cpa}$$

where τ_{resamp} is the time of one execution of the resampling algorithm.

Observe that Theorem 1.3.11 gives a reduction to mult-IND-CPA security tighter by a factor of roughly n compared to Corollary 1.3.10.

Exp $\mathsf{H}_k^{\mathcal{A}_{so}}(n)$
01 $\mathcal{I} \leftarrow \emptyset$
02 $(pk, sk) \leftarrow_\$ \mathsf{PKE.Gen}$
03 $(\mathfrak{D}, st_1) \leftarrow_\$ \mathcal{A}_{so,1}(pk, n)$
04 $\mathbf{m}^0 \leftarrow_\$ \mathfrak{D}$
05 $\mathbf{r} \leftarrow_\$ \mathcal{R}^n$
06 $\mathbf{c} \leftarrow \mathsf{PKE.Enc}_{pk}(\mathbf{m}^0; \mathbf{r})$
07 $st_2 \leftarrow_\$ \mathcal{A}_{so,2}^{\text{OPEN}}(st_1, \mathbf{c})$
08 $\mathbf{m} \leftarrow \mathsf{Resamp}_{\mathfrak{D}}(\mathbf{m}^0, [k+1, n] \cup \mathcal{I})$
09 $b' \leftarrow_\$ \mathcal{A}_{so,3}(st_2, \mathbf{m})$
10 Stop with b'

Oracle $\text{OPEN}(i)$
11 $\mathcal{I} \leftarrow \mathcal{I} \cup \{i\}$
12 Return (m_i^0, r_i)

Figure 1.6: Hybrid experiments $\mathsf{H}_k(n)$ used in the proof of Theorem 1.3.11. The experiment only differs from the IND-SO-CPA experiment (see Figure 1.2) by lines 08 (sampling a hybrid challenge vector) and 09 where $\mathcal{A}_{so}$ is invoked on it.

Proof of Theorem 1.3.11. Let $\mathcal{A}_{so} = (\mathcal{A}_{so,1}, \mathcal{A}_{so,2}, \mathcal{A}_{so,3})$ be an adversary that breaks the $(\tau_{so}, \varepsilon_{so})$-IND-SO-CPA security of PKE for some $n \in \mathbb{N}$. We proceed in a sequence of hybrid experiments $\mathsf{H}_0, \mathsf{H}_1, \ldots, \mathsf{H}_n$ as given in Figure 1.6. We switch m_{k+1} from sampled to resampled in the hybrid transition H_k to H_{k+1}. Hybrid H_k returns the sampled plaintexts for all positions $[k+1, n] \cup \mathcal{I}$, but resampled plaintexts on all positions $[k] \setminus \mathcal{I}$ where the resampling is conditioned on *every plaintext in* $[k+1, n] \cup \mathcal{I}$.

Hybrid H_0 is identical to $\mathsf{IND\text{-}SO\text{-}CPA}_0$ experiment and H_n is identical to the $\mathsf{IND\text{-}SO\text{-}CPA}_1$ experiment. Thus

$$\begin{aligned}
\left|\Pr\left[\mathsf{IND\text{-}SO\text{-}CPA}_0^{\mathcal{A}_{so}}(n) \Rightarrow 1\right] - \left|\Pr\left[\mathsf{IND\text{-}SO\text{-}CPA}_1^{\mathcal{A}_{so}}(n) \Rightarrow 1\right]\right|\right.& \\
= \left|\Pr\left[\mathsf{H}_0^{\mathcal{A}_{so}}(n) \Rightarrow 1\right] - \Pr\left[\mathsf{H}_n^{\mathcal{A}_{so}}(n) \Rightarrow 1\right]\right|& \\
\leq \sum_{k=0}^{n-1} \left|\Pr\left[\mathsf{H}_k^{\mathcal{A}_{so}}(n) \Rightarrow 1\right] - \Pr\left[\mathsf{H}_{k+1}^{\mathcal{A}_{so}}(n) \Rightarrow 1\right]\right| &\quad . \quad (1.2)
\end{aligned}$$

We proceed with Lemma 1.3.12 to bound the distance between two consecutive hybrids H_k and H_{k+1}.

Lemma 1.3.12 *There exists $\mathcal{A}_{m\text{-}cpa} = (\mathcal{A}_{m\text{-}cpa,1}, \mathcal{A}_{m\text{-}cpa,2})$ and $n_{m\text{-}cpa} \in \mathbb{N}$ such that $\mathcal{A}_{m\text{-}cpa}$ breaks the $(\tau_{m\text{-}cpa}, \varepsilon_{m\text{-}cpa})$-mult-IND-CPA security of* PKE *for $n_{m\text{-}cpa}$ where $\tau_{m\text{-}cpa} \approx \tau_{so} + 3 \cdot \tau_{resamp}$ and*

$$\varepsilon_{m\text{-}cpa} \geq \Pr\left[\overline{\textsc{Abort}_k}\right] \cdot \left|\Pr\left[\mathsf{H}_k^{\mathcal{A}_{so}}(n) \Rightarrow 1\right] - \Pr\left[\mathsf{H}_{k+1}^{\mathcal{A}_{so}}(n) \Rightarrow 1\right]\right| ,$$

where

$$\Pr\left[\overline{\textsc{Abort}_k}\right]^{-1} \leq \begin{cases} \sum_{i=0}^{B(G_n)-1} \binom{k}{i} & \text{for } k < n-1 \\ \sum_{i=0}^{B(G_n)} \binom{k}{i} & \text{for } k = n-1 \end{cases}$$

and τ_{resamp} is the time of one execution of the resampling algorithm.

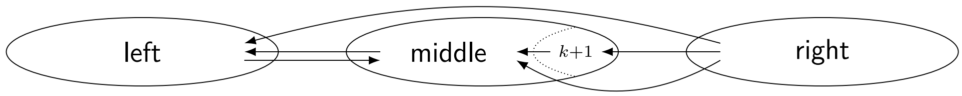

Figure 1.7: Structure of G. Edges between particular sets cannot exist if there is no arrow depicted. If right $\neq \emptyset$, there is at least one edge from right to middle since G is connected. left and middle are disconnected in $G_{\overline{\mathcal{I}}}$.

Proof Sketch We construct a mult-IND-CPA adversary $\mathcal{A}_{m\text{-}cpa}$ that interpolates between hybrids H_k and H_{k+1}. Ideally, $\mathcal{A}_{m\text{-}cpa}$ embeds its own challenge exactly at position $k+1$. However, it might have to resample some already resampled plaintexts in $\mathbf{m}_{[k]}$ to avoid inconsistencies as we see shortly.

We introduce some notation for the proof: Let middle denote the connected component in $G_{[k+1]\setminus\mathcal{I}}$ that contains $\mathbf{m}_{k+1}$. Let right $:= [k+2, n]$, and left $:= \overline{(\mathsf{middle} \cup \mathsf{right})}$ (Figure 1.7).

Plaintexts in right are not resampled in hybrids H_k or H_{k+1}. Further, middle and left are disconnected in $G_{\overline{\mathcal{I}}}$. Hence, $\mathcal{A}_{m\text{-}cpa}$ merely has to guess the neighborhood of middle in order to correctly resample middle in advance.

Recall that (in particular) plaintexts in left have to be resampled as specified by the hybrid experiment. However, since middle and left are disconnected in $G_{\overline{\mathcal{I}}}$, $\mathcal{A}_{m\text{-}cpa}$ can wait for all opening queries to happen before resampling the left plaintexts.

Finally, observe that G is connected, i.e., $N(\mathsf{middle})$ contains at least one vertex from right $= [k+2, n]$ as long as $k < n-1$.

Since right is fixed while resampling anyway, it suffices to guess $N(\mathsf{middle}) \cap [k]$ whereby for all $k < n-1$ we have $|N(\mathsf{middle}) \cap [k]| \leq B(G) - 1$.

Proof of Lemma 1.3.12. For $k \in [0, n]$ and $i \in [n]$ let $\textsc{Open}_k(i)$ denote the event that $\mathcal{A}_{so}$ queries $\textsc{Open}(i)$ in hybrid H_k. Note that the view of $\mathcal{A}_{so}$ is identical until it receives

Adversary $\mathcal{A}_{m\text{-}cpa,1}(pk, n_{m\text{-}cpa})$

$\mathfrak{D} \leftarrow_{\$} \mathcal{A}_{so,1}(pk, n)$

$N^* \leftarrow_{\$} \begin{cases} \{V' \subseteq [k] \colon |V'| \in [0, B(G)-1]\} & \text{for } k < n-1 \\ \{V' \subseteq [k] \colon |V'| \in [0, B(G)]\} & \text{else} \end{cases}$

// Let middle^* denote the connected component
// in $G_{[k+1]\setminus N^*}$ that contains vertex $k+1$.

$\mathbf{m}^0 \leftarrow_{\$} \mathfrak{D}$

$\mathbf{m}^{1,0} \leftarrow \mathsf{Resamp}_{\mathfrak{D}}(\mathbf{m}^0, N^* \cup \{k+1\} \cup \mathsf{right})$

$\mathbf{m}^{1,1} \leftarrow \mathsf{Resamp}_{\mathfrak{D}}(\mathbf{m}^0, N^* \cup \mathsf{right})$

Output $(\mathbf{m}^{1,0}_{\mathsf{middle}^*}, \mathbf{m}^{1,1}_{\mathsf{middle}^*})$

Adversary $\mathcal{A}_{m\text{-}cpa,2}(\mathbf{c}_{\mathsf{middle}^*})$

$\mathbf{r} \leftarrow_{\$} \mathcal{R}^n$

$c_i \leftarrow \begin{cases} c_i & \text{for } i \in \mathsf{middle}^* \\ \mathsf{PKE.Enc}_{pk}(m_i^0; r_i) & \text{else} \end{cases}$

$\mathbf{c} \leftarrow (c_1, \ldots, c_n)$

$\mathcal{I} \leftarrow \emptyset$

$() \leftarrow_{\$} \mathcal{A}_{so,2}^{\textsc{Open}}(\mathbf{c})$

If $N^* \not\subseteq \mathcal{I}$: Abort

$\mathbf{m}^1 \leftarrow_{\$} \mathsf{Resamp}_{\mathfrak{D}}(\mathbf{m}^0, \mathcal{I} \cup \mathsf{right})$

$m_i \leftarrow \begin{cases} m_i^1 & \text{for } i \in \mathsf{left} \\ m_i^0 & \text{else} \end{cases}$

$\mathbf{m} \leftarrow (m_1, \ldots, m_n)$

$b' \leftarrow_{\$} \mathcal{A}_{so,3}(\mathbf{m})$

Output b'

Oracle $\textsc{Open}(i)$

If $i \in \mathsf{middle}^*$: Abort

$\mathcal{I} \leftarrow \mathcal{I} \cup \{i\}$

Return (m_i, r_i)

Figure 1.8: Pseudocode of adversary $\mathcal{A}_{m\text{-}cpa} = (\mathcal{A}_{m\text{-}cpa,1}, \mathcal{A}_{m\text{-}cpa,2})$. $\mathcal{A}_{m\text{-}cpa}$ interpolates between hybrids H_k, H_{k+1} for $\mathcal{A}_{so}$. For clarity we abstain from making the states output by and returned to $\mathcal{A}_{so}$ and $\mathcal{A}_{m\text{-}cpa}$ explicit.

its challenge, hence $\Pr[\textsc{Open}_s(i)] = \Pr[\textsc{Open}_t(i)]$ for all $s, t \in [0, n]$ and all $i \in [n]$. Additionally, two consecutive hybrids H_k, H_{k+1} only differ on m_{k+1} *unless* $\mathcal{A}_{so}$ calls $\textsc{Open}(k+1)$, i.e. enforcing m_{k+1} to remain sampled. Thus, we have

$$\Pr\left[\mathsf{H}_k^{\mathcal{A}_{so}} \Rightarrow 1 \wedge \textsc{Open}_k(k+1)\right] = \Pr\left[\mathsf{H}_{k+1}^{\mathcal{A}_{so}} \Rightarrow 1 \wedge \textsc{Open}_{k+1}(k+1)\right]$$

and obtain

$$\begin{aligned} &\left|\Pr\left[\mathsf{H}_k^{\mathcal{A}_{so}}(n) \Rightarrow 1\right] - \Pr\left[\mathsf{H}_{k+1}^{\mathcal{A}_{so}}(n) \Rightarrow 1\right]\right| \\ &= \left|\Pr\left[\mathsf{H}_k^{\mathcal{A}_{so}} \Rightarrow 1 \wedge \overline{\textsc{Open}_k(k+1)}\right] - \Pr\left[\mathsf{H}_{k+1}^{\mathcal{A}_{so}} \Rightarrow 1 \wedge \overline{\textsc{Open}_{k+1}(k+1)}\right]\right| \quad . \end{aligned} \tag{1.3}$$

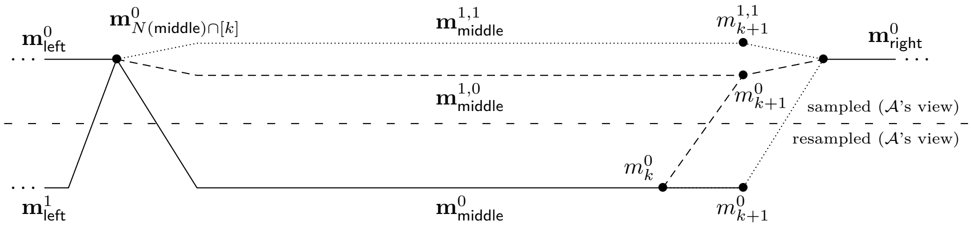

Figure 1.9: Structure of plaintexts sampled during the reduction by $\mathcal{A}_{m\text{-}cpa}$ interpolating between hybrids H_k and H_{k+1}. We assume that $\mathcal{A}_{m\text{-}cpa}$ guessed N^* correctly. Plaintexts above (resp. below) the horizontal dashed line appear sampled (resp. resampled) from $\mathcal{A}_{so}$'s view. If $\mathcal{A}_{m\text{-}cpa}$ challenge contains ciphertexts $\mathsf{PKE.Enc}_{pk}(\mathbf{m}^{1,1}_{\mathsf{middle}})$ (that is, middle is entirely resampled), then from $\mathcal{A}_{so}$'s view all (actually sampled) plaintexts $\mathbf{m}^0_{\mathsf{middle}}$ appear resampled. The relevant structures are indicated by solid and dotted lines. If $\mathcal{A}_{m\text{-}cpa}$ receives $\mathsf{PKE.Enc}_{pk}(\mathbf{m}^{1,0}_{\mathsf{middle}})$ (middle plaintexts resampled *except for* m_{k+1}), then from $\mathcal{A}_{so}$'s view all (actually sampled) plaintexts $\mathbf{m}^0_{\mathsf{middle}}$ appear resampled *except for* m_{k+1}. The relevant structures are indicated by solid and dashed lines.

We construct mult-IND-CPA adversary $\mathcal{A}_{m\text{-}cpa}$. Its pseudocode is given in Figure 1.8. Again, the choice of $n_{m\text{-}cpa}$ follows from the proof. It relays (pk, n) to $\mathcal{A}_{so}$. Receiving $\mathfrak{D}$, $\mathcal{A}_{m\text{-}cpa,1}$ guesses middle via its neighborhood N^* (lines 02, 03). We let middle^* denote $\mathcal{A}_{m\text{-}cpa}$'s guess for component middle. Adversary $\mathcal{A}_{m\text{-}cpa,1}$ then proceeds by sampling $\mathbf{m}^0$ (line 04) and resamples $\mathbf{m}^{1,0}$ (resp. $\mathbf{m}^{1,1}$) fixing plaintexts in $N^* \cup \{k+1\} \cup \mathsf{right}$ (resp. $N^* \cup \mathsf{right}$) to obtain its plaintext vectors $(\mathbf{m}^{1,0}_{\mathsf{middle}^*}, \mathbf{m}^{1,1}_{\mathsf{middle}^*})$ sent to the mult-IND-CPA experiment (lines 05-07).

$\mathcal{A}_{m\text{-}cpa,2}$ is started on $\mathbf{c}_{\mathsf{middle}^*}$, samples fresh randomness and encrypt plaintexts in $\overline{\mathsf{middle}^*}$ on its own while embedding its challenge in the middle^* positions (lines 08, 09). Then $\mathcal{A}_{so,2}$ is invoked on $\mathbf{c}$ and opening queries are answered honestly unless they occur on middle^* where $\mathcal{A}_{m\text{-}cpa,2}$ aborts as it guessed middle incorrectly (lines 19 and 21).[4] Once $\mathcal{A}_{so,2}$ terminates, $\mathcal{A}_{m\text{-}cpa,2}$ checks if $N^* \subseteq \mathcal{I}$ in line 13, if not $\mathcal{A}_{m\text{-}cpa}$'s guess for middle was wrong and it aborts. Otherwise, $\mathcal{A}_{m\text{-}cpa,2}$ resamples plaintexts fixing those at positions $\mathcal{I} \cup \mathsf{right}$ to obtain correctly distributed resampled plaintexts for positions left (line 14). Eventually, $\mathcal{A}_{so,3}$ is run on $(\mathbf{m}^1_{\mathsf{left}}, \mathbf{m}^0_{\overline{\mathsf{left}}})$ (see lines 15-17) and $\mathcal{A}_{m\text{-}cpa,2}$ outputs $\mathcal{A}_{so,3}$'s output (line 18).

Analysis Assume that $\mathcal{A}_{m\text{-}cpa}$ guessed correctly, i.e. N^* is the neighborhood of middle in $G_{[k]}$. Then $\mathsf{middle}^* = \mathsf{middle}$ holds and by definition of middle, $\mathcal{A}_{m\text{-}cpa}$ will not abort.

Clearly, $\mathcal{A}_{m\text{-}cpa}$ correctly simulates $\mathcal{A}_{so}$'s hybrid view in all left and right positions.

Note that $\mathcal{A}_{so,2}$ obtains encryptions of *resampled* encryptions $\mathsf{PKE.Enc}_{pk}(\mathbf{m}^{1,b}_{\mathsf{middle}})$ (line 12), but expects *sampled* encryptions $\mathsf{PKE.Enc}_{pk}(\mathbf{m}^0_{\mathsf{middle}})$. Further, $\mathcal{A}_{so,3}$ is run on

[4]Note that we condition our analysis on $\mathcal{A}_{so,2}$ not issuing $\textsc{Open}(k+1)$. See Equation (1.3).

sampled $\mathbf{m}^0_{\mathsf{middle}}$ expecting *resampled* $\mathbf{m}_{\mathsf{middle}}$ (line 17). Thus, *sampled* middle plaintexts become *resampled* middle plaintexts from $\mathcal{A}_{so}$'s view and vice versa.

However, we have $\mathbf{m}_{\mathsf{middle}} \equiv \mathbf{m}^0_{\mathsf{middle}}$ since $N(\mathsf{middle}) \subseteq \mathcal{I} \cup \mathsf{right}$, where $\mathcal{I} \cup \mathsf{right}$ is fixed when resampling $\mathbf{m}_{\mathsf{middle}}$.

For the next arguments the reader *may* find Figure 1.9 helpful. For $\mathcal{A}_{m\text{-}cpa}$ run in experiment $\mathsf{mult\text{-}IND\text{-}CPA}_1$, $\mathcal{A}_{so,2}$ receives $\mathsf{PKE.Enc}_{pk}(\mathbf{m}^{1,1}_{\mathsf{middle}})$ where $\mathbf{m}^{1,1}_{\mathsf{middle}} \equiv \mathbf{m}^0_{\mathsf{middle}}$ since $N^* \cup \mathsf{right} = N \cup \mathsf{right}$ is fixed when $\mathbf{m}^{1,1}$ is resampled. Hence, *all* middle plaintexts sent to $\mathcal{A}_{so,3}$ appear resampled to it and $\mathcal{A}_{so}$'s view is identical to hybrid H_{k+1}.

When $\mathcal{A}_{m\text{-}cpa}$ is run in the $\mathsf{mult\text{-}IND\text{-}CPA}_0$ experiment, it calls $\mathcal{A}_{so,2}$ on $\mathsf{PKE.Enc}_{pk}(\mathbf{m}^{1,0}_{\mathsf{middle}})$. Thereby $\mathbf{m}^{1,0}_{\mathsf{middle}} \equiv \mathbf{m}^1_{\mathsf{middle}}$ for the same reason as before. In particular, we have $m^0_{k+1} = m^{1,0}_{k+1}$ since m^0_{k+1} is fixed while resampling (see line 05). Consequently, each plaintext in middle *except* the $(k+1)^{th}$ appears resampled to $\mathcal{A}_{so,3}$ and its view is identical to hybrid H_k. Let $\textsc{Abort}_k$ denote the event that $\mathcal{A}_{m\text{-}cpa}$ aborts its execution because it guessed N^* incorrectly (see line 02 in Figure 1.8). Clearly, $\mathcal{A}_{m\text{-}cpa}$ outputs 1 in its mult-IND-CPA experiment iff $\mathcal{A}_{so}$ outputs 1 in its respective hybrid and $\mathcal{A}_{m\text{-}cpa}$ does not abort:

$$
\begin{aligned}
&\Pr\left[\mathsf{mult\text{-}IND\text{-}CPA}_0^{\mathcal{A}_{m\text{-}cpa}} \Rightarrow 1\right] = \Pr\left[\mathsf{H}_k^{\mathcal{A}_{so}} \Rightarrow 1 \wedge \overline{\textsc{Abort}_k} \wedge \overline{\textsc{Open}_k(k+1)}\right] \\
\text{and}\quad &\Pr\left[\mathsf{mult\text{-}IND\text{-}CPA}_1^{\mathcal{A}_{m\text{-}cpa}} \Rightarrow 1\right] = \Pr\left[\mathsf{H}_{k+1}^{\mathcal{A}_{so}} \Rightarrow 1 \wedge \overline{\textsc{Abort}_k} \wedge \overline{\textsc{Open}_{k+1}(k+1)}\right] .
\end{aligned}
$$

We conclude:

$$
\begin{aligned}
\varepsilon_{m\text{-}cpa} &= \left|\Pr\left[\mathsf{mult\text{-}IND\text{-}CPA}_0^{\mathcal{A}_{m\text{-}cpa}} \Rightarrow 1\right] - \Pr\left[\mathsf{mult\text{-}IND\text{-}CPA}_1^{\mathcal{A}_{m\text{-}cpa}} \Rightarrow 1\right]\right| \\
&= \left|\Pr\left[\mathsf{H}_k^{\mathcal{A}_{so}} \Rightarrow 1 \wedge \overline{\textsc{Abort}_k} \wedge \overline{\textsc{Open}_k(k+1)}\right]\right. \\
&\quad \left. - \Pr\left[\mathsf{H}_{k+1}^{\mathcal{A}_{so}} \Rightarrow 1 \wedge \overline{\textsc{Abort}_k} \wedge \overline{\textsc{Open}_{k+1}(k+1)}\right]\right| .
\end{aligned}
$$

Since $\overline{\textsc{Abort}_k}$ is independent of $\left(\mathsf{H}_i^{\mathcal{A}_{so}} \Rightarrow 1 \wedge \overline{\textsc{Open}_i(k+1)}\right)$ for $i \in \{k, k+1\}$ we have

$$
\begin{aligned}
&= \Pr\left[\overline{\textsc{Abort}_k}\right] \cdot \left|\Pr\left[\mathsf{H}_k^{\mathcal{A}_{so}} \Rightarrow 1 \wedge \overline{\textsc{Open}_k(k+1)}\right]\right. \\
&\qquad\qquad \left. - \Pr\left[\mathsf{H}_{k+1}^{\mathcal{A}_{so}} \Rightarrow 1 \wedge \overline{\textsc{Open}_{k+1}(k+1)}\right]\right| \\
&= \Pr\left[\overline{\textsc{Abort}_k}\right] \cdot \left|\Pr\left[\mathsf{H}_k^{\mathcal{A}_{so}}(n) \Rightarrow 1\right] - \Pr\left[\mathsf{H}_{k+1}^{\mathcal{A}_{so}}(n) \Rightarrow 1\right]\right| .
\end{aligned}
$$

Where we applied (1.3) in the last step. Hence, overall:

$$\varepsilon_{m\text{-}cpa} = \Pr\left[\overline{\textsc{Abort}_k}\right] \cdot \left|\Pr\left[\mathsf{H}_k^{\mathcal{A}_{so}}(n) \Rightarrow 1\right] - \Pr\left[\mathsf{H}_{k+1}^{\mathcal{A}_{so}}(n) \Rightarrow 1\right]\right| \ .$$

To conclude the proof of Lemma 1.3.12 we observe that $\mathcal{A}_{m\text{-}cpa}$ picks N^* uniformly from a set of size $\sum_{i=0}^{B(G_n)-1}\binom{k}{i}$ for $k < n-1$, and of size $\sum_{i=0}^{B(G_n)}\binom{k}{i}$ for $k = n-1$. Hence,

$$\Pr\left[\overline{\textsc{Abort}_k}\right]^{-1} \leq \begin{cases} \sum_{i=0}^{B(G_n)-1}\binom{k}{i} & \text{for } k < n-1 \\ \sum_{i=0}^{B(G_n)}\binom{k}{i} & \text{for } k = n-1 \ . \end{cases}$$

As $\mathcal{A}_{m\text{-}cpa}$ submits vectors of length $|\mathsf{middle}^*|$, we let $n_{m\text{-}cpa} := |\mathsf{middle}^*|$.

One easily verifies the running time of $\mathcal{A}_{m\text{-}cpa}$ to be roughly $\tau_{so} + 3 \cdot \tau_{resamp}$. ∎

The remaining proof of Theorem 1.3.11 consists of tedious computations. From Equation (1.2) and Lemma 1.3.12 we have

$$\left|\Pr\left[\mathsf{IND\text{-}SO\text{-}CPA}_0^{\mathcal{A}_{so}}(n) \Rightarrow 1\right] - \Pr\left[\mathsf{IND\text{-}SO\text{-}CPA}_1^{\mathcal{A}_{so}}(n) \Rightarrow 1\right]\right| \\ \leq \sum_{k=0}^{n-1} \Pr[\overline{\textsc{Abort}_k}]^{-1} \cdot \varepsilon_{m\text{-}cpa} \ .$$

Let $B := B(G_n)$. Since $n_{m\text{-}cpa} \leq k+1$ and by Lemma 1.2.3 we have

$$\sum_{k=0}^{n-1} \Pr\left[\overline{\textsc{Abort}_k}\right]^{-1} \cdot \varepsilon_{m\text{-}cpa} \leq \left(\sum_{k=0}^{n-2}(k+1) \cdot \sum_{i=0}^{B-1}\binom{k}{i} + n \cdot \sum_{i=0}^{B}\binom{n-1}{i}\right) \cdot \varepsilon_{cpa} \quad (1.4)$$

for a $(\tau_{cpa}, \varepsilon_{cpa})$-IND-CPA adversary with running time $\tau_{cpa} \approx \tau_{m\text{-}cpa}$. (For now) let $2 \leq B < n$. We evaluate the factor of ε_{cpa} in Equation (1.4).

$$\begin{aligned} &\sum_{k=0}^{n-2}(k+1) \cdot \sum_{i=0}^{B-1}\binom{k}{i} + n \cdot \sum_{i=0}^{B}\binom{n-1}{i} \\ &= \sum_{i=0}^{B-1}\binom{0}{i} + 2 \cdot \sum_{i=0}^{B-1}\binom{1}{i} + \sum_{k=2}^{n-2}(k+1) \cdot \sum_{i=0}^{B-1}\binom{k}{i} + n \cdot \sum_{i=0}^{B}\binom{n-1}{i} \\ &\leq 5 + \sum_{k=2}^{n-2}(k+1) \cdot \sum_{i=0}^{B-1} k^i + n \cdot \sum_{i=0}^{B}\binom{n-1}{i} \\ &= 5 + \sum_{k=2}^{n-2}(k+1) \cdot \frac{k^B - 1}{k-1} + n \cdot \sum_{i=0}^{B}\binom{n-1}{i} \end{aligned}$$

$$
\begin{aligned}
&= 5 + \sum_{k=2}^{n-2} \underbrace{\frac{k+1}{k-1}}_{\leq 3} \cdot (k^B - 1) + n \cdot \sum_{i=0}^{B} \binom{n-1}{i} \\
&\leq 5 + 3 \cdot \sum_{k=2}^{n-2} (k^B - 1) + n \cdot \sum_{i=0}^{B} \binom{n-1}{i} \\
&= 5 + 3 \cdot \sum_{k=2}^{n-2} k^B - 3 \cdot (n-3) + n \cdot \sum_{i=0}^{B} \binom{n-1}{i} \\
&= 14 - 3n + 3 \cdot \sum_{k=2}^{n-2} k^B + n \cdot \sum_{i=0}^{B} \binom{n-1}{i} \\
&= 11 - 3n + 3 \cdot \sum_{k=0}^{n-2} k^B + n \cdot \sum_{i=0}^{B} \binom{n-1}{i} \quad \text{since } B \geq 1 \\
&\leq 11 - 3n + 3 \cdot \sum_{k=0}^{n-2} k^B + n \cdot \sum_{i=0}^{B} n^i \\
&= 11 - 3n + 3 \cdot \sum_{k=0}^{n-2} k^B + n \cdot \frac{n^{B+1} - 1}{n-1} \\
&= 11 - 3n + 3 \cdot \sum_{k=0}^{n-2} k^B + \underbrace{\frac{n}{n-1}}_{\leq 2} \cdot (n^{B+1} - 1) \quad \text{since } n \geq 2 \\
&\leq 9 - 3n + 3 \cdot \sum_{k=0}^{n-2} k^B + 2 \cdot n^{B+1} \\
&\leq 9 - 3n + 3 \cdot \int_0^n k^B \mathrm{d}k + 2 \cdot n^{B+1} \\
&= 9 - 3n + 3 \cdot \frac{n^{B+1}}{B+1} + 2 \cdot n^{B+1} \\
&= 9 - 3n + \left(2 + \frac{3}{B+1}\right) \cdot n^{B+1} \\
&\leq 9 - 3n + 3 \cdot n^{B+1} \quad \text{since } B \geq 2 \\
&\leq 3 \cdot n^{B+1} \quad \text{since } n \geq 3 \ .
\end{aligned}
$$

Since G is connected we have $B = 0 \Leftrightarrow n = 1$ and $B = 1 \Leftrightarrow n = 2$. Thus, it is easily verified that the bound holds for $(B, n) \in \{(0, 1), (1, 2)\}$ as well to complete the proof of Theorem 1.3.11. ∎

Because Markov distributions are DAG-induced by chain graphs and the maximum border of a chain graph is 2 (see Definition 1.3.3), we immediately obtain a tighter version of Corollary 1.3.9 whose proof directly follows from Theorem 1.3.11.

Corollary 1.3.13 *If* PKE *is $(\tau_{cpa}, \varepsilon_{cpa})$-IND-CPA secure, then* PKE *is $(\tau_{so}, \varepsilon_{so})$-IND-SO-CPA secure w.r.t efficiently resampleable Markov distributions where*

$$\tau_{so} \leq \tau_{cpa} - 3 \cdot \tau_{resamp} \ , \qquad \varepsilon_{so}(n) \leq 3 \cdot n^3 \cdot \varepsilon_{cpa}$$

and τ_{resamp} is the time of one execution of the resampling algorithm.

Further Incremental Improvements Recall that the hybrids in the proof of Theorem 1.3.11 allowed for a reduction tighter by a factor of n as it suffices to guess a set of size at most $B(G) - 1$ instead of $B(G)$ for $k < n - 1$ as at least one vertex of the neighborhood of middle is contained in right, thus, fixed during resampling anyway. One may apply the novel hybrid structure introduced in the proof of Theorem 1.3.11 to improve the results of Theorem 1.3.7. Thereby, it suffices to guess a connected subgraph C_{k+1} in $[k+1]$ (instead of $[n]$ as done in the proof of Theorem 1.3.7) containing vertex $k+1$.

Now, recall that the set $\{k+1\} \cup \mathsf{right} = [k+1, n] \subset V$ of size $|[k+1, n]| = n - k$ is connected in G as G is connected.

The following observation shows that there are at least $n - k$ subsets of $[k+1, n]$ that contain $k+1$ and are connected in G: Take vertices $[k+1, n]$ and remove edges until a tree remains. Now, keeping $k+1$ but iteratively removing other vertices with degree one, results in a still connected subset of $[k+1, n]$ containing $k+1$.

Hence, each subset S_i such that $k+1 \in S_i \subseteq [k+1]$ that is connected in G can be extended to at least $n - k$ different sets $S_i^1, \dots S_i^{n-k} \subseteq [n]$ that are connected in G. Thus, if G has $S(G)$ connected subgraphs, then for any $k \in [n]$ graph $G_{[k+1]}$ has no more than $S(G)/(n-k)$ connected subgraphs. Now, guessing a connected subgraph from $[k+1]$ instead of $[n]$ increases the probability of guessing C_{k+1} correctly from at least $1/S(G)$ to at least $(n-k)/S(G)$.

Tedious but simple computations show that, eventually, the loss of $\mathcal{O}(n^2) \cdot S(G)$ (see Theorem 1.3.7) can be reduced to $\mathcal{O}(n \log n) \cdot S(G)$.

1.3.5 The Structure of Graphs with Low Connectivity Properties

In this section we devote some time to understanding the structure of graphs for which Theorem 1.3.7 and Corollary 1.3.10 ensure an at most polynomial loss (in n). Proposition 1.3.18 and parts of Proposition 1.3.21 were obtained when discussing the structure of B-good graphs with Jorge Villar [Vil15, HV17]. To ease our way of speaking we introduce some notation.

Definition 1.3.14 (good graph sequences). Let $\{G_n\}_{n\in\mathbb{N}}$ be a sequence of connected graphs such that for all $n \in \mathbb{N}$ graph G_n has n vertices. We say that $\{G_n\}_{n\in\mathbb{N}}$ is B-*good* iff $B(G_n) = \mathsf{const}$ for all $n \in \mathbb{N}$. We say $\{G_n\}_{n\in\mathbb{N}}$ is S-*good* iff $S(G_n) = \mathsf{poly}(n)$ for all $n \in \mathbb{N}$. We say $\{G_n\}_{n\in\mathbb{N}}$ is *good* iff $\{G_n\}_{n\in\mathbb{N}}$ is B-good or S-good. A member of a family of B- good (resp. S-good, good) graphs is called B-good (resp. S-good, good).

Thus, we address the following question in this section:

What do sequences of good graphs look like?

Given Theorem 1.3.5 and the observation at the end of Section 1.3.2 we recall that any B-good family is S-good, while the converse is generally false.

Clearly, families of chain graphs are good. The same applies to slight variations of chain graphs obtained by iteratively adding a constant number of vertices v to the graph such that $deg(v) = 1$. Further, graphs constructed by taking a chain graph and attaching[5] a constant number of chains to it preserves goodness.

However, playing around with these toy examples we are seemingly stuck with versions of chain graphs, i.e., quite 'stretched' graphs if we want to ensure goodness. So, are good families of graphs inherently 'slender'?

At first, one might be tempted trying to characterize the structure of good graphs through sparsity, as all good graphs discovered so far happen to have few edges. However, this approach is doomed to fail: Consider the sequences of star graphs and the sequence of chain graphs on n vertices. Members of both sequences have as few edges as possible such that they are connected. Though, each star graph is not good while each chain graph is B- and S-good.

Before turning towards defining what it is supposed to mean for a graph to be 'slender', we can readily derive a necessary condition for a graph to be B-good (resp. S-good).

Definition 1.3.15 (branching paths). Let $G = (V, E)$ be a graph and p be a path in G. We say that p *branches* $N(p)$ *times*.

Recall that $N(p)$ denotes the (open) neighborhood of p (see Section 1.3.1).

Proposition 1.3.16 *Any path in a graph G branches at most* $\min\{B(G), \log_2 S(G)\}$ *times.*

[5] Here *attach* is to be understood as taking one of the two degree-one vertices of the new chain that is to be added and place an edge between it and any vertex in the graph constructed so far.

Proof. Let $p = (p_1, \dots, p_\ell)$ be a path in G that branches k times, i.e. $N(p) = k$. Let $\{p\} := \{p_1, \dots, p_\ell\}$. As the induced subgraph $G_{\{p\}}$ is connected we have $k \le B(G)$.

Let $2^{N(p)}$ denote the power set over $N(p)$. Then for any $P \in 2^{N(p)}$ the induced subgraph $G_{\{p\}\cup P}$ is connected. Hence, $S(G) \ge |2^{N(p)}| = 2^{|N(p)|} = 2^k$. Thus we have $k \le \log_2 S(G)$. ■

An immediate consequence of Proposition 1.3.16 is that paths in a good graph do not branch more than $\mathcal{O}(\log n)$ times. Recall that sequences of graphs $\{G_n\}_{n\in\mathbb{N}}$ correspond to distributions, in particular, chain graphs capture Markov distributions. Proposition 1.3.16 shows us that good graphs essentially do consist of chain graphs as any path in it is a chain up to logarithmically many 'forks'[6] (constantly many, in the case of a B-good graph). Hence, distributions captured by good families of graphs are conceptually close to Markov distributions.

We now draw our attention towards B-good graphs. We begin by defining 'slenderness' in a graph-theoretic terminology.

Definition 1.3.17 (graph diameter). Let $G = (V, E)$ be a connected graph. We define the *diameter of* G, $D(G)$, as the maximum over the distances of any two distinct vertices in G:

$$D(G) := \max_{\substack{u,v\in V\\ u\neq v}} \{d(u,v)\} \quad .$$

We observe that the diameter of a graph reflects our intuitive understanding of the graph's shape. Graphs with a small diameter have only short distances, thus will be quite 'compact'. In contrast, a large diameter implies that there are vertices that are far apart. Hence, the graph is quite 'long' (and thus has to be 'slender').

Finally, we present two results on B-good graphs. First, the diameter of B-good graphs grows linearly in $|V|$. Secondly, B-good graphs are somewhat rare objects. For the second result we study Erdős–Rényi (random) graphs. We show that any size $|V|/2$ set is expected to have a neighborhood that grows linearly in $|V|$ rather than being constant as for B-good graphs.

Proposition 1.3.18 *Let $G = (V, E)$ be a connected, undirected graph. Then the following inequality holds*

$$|V| \le 1 + B(G) \cdot D(G) \quad .$$

Proof. For a vertex $v \in V$ and $k \in \mathbb{N}$ let $B_k(v) := \{v' \in V \mid d(v, v') \le k\} \subseteq V$ denote the ball of radius k centered on v. Fix any $v \in V$. Then the following two observations hold:

[6] Vertices of degree strictly greater than 2.

1. $N(B_k(v)) = B_{k+1}(v) \setminus B_k(v)$ for $k = 0, \dots, D-1$.

2. $N(B_k(v)) \leq B(G)$ for $k = 0, \dots, D-1$ as $B_k(v)$ is connected for $k = 0, \dots, D-1$.

It follows:

$$V = B_D(v) = \{v\} \dot{\bigcup_{i=1}^{D(G)}} (B_i(v) \setminus B_{i-1}(v)) \overset{1.}{=} \{v\} \cup \dot{\bigcup_{i=1}^{D(G)}} N(B_{i-1}(v)) \ .$$

Thus

$$|V| = 1 + \sum_{i=1}^{D(G)} |N(B_{i-1}(v))| \overset{2.}{\leq} 1 + \sum_{i=1}^{D(G)} B(G) = 1 + D(G) \cdot B(G) \ .$$

■

We instantly obtain our first result:

Corollary 1.3.19 *Let $\{G_n\}_{n\in\mathbb{N}} = (V_n, E_n)_{n\in\mathbb{N}}$ be a sequence of connected B-good graphs. Then the diameter of G_n grows linearly in $|V_n|$:*

$$D(G_n) = \Theta(|V_n|) \ .$$

The proof follows from Proposition 1.3.18 and the trivial upper bound $D(G_n) \leq |V_n|$.

We conclude by showing that we cannot expect random graphs to be B-good.

Definition 1.3.20 (Erdős–Rényi graph). Let $p \colon \mathbb{N} \to [0,1]$. For $n \in \mathbb{N}$ let $V = \{v_1, \dots, v_n\}$ be a set of vertices. For each $1 \leq i < j \leq n$ add an undirected edge between v_i and v_j with probability $p(n)$. We let $G_{n,p(n)} := (V, E)$ denote the obtained graph (as a random variable). We call $G_{n,p(n)}$ *Erdős–Rényi graph.*

Recall that we are solely interested in the structure of *connected* graphs. Thus, we can easily derive a lower bound on $p(n)$ as it ought to be chosen such that we can expect $G_{n,p(n)}$ to have at least $n-1$ undirected edges; a necessary condition for being connected. As there are $\binom{n}{2}$ pairs of distinct vertices, each of the pairs being connected with probability $p(n)$ we have to have $\mathbb{E}(|E|) = \binom{n}{2} \cdot p(n) \geq n-1$ implying that $p(n) = \Omega(n^{-1})$.

Proposition 1.3.21 *Let $p(n) = \Omega(n^{-1})$, $n \in \mathbb{N}$ and $p := p(n)$. Let $G_{n,p} = (V, E)$ denote a corresponding Erdős–Rényi graph. Let V' be an arbitrary subset of V of size $n/2$. Then*

$$\mathbb{E}\left[N(V')\right] = \Theta(n) \ .$$

Proof. Clearly, $\mathbb{E}[N(V')] \leq n$ and it suffices to show that $\mathbb{E}[N(V')] = \Omega(n)$. For the neighborhood of V' we have $N(V') = \sum_{v \in V \setminus V'} \mathbb{1}_{\substack{\exists v' \in V' \\ (v,v') \in E}}$. It follows

$$\begin{aligned}
\mathbb{E}[N(V')] &= \sum_{v \in V \setminus V'} \mathbb{E}[\mathbb{1}_{\substack{\exists v' \in V' \\ (v,v') \in E}}] \\
&= \sum_{v \in V \setminus V'} \Pr\left[\exists v' \in V' \text{ s.t. } (v,v') \in E\right] \\
&= \sum_{v \in V \setminus V'} \left(1 - \Pr\left[\forall v' \in V' \text{ s.t. } (v,v') \notin E\right]\right) \\
&= \sum_{v \in V \setminus V'} \left(1 - \Pr\left[\bigwedge_{v' \in V'} (v,v') \notin E\right]\right) \\
&= \sum_{v \in V \setminus V'} \left(1 - \prod_{v' \in V'} (1-p)\right) \\
&= \frac{n}{2} \cdot \left(1 - (1-p)^{n/2}\right)
\end{aligned}$$

where all probabilities are taken over the coins in the generation of $G_{n,p}$. As $p = \Omega(n^{-1})$ we conclude that for some constant $c > 0$ and for sufficiently large n we have

$$\mathbb{E}[N(V')] \geq \frac{n}{2} \cdot \left(1 - \left(1 - \frac{c}{n}\right)^{n/2}\right) \quad .$$

It remains to show that $\lim_{n \to \infty} \mathbb{E}[N(V')] \cdot n^{-1} > 0$. It suffices to show

$$\lim_{n \to \infty} \left(1 - \left(1 - \frac{c}{n}\right)^{n/2}\right) \stackrel{!}{>} 0 \quad .$$

which clearly holds as $\lim_{n \to \infty} \left(1 - \frac{c}{n}\right)^{n/2} = \exp(-c/2)$. ∎

1.4 Results for Decomposing Distributions

In this section we extend our results to distributions $\mathfrak{D}$ over plaintext spaces $\mathcal{M}$ that decompose into multiple independent distributions $\mathfrak{D} \simeq \mathfrak{D}_1 \times \ldots \times \mathfrak{D}_{n'}$, $n' \leq n$ over batches $\mathcal{M}_1, \ldots, \mathcal{M}_{n'}$, where $\mathcal{M} = \mathcal{M}_1 \times \ldots \times \mathcal{M}_{n'}$ and for all $i \in [n']$ we have that $\mathfrak{D}_i$ is a distribution on $\mathcal{M}_i$. If a PKE scheme ensures SO security for all distributions $\mathfrak{D}_i$, we can lift the security to $\mathfrak{D}$ by a fairly straight-forward hybrid argument. Importantly, our results subsume the early positive result of [DNRS99, BY09] assuming all plaintexts to be independently distributed. In fact Section 1.4 does not only extend our results but all results to distributions as described above.

Definition 1.4.1 (Decomposable Distribution). Let $\{\mathfrak{D}_n\}_{n\in\mathbb{N}}$ be a sequence of distributions such that for all $n \in \mathbb{N}$, $\mathfrak{D}_n$ is a distribution over some plaintext space $\mathcal{M}^n$.

We say $\{\mathfrak{D}_n\}_{n\in\mathbb{N}}$ is *decomposable* if one can efficiently find $n' \colon \mathbb{N}^{>0} \to \mathbb{N}^{>0}$, $n' := n'(n)$, $\mu_i \colon \mathbb{N} \to \mathbb{N}$ and distributions $\mathfrak{D}_{\mu_i(n)}$ over $\mathcal{M}^{\mu_i(n)}$ for $i = 1, \ldots, n'$, such that for all $n \in \mathbb{N}$:

$$\mathfrak{D}_n \simeq \mathfrak{D}_{\mu_1(n)} \times \ldots \times \mathfrak{D}_{\mu_{n'}(n)} \quad .$$

That is, for all $n \in \mathbb{N}$ one can efficiently write $\mathfrak{D}_n$ as a product of distributions $\mathfrak{D}_{\mu_i(n)}$ over $\mathcal{M}^{\mu_i(n)}$ and $\sum_{i=1}^{n'} \mu_i(n) = n$.

We wrap the process of decomposing a distribution into an algorithm Decomp.

We may write a sequence of decomposable distributions $\{\mathfrak{D}_n\}_{n\in\mathbb{N}}$ as the product of sequences of distributions $\{\mathfrak{D}_{\mu_1(n)}\}_{n\in\mathbb{N}} \times \ldots \times \{\mathfrak{D}_{\mu_{n'}(n)}\}_{n\in\mathbb{N}}$.

Observation 1.4.2 A decomposable sequence $\{\mathfrak{D}_n\}_{n\in\mathbb{N}}$ is efficiently resampleable iff all sequences $\{\mathfrak{D}_{\mu_i(n)}\}_{n\in\mathbb{N}}$ are efficiently resampleable.

Further, if a sequence of distributions $\{\mathfrak{D}_n\}_{n\in\mathbb{N}}$ is decomposable with $n'(n) = n$ for all n, then $\{\mathfrak{D}_n\}_{n\in\mathbb{N}}$ is efficiently resampleable.

This is due to the fact that for all $n \in \mathbb{N}$, $\mathfrak{D}_n$ can be written as product of n distributions $\mathfrak{D}_n \simeq \mathfrak{D}_{\mu_1(n)} \times \ldots \times \mathfrak{D}_{\mu_n(n)}$ for $\mu_i(n) \equiv 1$ for all $n \in \mathbb{N}$. Thus resampling of positions $i \in [n] \setminus \mathcal{I}$ can be done by sampling from distributions $\mathfrak{D}_{\mu_i(n)}$ for all $i \in [n] \setminus \mathcal{I}$.

For notational brevity we write $\{\mu_j\}$ for the set $\left[1 + \sum_{i=1}^{j-1} \mu_j, 1 + \sum_{i=1}^{j} \mu_j\right]$. Hence, we may write n-dimensional vectors $\mathbf{v}$ as $\mathbf{v} = (\mathbf{v}_{\{\mu_1\}}, \ldots, \mathbf{v}_{\{\mu_{n'}\}}) \in S^n$ where for $i = 1, \ldots, n'$, $\mathbf{v}_{\{\mu_i\}}$ is of dimension μ_i.

Example 1.4.3 As a toy example consider the sequence of uniform distributions over $\mathcal{M}^n$: $\{U_n\}_{n\in\mathbb{N}}$. Clearly, $\{U_n\}_{n\in\mathbb{N}}$, for $n'(n) := n$, can be decomposed into a product of n distributions $\{U_1\}_{n\in\mathbb{N}} \times \ldots \times \{U_1\}_{n\in\mathbb{N}}$ each over $\mathcal{M}$.

Alternatively, let $n' \colon \mathbb{N} \to \mathbb{N}$ be arbitrary and let $\mu_1(n), \ldots, \mu_{n'}(n)$ be such that for all $n \in \mathbb{N}$: $\sum_{i=1}^{n'} \mu_i(n) = n$. Then $\{U_n\}_{n\in\mathbb{N}}$ can be decomposed into distributions $\{U_{\mu_1(n)}\}_{n\in\mathbb{N}} \times \ldots \times \{U_{\mu_{n'}(n)}\}_{n\in\mathbb{N}}$.

Theorem 1.4.4 *Let $\{\mathfrak{D}_n\}_{n\in\mathbb{N}}$ be a decomposable, efficiently resampleable sequence of distributions. If for all $i \in [n']$ scheme* PKE *is $(\tau_{so,i}, \varepsilon_{so,i})$-IND-SO-CPA secure w.r.t. $\{\mathfrak{D}_{\mu_i(n)}\}_{n\in\mathbb{N}}$, then* PKE *is $(\tau_{so}, \varepsilon_{so})$-IND-SO-CPA secure w.r.t. $\{\mathfrak{D}_n\}_{n\in\mathbb{N}}$ where*

$$\tau_{so} \leq \min_{i\in[n']}\{\tau_{so,i}\} - \tau_{resamp} - \tau_{decomp} \ , \qquad \varepsilon_{so}(n) \leq \sum_{i=1}^{n'} \varepsilon_{so,i}(n) \ .$$

Here τ_{resamp} is the time of one execution of the resampling algorithm and τ_{decomp} is the time of one execution of the decomposition algorithm.

Proof of Theorem 1.4.4. As already mentioned the proof follows from a straight-forward hybrid argument. The hybrid experiments are given in Figure 1.10. In the hybrid step from H_k to H_{k+1} plaintexts coming from distribution $\mathfrak{D}_{\mu_{k+1}}$ (we drop n as it is already fixed) are replaced by resampled plaintexts. We employ the IND-SO-CPA security w.r.t. $\mathfrak{D}_{\mu_{k+1}}$ to bound the distance between hybrid experiments H_k and H_{k+1}. Let $\mathcal{A}_{so} = (\mathcal{A}_{so,1}, \mathcal{A}_{so,2}, \mathcal{A}_{so,3})$ be an adversary against the $(\tau_{so}, \varepsilon_{so})$-IND-SO-CPA security of PKE w.r.t. $\mathfrak{D}$.

Exp $\mathsf{H}_k^{\mathcal{A}_{so}}(n)$
$\mathcal{I} \leftarrow \emptyset$
$(pk, sk) \leftarrow_\$ \mathsf{PKE.Gen}$
$(\mathfrak{D}, st_1) \leftarrow_\$ \mathcal{A}_{so,1}(pk, n)$
$(\mathfrak{D}_{\mu_1}, \dots, \mathfrak{D}_{\mu_{n'}}) \leftarrow \mathsf{Decomp}(\mathfrak{D})$
For $i \leftarrow 1$ to n':
 $\mathbf{m}^0_{\{\mu_i\}} \leftarrow_\$ \mathfrak{D}_{\mu_i}$
 $\mathbf{r}_{\{\mu_i\}} \leftarrow_\$ \mathcal{R}^{\mu_i}$
 $\mathbf{c}_{\{\mu_i\}} \leftarrow \mathsf{PKE.Enc}_{pk}(\mathbf{m}^0_{\{\mu_i\}}; \mathbf{r}_{\{\mu_i\}})$
$\mathbf{c} \leftarrow (\mathbf{c}_{\{\mu_1\}}, \dots, \mathbf{c}_{\{\mu_{n'}\}})$
$st_2 \leftarrow_\$ \mathcal{A}_{so,2}^{\textsc{Open}}(st_1, \mathbf{c})$
For all $i \leftarrow 1$ to k:
 $\mathbf{m}^1_{\{\mu_i\}} \leftarrow_\$ \mathsf{Resamp}_{\mathfrak{D}_{\mu_i}}(\mathbf{m}^0_{\{\mu_i\}}, \mathcal{I} \cap \{\mu_i\})$
$\mathbf{m} \leftarrow (\mathbf{m}^1_{\{\mu_1\}}, \dots, \mathbf{m}^1_{\{\mu_k\}}, \mathbf{m}^0_{\{\mu_{k+1}\}}, \dots, \mathbf{m}^0_{\{\mu_{n'}\}})$
$b' \leftarrow_\$ \mathcal{A}_{so,3}(st_2, \mathbf{m})$
Stop with b'

Oracle $\textsc{Open}(i)$
$\mathcal{I} \leftarrow \mathcal{I} \cup \{i\}$
Return (m_i^0, r_i)

Figure 1.10: Hybrid experiments $\mathsf{H}_k(n)$ used in the proof of Theorem 1.4.4.

Note that H_0 is identical to the $\mathsf{IND\text{-}SO\text{-}CPA}_0$ experiment and $\mathsf{H}_{n'}$ is identical to the $\mathsf{IND\text{-}SO\text{-}CPA}_1$ experiment (see Figure 1.2). Thus

$$\begin{aligned} &\left|\Pr\left[\mathsf{IND\text{-}SO\text{-}CPA}_0^{\mathcal{A}}(n) \Rightarrow 1\right] - \Pr\left[\mathsf{IND\text{-}SO\text{-}CPA}_1^{\mathcal{A}}(n) \Rightarrow 1\right]\right| \\ =&\left|\Pr\left[\mathsf{H}_0^{\mathcal{A}_{so}}(n) \Rightarrow 1\right] - \Pr\left[\mathsf{H}_{n'}^{\mathcal{A}_{so}}(n) \Rightarrow 1\right]\right| \\ \leq& \sum_{k=0}^{n'-1} \left|\Pr\left[\mathsf{H}_k^{\mathcal{A}_{so}}(n) \Rightarrow 1\right] - \Pr\left[\mathsf{H}_{k+1}^{\mathcal{A}_{so}}(n) \Rightarrow 1\right]\right| \quad . \end{aligned} \tag{1.5}$$

We proceed with the following Lemma.

Lemma 1.4.5 *For each $k \in \{1, \ldots, n'\}$ there exists $\mathcal{A}_{so,k} = (\mathcal{A}_{so,k,1}, \mathcal{A}_{so,k,2}, \mathcal{A}_{so,k,3})$ and $n_{so} \in \mathbb{N}$ that $(\tau_{so,k}, \varepsilon_{so,k})$-breaks the IND-SO-CPA security of* PKE *w.r.t. to $\mathfrak{D}_{\mu_k}$ where*

$$\tau_{so,k} \approx \tau_{so} + \tau_{resamp} + \tau_{decomp} \ , \qquad \varepsilon_{so,k} \geq \left| \Pr\left[\mathsf{H}_{k-1}^{\mathcal{A}_{so}}(n) \Rightarrow 1\right] - \Pr\left[\mathsf{H}_{k}^{\mathcal{A}_{so}}(n) \Rightarrow 1\right]\right| \ ,$$

and τ_{resamp} is the time of one execution of the resampling algorithm and τ_{decomp} is the time of one execution of the decomposition algorithm.

Proof of Lemma 1.4.5. We describe adversary $\mathcal{A}_{so,k+1}$ as given in Figure 1.11 interpolating between hybrids H_k and H_{k+1} for $\mathcal{A}_{so}$. The value of $n_{so} \in \mathbb{N}$ follows from the proof.

$\mathcal{A}_{so,k+1,1}$ relays (pk, n) to $\mathcal{A}_{so}$. Receiving $\mathfrak{D}$, the distribution is decomposed into $(\mathfrak{D}_{\mu_1}, \ldots, \mathfrak{D}_{\mu_{n'}})$ (see line 02). $\mathcal{A}_{so,k+1,1}$ outputs $\mathfrak{D}_{\mu_{k+1}}$ and halts.

$\mathcal{A}_{so,k+1,2}(\mathbf{c}_{\{\mu_{k+1}\}})$ simulates the IND-SO-CPA experiment on all $\mathbf{m}_{[n]\setminus\{\mu_{k+1}\}}$ on its own (lines 04-07) and invokes $\mathcal{A}_{so,2}$ on ciphertexts $(\mathbf{c}_{\{\mu_1\}}, \ldots, \mathbf{c}_{\{\mu_{n'}\}})$ (line 07).

$\mathcal{A}_{so,k+1,2}$ answers opening queries on its own unless they occur on $\{\mu_{k+1}\}$, where it invokes its own opening oracle $\textsc{Open}_{so}$ to answer. Once $\mathcal{A}_{so,2}$ terminates, so does $\mathcal{A}_{so,k+1,2}$.

$\mathcal{A}_{so,k+1,3}(\mathbf{m}_{\{\mu_{k+1}\}})$ resamples plaintexts at positions $\cup_{i\in[k]}\{\mu_i\}$ on its own (line 12), embeds its challenge at positions $\{\mu_{k+1}\}$ and keeps the sampled plaintexts $\mathbf{m}^0$ at all remaining positions. It invokes $\mathcal{A}_{so,3}$ on a vector

$$(\mathbf{m}^1_{\{\mu_1\}}, \ldots, \mathbf{m}^1_{\{\mu_k\}}, \mathbf{m}_{\{\mu_{k+1}\}}, \mathbf{m}^0_{\{\mu_{k+2}\}}, \ldots, \mathbf{m}^0_{\{\mu_{n'}\}})$$

and replay $\mathcal{A}_{so,3}$'s output to its experiment.

Analysis One easily verifies that $\mathcal{A}_{so,k+1}$ correctly simulates hybrid experiments H_k and H_{k+1} at all positions until $\mathcal{A}_{so,2}$ halts.

Now, when $\mathcal{A}_{so,k+1}$ is run in experiment $\mathsf{IND\text{-}SO\text{-}CPA}_0$, adversary $\mathcal{A}_{so,k+1,3}$ obtains sampled plaintexts $\mathbf{m}_{\{\mu_{k+1}\}}$ thereby simulating $\mathcal{A}_{so}$'s view as in H_k. If $\mathcal{A}_{so,k+1,3}$ receives resampled plaintexts at positions $\{\mu_{k+1}\}$, adversary $\mathcal{A}_{so}$ is run in experiment H_{k+1}. Hence

$$\begin{aligned} &\Pr\left[\mathsf{IND\text{-}SO\text{-}CPA}_0^{\mathcal{A}_{so,k+1}} \Rightarrow 1\right] = \Pr\left[\mathsf{H}_k^{\mathcal{A}_{so}}(n) \Rightarrow 1\right] \\ \text{and} \quad &\Pr\left[\mathsf{IND\text{-}SO\text{-}CPA}_1^{\mathcal{A}_{so,k+1}} \Rightarrow 1\right] = \Pr\left[\mathsf{H}_{k+1}^{\mathcal{A}_{so}}(n) \Rightarrow 1\right] \ . \end{aligned}$$

The running time of $\mathcal{A}_{so,k+1}$ is roughly the same as the running time of $\mathcal{A}_{so}$ except for one additional call of Resamp and Decomp. $\mathcal{A}_{so,k+1}$ submits a distribution over $\mathcal{M}^{\mu_{k+1}}$,

Adversary $\mathcal{A}_{so,k+1,1}(pk, n)$
01 $(\mathfrak{D}) \leftarrow_\$ \mathcal{A}_{so,1}(pk, n)$
02 $(\mathfrak{D}_{\mu_1}, \ldots, \mathfrak{D}_{\mu_{n'}}) \leftarrow \mathsf{Decomp}(\mathfrak{D})$
03 Output $\mathfrak{D}_{\mu_{k+1}}$

Adversary $\mathcal{A}_{so,k+1,2}(\mathbf{c}_{\{\mu_{k+1}\}})$
04 For all $i \in [n'] \setminus \{k+1\}$:
05 $\quad \mathbf{m}^0_{\{\mu_i\}} \leftarrow_\$ \mathfrak{D}_{\mu_i}$
06 $\quad \mathbf{r}_{\{\mu_i\}} \leftarrow_\$ \mathcal{R}^{\mu_i}$
07 $\quad \mathbf{c}_{\{\mu_i\}} \leftarrow \mathsf{PKE.Enc}_{pk}(\mathbf{m}^0_{\{\mu_i\}}; \mathbf{r}_{\{\mu_i\}})$
08 $\mathbf{c} \leftarrow (\mathbf{c}_{\{\mu_1\}}, \ldots, \mathbf{c}_{\{\mu_{n'}\}})$
09 $() \leftarrow_\$ \mathcal{A}^{\textsc{Open}}_{so,2}(\mathbf{c})$
10 Output $()$

Adversary $\mathcal{A}_{so,k+1,3}(\mathbf{m}_{\{\mu_{k+1}\}})$
11 For all $i \in [k]$:
12 $\quad \mathbf{m}^1_{\{\mu_i\}} \leftarrow \mathsf{Resamp}_{\mathfrak{D}_{\mu_i}}(\mathbf{m}^0_{\{\mu_i\}}, \mathcal{I} \cap \{\mu_i\})$
13 $\mathbf{m} \leftarrow (\mathbf{m}^1_{\{\mu_1\}}, \ldots, \mathbf{m}^1_{\{\mu_k\}}, \mathbf{m}_{\{\mu_{k+1}\}}, \mathbf{m}^0_{\{\mu_{k+2}\}}, \ldots, \mathbf{m}^0_{\{\mu_{n'}\}})$
14 $b' \leftarrow_\$ \mathcal{A}_{so,3}(\mathbf{m})$
15 Output b'

Oracle $\textsc{Open}(i)$
16 $\mathcal{I} \leftarrow \mathcal{I} \cup \{i\}$
17 If $i \in \mu_{k+1}$:
18 $\quad (m^0_i, r_i) \leftarrow \textsc{Open}_{so}(i)$
19 Return (m^0_i, r_i)

Figure 1.11: Pseudocode of adversary $\mathcal{A}_{so,k+1} = (\mathcal{A}_{so,k+1,1}, \mathcal{A}_{so,k+1,2}, \mathcal{A}_{so,k+1,3})$ run in the IND-SO-CPA experiment (w.r.t. $\mathfrak{D}_{\mu_{k+1}}$). $\mathcal{A}_{so,k+1}$ interpolates between hybrids H_k, H_{k+1} for $\mathcal{A}_{so}$. We abstain from making the states output by and returned to $\mathcal{A}_{so,k+1}$ and $\mathcal{A}_{so}$ explicit. $\textsc{Open}_{so}$ denotes the opening oracle provided by the IND-SO-CPA for $\mathcal{A}_{so,k+1}$ (line 18) while $\textsc{Open}$ denotes the opening oracle provided by $\mathcal{A}_{so,k+1}$ for $\mathcal{A}_{so}$.

thus $\mathcal{A}_{so,k+1}$ breaks the IND-SO-CPA security for $n_{so} := \mu_{k+1}$. Lemma 1.4.5 follows. ■

We proceed from Equation (1.5):

$$\begin{aligned}
\Big|\Pr\Big[\mathsf{IND\text{-}SO\text{-}CPA}_0^{\mathcal{A}_{so}}(n) \Rightarrow 1\Big] &- \Pr\Big[\mathsf{IND\text{-}SO\text{-}CPA}_1^{\mathcal{A}_{so}}(n) \Rightarrow 1\Big]\Big| \\
&= \Big|\Pr\Big[\mathsf{H}_0^{\mathcal{A}_{so}}(n) \Rightarrow 1\Big] - \Pr\Big[\mathsf{H}_{n'}^{\mathcal{A}_{so}}(n) \Rightarrow 1\Big]\Big| \\
&\leq \sum_{k=0}^{n'-1} \Big|\Pr\Big[\mathsf{H}_k^{\mathcal{A}_{so}}(n) \Rightarrow 1\Big] - \Pr\Big[\mathsf{H}_{k+1}^{\mathcal{A}_{so}}(n) \Rightarrow 1\Big]\Big| \\
&= \sum_{k=1}^{n'} \Big|\Pr\Big[\mathsf{H}_{k-1}^{\mathcal{A}_{so}}(n) \Rightarrow 1\Big] - \Pr\Big[\mathsf{H}_k^{\mathcal{A}_{so}}(n) \Rightarrow 1\Big]\Big| \\
&\leq \sum_{k=1}^{n'} \varepsilon_{so,k}
\end{aligned}$$

to conclude the proof of Theorem 1.4.4. ■

1.5 Extending all Results to Active Attacks

We conclude with a rather short section. We show that all results established for the relation amongst IND-CPA and IND-SO-CPA security can be lifted to hold between IND-CCA and IND-SO-CCA security as well. We begin by defining standard and selective opening security under active attacks.

1.5.1 Security Notions under Active Attacks

Definition 1.5.1 (IND-CCA secure PKE). For $\varepsilon \in \mathbb{R}^{\geq 0}$, $q_d \in \mathbb{N}$ we say that PKE is (τ, q_d, ε)-*IND-CCA secure* if for all τ-time adversaries $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ that interact in the $\mathsf{IND\text{-}CCA}_b$ experiments as given in Figure 1.12 and query the PKE.DEC oracle at most q_d times we have

$$\left|\Pr\left[\mathsf{IND\text{-}CCA}_0^{\mathcal{A}} \Rightarrow 1\right] - \Pr\left[\mathsf{IND\text{-}CCA}_1^{\mathcal{A}} \Rightarrow 1\right]\right| \leq \varepsilon \ .$$

Exp $\mathsf{IND\text{-}CCA}_b^{\mathcal{A}}$	**Oracle** $\text{PKE.D}\textsc{ec}(c)$
$(pk, sk) \leftarrow_\$ \mathsf{PKE.Gen}$	If $c = c^*$: Abort
$(m^0, m^1, st) \leftarrow_\$ \mathcal{A}_1^{\text{PKE.D}\textsc{ec}}(pk, n)$	$m \leftarrow \mathsf{PKE.Dec}_{sk}(c)$
$c^* \leftarrow_\$ \mathsf{PKE.Enc}_{pk}(m^b)$	Return m
$b' \leftarrow_\$ \mathcal{A}_2^{\text{PKE.D}\textsc{ec}}(st, c)$	
Return b'	

Figure 1.12: The $\mathsf{IND\text{-}CCA}_b$ experiments as used in Definition 1.5.1.

In informal discussions we say that a scheme is *IND-CCA secure* if for all efficient adversaries ε is small. We abstain from giving the formal definition of mult-IND-CCA security that allows an adversary to submit plaintext vectors instead of single plaintexts. The reader may assemble the mult-IND-CCA experiment by considering Figure 1.1 and adding the decryption oracle from Figure 1.12 to it. We have a classical result similar to Lemma 1.2.3:

Lemma 1.5.2 *Let* PKE *be a* $(\tau_{cca}, q_d, \varepsilon_{cca})$*-IND-CCA secure PKE. Then* PKE *is* $(\tau_{m\text{-}cca}, q_d, \varepsilon_{m\text{-}cca})$*-mult-IND-CCA secure where*

$$\tau_{m\text{-}cca} \approx \tau_{cca} \ , \qquad \varepsilon_{m\text{-}cca}(n) \leq n \cdot \varepsilon_{cca} \ .$$

Definition 1.5.3 (IND-SO-CCA secure PKE). Let $\mathcal{D}$ be a subset of the class of sequences of efficiently resampleable distributions as in Definition 1.2.4. For a function $\varepsilon\colon \mathbb{N} \to \mathbb{R}^{\geq 0}$ and $q_d \in \mathbb{N}$ we say that PKE is (τ, q_d, ε)-*IND-SO-CCA secure with respect to* $\mathcal{D}$ if for all τ-time adversaries $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3)$ that interact in the $\mathsf{IND\text{-}SO\text{-}CCA}_b$ experiments as given in Figure 1.13 and query the PKE.DEC oracle at most q_d times and all $n \in \mathbb{N}$, we have

$$\left|\Pr\left[\mathsf{IND\text{-}SO\text{-}CCA}_0^{\mathcal{A}}(n) \Rightarrow 1\right] - \Pr\left[\mathsf{IND\text{-}SO\text{-}CCA}_1^{\mathcal{A}}(n) \Rightarrow 1\right]\right| \leq \varepsilon(n) \ .$$

Exp $\mathsf{IND\text{-}SO\text{-}CCA}_b^{\mathcal{A}}$
$\mathcal{I} \leftarrow \emptyset$; $\mathbf{c} \leftarrow \emptyset$
$(pk, sk) \leftarrow_\$ \mathsf{PKE.Gen}$
$(\mathfrak{D}, st) \leftarrow_\$ \mathcal{A}_1^{\text{PKE.DEC}}(pk, n)$
$\mathbf{m}^0 \leftarrow_\$ \mathfrak{D}$
$\mathbf{r} \leftarrow_\$ \mathcal{R}^n$
$\mathbf{c} \leftarrow \mathsf{PKE.Enc}_{pk}(\mathbf{m}^0; \mathbf{r})$
$st' \leftarrow_\$ \mathcal{A}_2^{\text{OPEN},\text{PKE.DEC}}(st, \mathbf{c})$
$\mathbf{m}^1 \leftarrow_\$ \mathsf{Resamp}_{\mathfrak{D}}(\mathbf{m}^0, \mathcal{I})$
$b' \leftarrow_\$ \mathcal{A}_3^{\text{PKE.DEC}}(st', \mathbf{m}^b)$
Stop with b'

Oracle OPEN(i)
$\mathcal{I} \leftarrow \mathcal{I} \cup \{i\}$
Return (m_i^0, r_i)

Oracle PKE.DEC(c)
If $c \in \mathbf{c}$: Abort
$m \leftarrow \mathsf{PKE.Dec}_{sk}(c)$
Return m

Figure 1.13: Security experiments $\mathsf{IND\text{-}SO\text{-}CCA}_b$ used in Definition 1.5.3. We require $\mathcal{A}_1$ to output $\mathfrak{D}$ such that $\mathfrak{D} \in \mathcal{D}$ and $\mathfrak{D}$ is a distribution over $\mathcal{M}^n$. $\mathcal{A}_2$ may call OPEN(i) for $i \in [n]$.

If $\mathcal{D}$ is the class of *all* sequences of efficiently resampleable distributions, we say that PKE is (τ, q_d, ε)-*IND-SO-CCA secure*. In informal statements we say that a PKE scheme is *IND-SO-CCA secure*, if for all efficient adversaries and all $n \in \mathbb{N}$, we have that ε is small.

1.5.2 Results for Active Attacks

We state our main results on the relations between IND-CCA and IND-SO-CCA security. These are essentially the same results established between IND-CPA and IND-SO-CPA security, adapted to the CCA setting. A proof strategy is sketched at the end of this section.

Theorem 1.5.4 *Let $\mathcal{D}$ be the class of efficiently resampleable sequences of distributions induced by sequences of connected graphs $\{G_n\}_{n\in\mathbb{N}}$.*

If PKE *is $(\tau_{cca}, q_d, \varepsilon_{cca})$-IND-CCA secure, then* PKE *is $(\tau_{\text{so-cca}}, q_d, \varepsilon_{\text{so-cca}})$-IND-SO-CCA secure where*

$$\tau_{\text{so-cca}} \leq \tau_{cca} - 2 \cdot \tau_{resamp} \ , \qquad \varepsilon_{\text{so-cca}}(n) \leq n \cdot (n-1) \cdot S(G_n) \cdot \varepsilon_{cca}$$

where τ_{resamp} is the time of one execution of the resampling algorithm.

Theorem 1.5.5 *Let $\mathcal{D}$ be the class of efficiently resampleable sequences of distributions induced by sequences of connected graphs $\{G_n\}_{n\in\mathbb{N}}$.*

If PKE *is $(\tau_{cca}, q_d, \varepsilon_{cca})$-IND-CCA secure, then* PKE *is $(\tau_{\text{so-cca}}, q_d, \varepsilon_{\text{so-cca}})$-IND-SO-CCA secure where*

$$\tau_{\text{so-cca}} \leq \tau_{cca} - 2 \cdot \tau_{resamp} \ , \qquad \varepsilon_{\text{so-cca}}(n) \leq \frac{2 \cdot (n-1)}{(B(G_n)-1)!} \cdot n^{B(G_n)+1} \cdot \varepsilon_{cca}$$

where τ_{resamp} is the time of one execution of the resampling algorithm.

Theorem 1.5.6 *Let $\mathcal{D}$ be the class of efficiently resampleable sequences of distributions induced by sequences of connected DAGs.*

If PKE *is $(\tau_{cca}, q_d, \varepsilon_{cca})$-IND-CCA secure, then* PKE *is $(\tau_{\text{so-cca}}, q_d, \varepsilon_{\text{so-cca}})$-IND-SO-CCA secure where*

$$\tau_{\text{so-cca}} \leq \tau_{cca} - 3 \cdot \tau_{resamp} \ , \qquad \varepsilon_{\text{so-cca}}(n) \leq 3 \cdot n^{B(G_n)+1} \cdot \varepsilon_{cca}$$

where τ_{resamp} is the time of one execution of the resampling algorithm.

Corollary 1.5.7 *If a PKE scheme* PKE *is $(\tau_{cca}, q_d, \varepsilon_{cca})$-IND-CCA secure, then* PKE *is $(\tau_{\text{so-cca}}, \varepsilon_{\text{so-cca}})$-IND-SO-CCA secure w.r.t efficiently resampleable Markov distributions where*

$$\tau_{\text{so-cca}} \leq \tau_{cca} - 3 \cdot \tau_{resamp} \ , \qquad \varepsilon_{\text{so-cca}}(n) \leq 3 \cdot n^3 \cdot \varepsilon_{cca}$$

where τ_{resamp} is the time of one execution of the resampling algorithm.

Theorem 1.5.8 *Let $\{\mathfrak{D}_n\}_{n\in\mathbb{N}}$ be a decomposable, efficiently resampleable sequence of distributions. If for all $i \in [n']$ scheme* PKE *is $(\tau_{\text{so-cca},i}, q_{d,\text{so-cca}}, \varepsilon_{\text{so-cca},i})$-IND-SO-CCA secure w.r.t. $\{\mathfrak{D}_{\mu_i(n)}\}_{n\in\mathbb{N}}$, then* PKE *is $(\tau_{\text{so-cca}}, q_{d,\text{so-cca}}, \varepsilon_{\text{so-cca}})$-IND-SO-CCA secure w.r.t. $\{\mathfrak{D}_n\}_{n\in\mathbb{N}}$ where*

$$\tau_{\text{so-cca}} \leq \min_{i\in[n']}\{\tau_{\text{so-cca},i}\} - \tau_{resamp} - \tau_{decomp} \ , \qquad \varepsilon_{\text{so-cca}}(n) \leq \sum_{i=1}^{n'} \varepsilon_{\text{so-cca},i}(n) \ .$$

Here τ_{resamp} is the time of one execution of the resampling algorithm and τ_{decomp} is the time of one execution of the decomposition algorithm.

The proofs of Theorems 1.5.4 to 1.5.6 and 1.5.8 and Corollary 1.5.7 immediately follow from the proofs in Section 1.3. To this end, one observes that whenever a reduction from IND-SO-CCA to (mult)-IND-CCA security submits a decryption query leading to an abort, the same query posed by an IND-SO-CCA attacker leads to an abort as well. Hence, whenever an $\mathcal{A}_{so\text{-}cca}$ attacker against the IND-SO-CCA security of PKE issues a decryption query, the reduction can relay the query to the (mult)-IND-CCA experiment.

Hence, the claims follow from the proofs of Theorems 1.3.7, 1.3.11 and 1.4.4 and Corollaries 1.3.10 and 1.3.13.

Part II

Results in Idealized Models of Computation

Chapter 2

Selective Opening Security via Generic Transformations

In this chapter we study three well-known transformations that are known to give rise to IND-CCA secure PKE schemes from (merely) one-way secure cryptographic primitives in the random oracle model. We show that, in fact, all transformations do construct SO-CCA secure PKE schemes. Surprisingly, we *do not* require stronger assumptions to establish our results than those used to prove IND-CCA security.

In Section 2.2 we discuss a transformation that can be instantiated with any key encapsulation mechanism that is one-way secure under plaintext-checking attacks. Most notably, it covers an instantiation of the widely-employed DHIES. Section 2.3 considers the OAEP padding. When followed by a trapdoor permutation it results in an IND-CCA PKE scheme. Eventually, in Section 2.4, we study the Fujisaki-Okamoto transformation that can be instantiated with any one-way secure PKE scheme.

2.1 Selective Opening Security under Active Attacks

We define the confidentiality notion of SIM-SO-CCA for PKE schemes. Our model builds on the works of [BHK12, FHKW10]. We discuss the details below.

Definition 2.1.1 (SIM-SO-CCA secure PKE). Consider the experiments from Figure 2.1. For a function $\varepsilon\colon \mathbb{N} \to \mathbb{R}^{\geq 0}$ we say that a PKE scheme is (τ, q_d, ε)-SIM-SO-CCA secure if for all τ-time adversaries $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ that interact in the (real) $\mathsf{r\text{-}SO\text{-}CCA}$ experiment and issue at most q_d decryption queries to PKE.Dec there exists a (roughly) τ-time simulator $\mathcal{S} = (\mathcal{S}_1, \mathcal{S}_2)$ that interacts in the (ideal) $\mathsf{i\text{-}SO\text{-}CCA}$ experiment such that for all efficient distinguishers $\mathsf{Pred}\colon \{0,1\}^* \to \{0,1\}$ and all $n \in \mathbb{N}$ we have

$$\left|\Pr\left[\mathsf{r\text{-}SO\text{-}CCA}^{\mathcal{A}}(n) \Rightarrow 1\right] - \Pr\left[\mathsf{i\text{-}SO\text{-}CCA}^{\mathcal{S}}(n) \Rightarrow 1\right]\right| \leq \varepsilon(n) \ .$$

In idealized models we specify an upper bound on the number of allowed queries to an ideal primitive. For instance, if a PKE scheme involves a hash function and we assume the hash function to be modeled as a random oracle, we consider $(\tau, q_d, q_h, \varepsilon)$-SIM-SO-CCA security where an adversary may query the hash function at most q_h times.

In informal discussions we say a PKE scheme is SIM-SO-CCA secure if for all efficient adversaries $\mathcal{A}$ in the r-SO-CCA experiment there exists an efficient simulator $\mathcal{S}$ in the i-SO-CCA experiment such that for all efficient distinguishers and all $n \in \mathbb{N}$ we have that ε is small. Note that $\mathcal{A}$ may submit any distribution to its real experiment. In particular, the definition does not suffer from the restriction to efficiently resampleable distributions as Definition 1.2.5 does.

Exp $\mathsf{r\text{-}SO\text{-}CCA}^{\mathcal{A}}(n)$
$\mathcal{I} \leftarrow \emptyset$; $\mathbf{c} \leftarrow \emptyset$
$(pk, sk) \leftarrow_{\$} \mathsf{PKE.Gen}$
$(\mathfrak{D}, st) \leftarrow_{\$} \mathcal{A}_1^{\text{PKE.DEC}}(pk, n)$
$\mathbf{m} \leftarrow_{\$} \mathfrak{D}$
$\mathbf{r} \leftarrow_{\$} \mathcal{R}^n$
$\mathbf{c} \leftarrow \mathsf{PKE.Enc}_{pk}(\mathbf{m}; \mathbf{r})$
$out \leftarrow_{\$} \mathcal{A}_2^{\text{OPEN},\text{PKE.DEC}}(st, \mathbf{c})$
Stop with $\mathsf{Pred}(\mathfrak{D}, \mathbf{m}, \mathcal{I}, out)$

Oracle OPEN(i)
$\mathcal{I} \leftarrow \mathcal{I} \cup \{i\}$
Return (m_i, r_i)

Oracle PKE.DEC(c)
If $c \in \mathbf{c}$: Abort
$m \leftarrow \mathsf{PKE.Dec}_{sk}(c)$
Return m

Exp $\mathsf{i\text{-}SO\text{-}CCA}^{\mathcal{S}}(n)$
$\mathcal{I} \leftarrow \emptyset$
$(\mathfrak{D}, st) \leftarrow_{\$} \mathcal{S}_1(n)$
$\mathbf{m} \leftarrow_{\$} \mathfrak{D}$
$out \leftarrow_{\$} \mathcal{S}_2^{\text{OPEN}}(st, |m_1|, \ldots, |m_n|)$
Stop with $\mathsf{Pred}(\mathfrak{D}, \mathbf{m}, \mathcal{I}, out)$

Oracle OPEN(i)
$\mathcal{I} \leftarrow \mathcal{I} \cup \{i\}$
Return m_i

Figure 2.1: Security experiments for defining SIM-SO-CCA security of PKE. With $\mathfrak{D}$ we denote a distribution over $\mathcal{M}^n$. The randomness space of PKE.Enc is denoted with $\mathcal{R}$. Oracle OPEN may be called for all $i \in [n]$. We show the lines of i-SO-CCA aligned to the ones of r-SO-CCA for easier comparison.

Discussion We give rationale on this formalization of SO security. The notion compares the information an adversary can deduce about a set of challenge plaintexts in two settings: a real setting (experiment r-SO-CCA) and an idealized setting (experiment i-SO-CCA). The real experiment starts with the generation of a key pair. The adversary receives the public key and specifies a plaintext distribution $\mathfrak{D}$. Plaintexts $m_1, \ldots, m_n$ are sampled according to this distribution and encrypted using fresh randomnesses $r_1, \ldots, r_n$. The ciphertexts are given to the adversary who derives some information out.

The adversary is supported by two oracles: one that decrypts arbitrary ciphertexts and one that opens ciphertexts by revealing the corresponding plaintext and the randomness used to encrypt it (to model sender corruption).

The ideal experiment is similar but with all the artifacts of public-key encryption removed: there is no key generation, no ciphertext generation, and no decryption oracle. Beyond that, the adversary (in this context called 'simulator') performs as above: it specifies a plaintext distribution, adaptively requests openings, and derives some information *out*.

Clearly, in the ideal setting the confidentiality of plaintexts from unopened ciphertexts is granted (only their lengths leak in line 17, but this is unavoidable for any practical PKE scheme and implicitly also happens in line 07). We thus deem a public-key encryption scheme secure under selective opening attacks if the adversary in the real setting cannot draw more conclusions about the plaintexts from unopened ciphertexts than can be drawn in the ideal setting. Formally, it is required that for every $\mathcal{A}$ for r-SO-CCA there exists a corresponding $\mathcal{S}$ for i-SO-CCA that derives the same information. This is tested by distinguisher Pred (outputting some *predicate*), which also takes further environmental information into account, for instance the recorded opening history $\mathcal{I}$. We proceed with some remarks on the model.

In prior works that give simulation-based definitions of SO security there does not seem to be consensus on the order of quantification of $\mathcal{S}$ and Pred. While most papers (see [HJKS15, LP15]) allow for the simulator to depend on the distinguishing predicate, the work of [BHK12] implicitly defines a stronger notion that requires the existence of a simulator that is universal. (Interestingly, many papers that exclusively consider the weaker notion actually *do* construct universal simulators.) We adopt the stronger notion and require the simulator to work for any distinguisher Pred.

Proving SIM-SO-CCA Security The goal is to show that for every adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ for the r-SO-CCA experiment there exists a simulator $\mathcal{S} = (\mathcal{S}_1, \mathcal{S}_2)$ for the i-SO-CCA experiment that deduces the same information.

We walk the reader through the design principles of our simulator. What we referred to as 'deduces the same information' above formally requires that the inputs $(\mathfrak{D}, \mathbf{m}, \mathcal{I}, out)$ of Pred invocations in the r-SO-CCA and i-SO-CCA experiments be similar. This is achieved by letting $\mathcal{S}$ simulate for $\mathcal{A}$ the environment of r-SO-CCA in a way such that: $\mathcal{S}_1$ forwards the distribution $\mathfrak{D}$ obtained from $\mathcal{A}_1$ without modification (this also ensures that the distributions of $m_1, \ldots, m_n$ match), $\mathcal{S}_2$ keeps the index sets $\mathcal{I}$ corresponding to $\mathcal{A}_2$'s and its own Open queries consistent (by forwarding the queries), and $\mathcal{S}_2$ forwards $\mathcal{A}_2$'s output *out* without modification.

Running $\mathcal{A}$ as a subroutine leads to useful results only if $\mathcal{A}$ is exposed to an r-SO-CCA-like environment. Effectively this means that $\mathcal{S}$ has to 'fill the blanks' of the i-SO-CCA experiment in Figure 2.1. Concretely this involves (a) generating and providing a public key for $\mathcal{A}_1$, (b) providing ciphertexts to $\mathcal{A}_2$ that correspond to plaintexts $m_1, \ldots, m_n$, (c) providing adequate randomness when processing opening queries by $\mathcal{A}_2$, and (d) handling decryption queries by $\mathcal{A}_1$ and $\mathcal{A}_2$.

We proceed with the first transformation.

2.2 Transformation from any OW-PCA secure KEM

The first construction we consider is a generalization of the 'Diffie-Hellman Integrated Encryption Scheme' (DHIES) [BR97, ABR01]. (DHIES or 'Hashed Elgamal Encryption' uses a MAC to make plain Elgamal [Gam84] IND-CCA secure in the ROM.)

This generic idea behind DHIES was formalized by Steinfeld *et al.* [SBZ02] who showed how to build an IND-CCA secure public-key encryption system from a key encapsulation mechanism (KEM) that is one-way under plaintext checking attacks (OW-PCA). The notion of OW-PCA is a comparatively weak notion of security in which the adversary's main task is to decapsulate a given encapsulation. In addition to the public key, the adversary has only access to an oracle that checks, given a tuple (k, c), whether c is an encapsulation of k.

This construction is IND-CCA secure in the random oracle model [SBZ02]. We show that it is furthermore SIM-SO-CCA secure in the random oracle model. We stress that our result generically holds for the entire construction and therefore for any concrete instantiation of it. Most importantly, it covers the well-known DHIES scheme (when instantiated with a one-time pad) that is contained in several public-key encryption standards like IEEE P1363a, SECG, and ISO 18033-2. DHIES is the de-facto standard for elliptic-curve encryption.

Provable Security of DHIES The IND-CCA security of DHIES in the random oracle model has been shown to be equivalent to the strong Diffie-Hellman (sDH) assumption [ABR01, SBZ02].

2.2.1 Key Encapsulation Mechanisms and Message Authentication Codes

Definition 2.2.1 (key encapsulation mechanism). A *key encapsulation mechanism* (KEM) for a finite key space $\mathcal{K}$ consists of a public-key space $\mathcal{PK}$, a secret-key space $\mathcal{SK}$, a ciphertext space $\mathcal{C}$, and a triple of efficient algorithms denoted as $\mathsf{KEM} = (\mathsf{KEM.Gen}, \mathsf{KEM.Enc}, \mathsf{KEM.Dec})$ of the form

$$\mathsf{KEM.Gen}\colon \to_\$ \mathcal{PK} \times \mathcal{SK} \quad \mathsf{KEM.Enc}\colon \mathcal{PK} \to_\$ \mathcal{K} \times \mathcal{C} \quad \mathsf{KEM.Dec}\colon \mathcal{SK} \times \mathcal{C} \to \mathcal{K} \cup \{\bot\} \ ,$$

where symbol '$\bot$' may be used to indicate errors. The randomness space of $\mathsf{KEM.Enc}$ is typically denoted with $\mathcal{R}$. Correctness requires that for all $(pk, sk) \in [\mathsf{KEM.Gen}]$, if $(k, c) \in [\mathsf{KEM.Enc}_{pk}]$ then $\mathsf{KEM.Dec}_{sk}(c) = k$.

KEM has *unique encapsulations* if for all $(pk, sk) \in [\mathsf{KEM.Gen}]$ and for all $c, c' \in \mathcal{C}$ we have $\mathsf{KEM.Dec}_{sk}(c) = \mathsf{KEM.Dec}_{sk}(c') \Rightarrow c = c'$.

In the following KEM will always denote a key encapsulation mechanism. For the results in this section it suffices to assume that the keys output by $\mathsf{KEM.Enc}_{pk}$ have high min-entropy given pk.[1] However, for simplicity, we assume that for all $(pk, sk) \in [\mathsf{KEM.Enc}]$ and all $(k, c) \in [\mathsf{KEM.Enc}_{pk}]$, key k is uniform in $\mathcal{K}$ in this section.

As public-key encryption tends to be computationally expensive, in practice, KEMs are usually employed to merely transport a rather short (symmetric) key. They are then combined with highly efficient (symmetric) data encapsulation mechanisms (DEMs) to obtain efficient hybrid public-key encryption [CS03]. The selective opening security of hybrid encryption is investigated in Chapter 3. We define *one-way* security in the presence of a *plaintext-checking oracle* (OW-PCA).

Definition 2.2.2 (OW-PCA secure KEM [OP01]). We say a KEM is (τ, q_c, ε)-*OW-PCA secure* if for all τ-time adversaries $\mathcal{A}$ that interact in the $\mathsf{OW\text{-}PCA}$ experiment as given in Figure 2.2 and query the KEM.CHECK oracle at most q_c times we have

$$\Pr\left[\mathsf{OW\text{-}PCA}^{\mathcal{A}} \Rightarrow 1\right] \leq \varepsilon \ .$$

Exp $\mathsf{OW\text{-}PCA}^{\mathcal{A}}$	**Oracle** KEM.CHECK(k, c)
$(pk, sk) \leftarrow_\$ \mathsf{KEM.Gen}$	Return $(\mathsf{KEM.Dec}_{sk}(c) =_? k)$
$(k^*, c^*) \leftarrow_\$ \mathsf{KEM.Enc}_{pk}$	
$k \leftarrow \mathcal{A}^{\text{KEM.CHECK}}(pk, c^*)$	
Stop with $(k =_? k^*)$	

Figure 2.2: $\mathsf{OW\text{-}PCA}$ experiment as used in Definition 2.2.2.

Informally, a KEM is *OW-PCA secure* if for all efficient adversaries ε is small.

We continue by defining message authentication codes.

[1] Observe that this property already follows if the KEM is one-way secure.

Exp $\mathsf{sUF\text{-}OT\text{-}CMA}^{\mathcal{A}}$
$k \leftarrow_\$ \mathsf{MAC.Gen}$
$(m^*, st) \leftarrow_\$ \mathcal{A}_1^{\mathrm{MAC.V{\small RFY}}}$
$t^* \leftarrow_\$ \mathsf{MAC.Tag}_k(m^*)$
$(\tilde{m}, \tilde{t}) \leftarrow_\$ \mathcal{A}_2^{\mathrm{MAC.V{\small RFY}}}(st, t^*)$
Stop with
$(\mathsf{MAC.Vrfy}_k(\tilde{m}, \tilde{t}) \wedge (\tilde{m}, \tilde{t}) \neq (m, t))$

Oracle $\mathrm{MAC.V{\small RFY}}(m, t)$
Return $\mathsf{MAC.Vrfy}_k(m, t)$

Figure 2.3: sUF-OT-CMA experiment used in Definition 2.2.4.

Definition 2.2.3 (message authentication code). A *message authentication code* (MAC) for a message space $\mathcal{M}$ consists of a key space $\mathcal{K}$, a tag space $\mathcal{T}$ and a triple of efficient algorithms $\mathsf{MAC} = (\mathsf{MAC.Gen}, \mathsf{MAC.Tag}, \mathsf{MAC.Vrfy})$ of the form

$$\mathsf{MAC.Gen}\colon \rightarrow_\$ \mathcal{K} \qquad \mathsf{MAC.Tag}\colon \mathcal{K} \times \mathcal{M} \rightarrow_\$ \mathcal{T} \qquad \mathsf{MAC.Vrfy}\colon \mathcal{K} \times \mathcal{M} \times \mathcal{T} \rightarrow \{0,1\} \ .$$

For the MAC to be correct, we require that for all $k \in [\mathsf{MAC.Gen}]$ and for all $m \in \mathcal{M}$ if $t \in [\mathsf{MAC.Tag}_k(m)]$ then $\mathsf{MAC.Vrfy}_k(m, t) = 1$. We say that for $k \in \mathcal{K}$ and $m \in \mathcal{M}$ a tag t (resp. a tuple (m, t)) is *valid*, if $\mathsf{MAC.Vrfy}_k(m, t) = 1$.

In the following MAC will always denote a message authentication code. We define strong unforgeability under one-time chosen message attacks (sUF-OT-CMA) next:

Definition 2.2.4 (sUF-OT-CMA secure MAC). We say a MAC is (τ, q_v, ε)-*sUF-OT-CMA secure* if for all τ-time adversaries $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ that interact in the sUF-OT-CMA experiment as given in Figure 2.3 and query the MAC.VRFY oracle at most q_v times, we have

$$\Pr\left[\mathsf{sUF\text{-}OT\text{-}CMA}^{\mathcal{A}} \Rightarrow 1\right] \leq \varepsilon \ .$$

2.2.2 A Transformation from any OW-PCA KEM

We recall a well known transformation [SBZ02] to turn a KEM into a PKE scheme. Observe that we do not give the transformation in its full generality as we have instantiated the symmetric encryption with a one-time-pad.

Notation Whenever an encryption procedure of a PKE scheme outputs ciphertexts c_i composed of multiple components $c_i^{(1)}, c_i^{(2)}, \ldots$ we write $\langle c_i^{(1)}, c_i^{(2)}, \ldots \rangle$ for the encoding of the components into single one.

Construction 2.2.5 *For $\ell \in \mathbb{N}$ let $\mathcal{M} = \{0,1\}^\ell$ be a plaintext space,* KEM *be a KEM for key space $\mathcal{K}$, and* MAC *be a MAC for messages in $\mathcal{M}$ with key space $\mathcal{K}_{\mathsf{MAC}}$. Let $H \colon \mathcal{K} \to \mathcal{M} \times \mathcal{K}_{\mathsf{MAC}}$ be a (hash) function. Then the procedures given in Figure 2.4 form a PKE scheme.*

For clarity, the encryption procedure is illustrated in Figure 2.5.

Proc $\mathsf{PKE.Gen}$
$(pk, sk) \leftarrow_\$ \mathsf{KEM.Gen}$
Return (pk, sk)

Proc $\mathsf{PKE.Enc}_{pk}(m; r)$
$(k, c^{(1)}) \leftarrow \mathsf{KEM.Enc}_{pk}(r)$
$(k^{sym}, k^{mac}) \leftarrow H(k)$
$c^{(2)} \leftarrow k^{sym} \oplus m$
$c^{(3)} \leftarrow_\$ \mathsf{MAC.Tag}_{k^{mac}}(c^{(2)})$
Return $\langle c^{(1)}, c^{(2)}, c^{(3)} \rangle$

Proc $\mathsf{PKE.Dec}_{sk}(\langle c^{(1)}, c^{(2)}, c^{(3)} \rangle)$
$k \leftarrow \mathsf{KEM.Dec}_{sk}(c^{(1)})$
$(k^{sym}, k^{mac}) \leftarrow H(k)$
If $\mathsf{MAC.Vrfy}_{k^{mac}}(c^{(2)}, c^{(3)}) = 0$:
    Return $\perp$
Else:
    $m \leftarrow c^{(2)} \oplus k^{sym}$
    Return m

Figure 2.4: Construction of a PKE from a KEM, MAC and a hash function H.

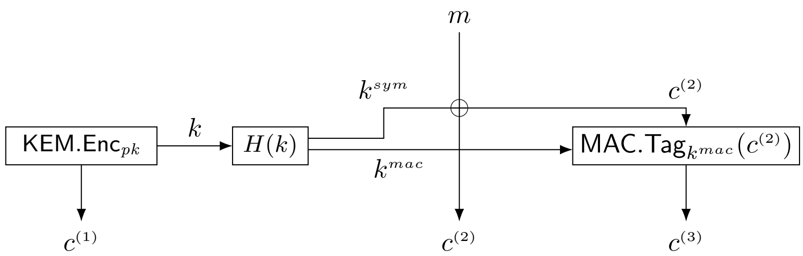

Figure 2.5: Encryption process of Construction 2.2.5. $\langle c^{(1)}, c^{(2)}, c^{(3)} \rangle \leftarrow \mathsf{PKE.Enc}_{pk}(m; r)$.

Observation 2.2.6 Construction 2.2.5 implicitly strengthens a OW-PCA KEM to an IND-CCA secure KEM by letting $(H(k), c)$ be the output of the encapsulation, rather than (k, c), for a random oracle H.[2] Further, the one-time pad constitutes a OT-CPA secure data encapsulation mechanism (DEM), i.e., a symmetric encryption scheme. Combined with a sUF-OT-CMA secure MAC, we obtain a OT-IND-CCA secure DEM. It is well known that a IND-CCA KEM in combination with a OT-IND-CCA DEM results in a IND-CCA secure PKE [HHK10].

Our next theorem strengthens this result by showing that the obtained PKE is even SIM-SO-CCA secure.

[2]We implicitly prove the IND-CCA security of the modified KEM in the proof of Theorem 2.2.7. See the proof of Claim 2.2.12 employing the *oracle patching technique*.

Theorem 2.2.7 *Let* KEM, MAC *and* H *be as required in Construction 2.2.5.*

If KEM *is* $(\tau_{pca}, q_c, \varepsilon_{pca})$*-OW-PCA secure and has unique encapsulations,* MAC *is* $(\tau_{cma}, q_v, \varepsilon_{cma})$*-sUF-OT-CMA secure, then the PKE scheme as given in Figure 2.4 is* $(\tau_{so\text{-}cca}, q_d, q_h, \varepsilon_{so\text{-}cca})$*-SIM-SO-CCA secure where* $\tau_{so\text{-}cca} \leq \min\{\tau_{cma}, \tau_{pca} - \mathcal{O}(q_h \cdot q_d)\}$ *and*

$$\varepsilon_{so\text{-}cca}(n) \leq n \cdot \left(\varepsilon_{cma} + \varepsilon_{pca} + \frac{q_h}{|\mathcal{K}| - q_h} + \frac{q_d}{|\mathcal{C}| - q_d} \right) ,$$

and $q_d \leq \min\{q_v, \frac{q_c - q_h}{2 \cdot q_h}\}$*. Further* H *is modeled as a random oracle that may be queried at most* q_h *times.*

Proof Sketch Eventually, we construct a simulator $\mathcal{S}$. When $\mathcal{S}$ is run in the (ideal) i-SO-CCA experiment it is capable to simulate the (real) r-SO-CCA experiment for a SIM-SO-CCA adversary $\mathcal{A}$. Essentially, $\mathcal{S}$ has to enrich its interaction with $\mathcal{A}$ with everything that was removed from the real SIM-SO-CCA experiment to obtain the ideal i-SO-CCA experiment. That is, $\mathcal{S}$ has to compute pk, encrypt plaintexts coming from a distribution specified by $\mathcal{A}$ and provide oracles Open and PKE.Dec (see Figure 2.1).

The simulator $\mathcal{S}$ will provide $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ with 'non-committing' encryptions that can be opened to any plaintext. The simulator will exploit the programmability of random oracle H in order to open some $c^{(2)} = m \oplus k^{sym}$ where $(k^{sym}, \cdot) \leftarrow H(k)$ to any plaintext. Hence, $\mathcal{S}$ has to ensure that $H(k)$ remains unevaluated until $\mathcal{A}$ asks to open ciphertext $\langle c^{(1)}, c^{(2)}, c^{(3)} \rangle$.

Assume that $\mathcal{A}_2$ is invoked on ciphertexts $\mathbf{c}$. Now, adversary $\mathcal{A}_2$ may arbitrarily query oracles H, Open and PKE.Dec (with the usual restriction for PKE.Dec). Say, for some $i \in [n]$, $\mathcal{A}_2$ queries $\mathrm{H}(k_i)$ or submits a valid ciphertext $\langle c_i^{(1)}, \cdot, \cdot \rangle$ for decryption but did not query Open(i). Answering such queries would make the simulator commit to $H(k_i) = (k_i^{sym}, k_i^{mac})$ and hence to $m = \mathsf{PKE.Dec}_{sk}(c_i)$, rendering it impossible to open c_i to any plaintext.[3]

Our proposed simulator will abort when faced with such a query from $\mathcal{A}$. However, assuming the OW-PCA security of the KEM, and the sUF-OT-CMA security of the MAC we will argue that it is unlikely for the simulator to abort due to queries as described above.

Consider $\mathcal{A}_2$ submitting a valid decryption query of the form $\langle c_i^{(1)}, \cdot, \cdot \rangle$ while it did not query Open(i). Then two cases can occur:

[3]Clearly, $\mathcal{S}$ must not query Open(i) for all $i \in [n]$ to learn all m_i as the set $\mathcal{I}$ tracking the issued opening queries is an input to Pred.

1. $H(k_i)$ is still unevaluated.
   Thus, k_i^{mac} is still uniform and we can employ $\mathcal{A}$ to break the sUF-OT-CMA security of the MAC.

2. $H(k_i)$ has already been evaluated.
   Then $\mathcal{A}$ queried H on k_i. Hence, we can use $\mathcal{A}$ to break the OW-PCA security of the KEM.

We proceed with the detailed proof.

Proof of Theorem 2.2.7. Let $\mathcal{A}_{so} = (\mathcal{A}_{so,1}, \mathcal{A}_{so,2})$ be an adversary against the $(\tau_{so\text{-}cca}, q_d, q_h, \varepsilon_{so\text{-}cca})$-SIM-SO-CCA security of PKE.

We gradually modify the real r-SO-CCA experiments in a sequence of experiments up to a point where we can give a simulator that, when run in the i-SO-CCA experiment, simulates the r-SO-CCA experiment for an r-SO-CCA attacker. The sequence of experiments is given in Figures 2.6 and 2.7.

Exp $\mathsf{Exp}_0^{\mathcal{A}_{so}}(n)$ – $\mathsf{Exp}_4^{\mathcal{A}_{so}}(n)$

$\mathcal{I} \leftarrow \emptyset$; $\mathbf{c} \leftarrow \emptyset$; $L_H \leftarrow \emptyset$
$(pk, sk) \leftarrow_\$ \mathsf{KEM.Gen}$
For $i \leftarrow 1$ to n:
  $r_i \leftarrow_\$ \mathcal{R}$
  $(k_i, c_i^{(1)}) \leftarrow_\$ \mathsf{KEM.Enc}_{pk}(r_i)$
  $(k_i^{sym}, k_i^{mac}) \leftarrow_\$ \{0,1\}^\ell \times \mathcal{K}_{\mathsf{MAC}}$ // Exp_0 – Exp_1
  $H(k_i) \leftarrow (k_i^{sym}, k_i^{mac})$ // Exp_0 – Exp_1
$(\mathfrak{D}, st) \leftarrow_\$ \mathcal{A}_1^{\mathrm{H},\mathrm{PKE.Dec}}(pk, n)$
$\mathbf{m} \leftarrow_\$ \mathfrak{D}$
For $i \leftarrow 1$ to n:
  $c_i \leftarrow \langle c_i^{(1)}, m_i \oplus k_i^{sym}, \mathsf{MAC.Tag}_{k_i^{mac}}(m_i \oplus k_i^{sym})\rangle$ // Exp_0 – Exp_1
  $(\sigma_i^{sym}, \sigma_i^{mac}) \leftarrow_\$ \{0,1\}^\ell \times \mathcal{K}_{\mathsf{MAC}}$ // Exp_2 – Exp_4
  $c_i \leftarrow \langle c_i^{(1)}, \sigma_i^{mac}, \mathsf{MAC.Tag}_{\sigma_i^{mac}}(\sigma_i^{sym})\rangle$ // Exp_2 – Exp_4
$\mathbf{c} \leftarrow (c_1, \ldots, c_n)$
$out \leftarrow_\$ \mathcal{A}_2^{\mathrm{H},\mathrm{OPEN},\mathrm{PKE.Dec}}(st, \mathbf{c})$
Stop with $\mathsf{Pred}(\mathfrak{D}, \mathbf{m}, \mathcal{I}, out)$

Figure 2.6: Sequence of experiments $\mathsf{Exp}_0(n)$ – $\mathsf{Exp}_4(n)$ as used in the proof of Theorem 2.2.7. Provided oracles H, OPEN and PKE.DEC are specified in Figure 2.7 on page 76.

Experiment Exp_0. Experiment Exp_0 implements random oracle H by lazy sampling. To this end, it keeps track of queries $\mathrm{H}(s)$ (either by internal procedures or $\mathcal{A}_{so}$) via maintaining a list L_H. For a query s, $\mathrm{H}(s)$ returns h_s if there is an entry $(s, h_s) \in L_H$

Oracle $\mathrm{H}(s)$

If $(s, \cdot) \notin L_H$:
  If $s \in (k_1, \ldots, k_n)$:
    Let i s.t. $s = k_i$
      Abort // $\mathcal{A}_{so,1}$: $\mathsf{Exp}_1 - \mathsf{Exp}_4$
      Abort // $\mathcal{A}_{so,2}$: Exp_4
      $H(k_i) \leftarrow (\sigma_i^{sym} \oplus m_i, \sigma_i^{mac})$ // $\mathsf{Exp}_2 - \mathsf{Exp}_4$
  Else:
    $h_s \leftarrow_\$ \{0,1\}^\ell \times \mathcal{K}_{\mathsf{MAC}}$
    $H(s) \leftarrow h_s$
Return h_s

Oracle $\textsc{Open}(i)$

$\mathcal{I} \leftarrow \mathcal{I} \cup \{i\}$
$H(k_i) \leftarrow (\sigma_i^{sym} \oplus m_i, \sigma_i^{mac})$ // $\mathsf{Exp}_2 - \mathsf{Exp}_4$
Return (m_i, r_i)

Oracle $\mathrm{PKE.}\textsc{Dec}(\langle c^{(1)}, c^{(2)}, c^{(3)} \rangle)$

If $\langle c^{(1)}, c^{(2)}, c^{(3)} \rangle \in \mathbf{c}$: Abort
If $c^{(1)} \in \mathbf{c}^{(1)}$: Abort // $\mathcal{A}_{so,1}$: $\mathsf{Exp}_1 - \mathsf{Exp}_4$
$k \leftarrow \mathsf{KEM.Dec}_{sk}(c^{(1)})$
If $\begin{pmatrix} c^{(1)} \in \mathbf{c}^{(1)} \wedge (k, \cdot) \notin L_H \wedge \\ \mathsf{MAC.Vrfy}_{\sigma_i^{mac}}(c^{(2)}, c^{(3)}) = 1 \end{pmatrix}$: Abort // $\mathsf{Exp}_3 - \mathsf{Exp}_4$
$(k^{sym}, k^{mac}) \leftarrow H(k)$
If $\mathsf{MAC.Vrfy}_{k^{mac}}(c^{(2)}, c^{(3)}) = 0$:
  Return $\perp$
Else:
  Return $c^{(2)} \oplus k^{sym}$

Figure 2.7: Pseudocode for oracles provided to $\mathcal{A}_{so}$ run in the sequence of experiments as given in Figure 2.6.

(line 17), otherwise H samples h_s at random, and returns h_s (lines 24, 25). We implicitly assume an update operation $L_H \leftarrow L_H \cup \{(s, h_s)\}$ to happen in the background.

We introduce some syntactical changes: Experiment Exp_0 runs $\mathsf{KEM.Enc}_{pk}$ to generate $(k_i, c_i^{(1)})$ for $i \in [n]$ before $\mathcal{A}_{so,1}$ is started (lines 04, 05). Further, Exp_0 for all $i \in [n]$ samples (k_i^{sym}, k_i^{mac}) uniformly at random and sets $H(k_i) \leftarrow (k_i^{sym}, k_i^{mac})$ before $\mathcal{A}_{so,1}$ is invoked (lines 06, 07).

Claim 2.2.8 It holds $\Pr\left[\mathsf{r\text{-}SO\text{-}CCA}^{\mathcal{A}_{so}}(n) \Rightarrow 1\right] = \Pr\left[\mathsf{Exp}_0^{\mathcal{A}_{so}}(n) \Rightarrow 1\right]$.

Proof of Claim 2.2.8. Clearly, it makes no difference if the experiment for all $i \in [n]$ samples r_i and runs $\mathsf{KEM.Enc}(r_i)$ on demand or in advance before $\mathcal{A}_{so,1}$ returns $\mathfrak{D}$.

Since H is considered a random oracle, $H(s)$ is sampled uniformly random for every fresh query H(s). Hence, it does not change the distribution to sample (k_i^{sym}, k_i^{mac}) uniformly in the first place and setting $H(k_i) \leftarrow (k_i^{sym}, k_i^{mac})$ afterwards. ■

Experiment Exp_1. We add two abort instructions. Experiment Exp_1 aborts if for any $i \in [n]$ $\mathcal{A}_{so,1}$ queries H(k_i) (see line 20) or PKE.Dec$(\langle c_i^{(1)}, \cdot, \cdot \rangle)$ (line 31), even if the latter might be invalid.

Claim 2.2.9 It holds

$$\left|\Pr\left[\mathsf{Exp}_0^{\mathcal{A}_{so}}(n) \Rightarrow 1\right] - \Pr\left[\mathsf{Exp}_1^{\mathcal{A}_{so}}(n) \Rightarrow 1\right]\right| \leq n \cdot \left(\frac{q_h}{|\mathcal{K}| - q_h} + \frac{q_d}{|\mathcal{C}| - q_d}\right) \quad .$$

Proof of Claim 2.2.9. Let ABORT denote the event that experiment Exp_1 aborts due to lines 20 or 31. As experiments Exp_0 and Exp_1 are identical until ABORT happens, it follows that $|\Pr[\mathsf{Exp}_0^{\mathcal{A}_{so}}(n) \Rightarrow 1] - \Pr[\mathsf{Exp}_1^{\mathcal{A}_{so}}(n) \Rightarrow 1]| \leq \Pr[\text{ABORT}]$ holds.[4]

Let viaHash (resp. viaDec) denote the event that ABORT was caused by a hash (resp. a decryption) query of $\mathcal{A}_{so}$. Let s_i denote the i^{th} hash and $d_i = \langle d_i^{(1)}, d_i^{(2)}, d_i^{(3)} \rangle$ the i^{th} decryption query of $\mathcal{A}_{so}$. Then

[4] 'Fundamental Lemma of game playing' [Sho04c].

$$
\begin{aligned}
&\Pr[\textsc{Abort}] = \Pr[\textsc{viaHash}] + \Pr[\textsc{viaDec}] \\
&\le \Pr[s_1 \in \{k_i\}_{i=1}^n] + \sum_{i=2}^{q_h} \Pr\left[s_i \in \{k_i\}_{i=1}^n \,\middle|\, \bigwedge_{j=1}^{i-1} s_j \notin \{k_i\}_{i=1}^n \right] \\
&+ \Pr[d_i^{(1)} \in \{c_i^{(1)}\}_{i=1}^n] + \sum_{i=2}^{q_d} \Pr\left[d_i^{(1)} \in \{c_i^{(1)}\}_{i=1}^n \,\middle|\, \bigwedge_{j=1}^{i-1} d_j^{(1)} \notin \{c_i^{(1)}\}_{i=1}^n \right] \\
&= \sum_{i=1}^{q_h} \frac{n}{|\mathcal{K}| - (i-1)} + \sum_{i=1}^{q_d} \frac{n}{|\mathcal{C}| - (i-1)} \\
&\le \sum_{i=1}^{q_h} \frac{n}{|\mathcal{K}| - q_h} + \sum_{i=1}^{q_d} \frac{n}{|\mathcal{C}| - q_d} = n \cdot \left(\frac{q_h}{|\mathcal{K}| - q_h} + \frac{q_d}{|\mathcal{C}| - q_d} \right) \quad ,
\end{aligned}
$$

which holds as $\mathsf{KEM.Enc}$ samples $k \leftarrow_{\$} \mathcal{K}$ and KEM has unique encapsulations. ■

Experiment Exp_2. In experiment Exp_2 we change the way plaintexts are encrypted and answer hash queries in a different way. The experiment does not sample keys (k_i^{sym}, k_i^{mac}) before $\mathcal{A}_{so,1}$ is run (see line 06). Neither is $H(k_i)$ programmed accordingly (line 07).

For all $i \in [n]$ ciphertext $c_i^{(2)}$ is replaced by a uniform element σ_i^{sym} (lines 12, 13). Further, a uniform MAC key σ_i^{mac} is sampled and $c_i^{(3)}$ is computed as $c_i^{(3)} \leftarrow_{\$} \mathsf{MAC.Tag}_{\sigma_i^{mac}}(\sigma_i^{sym})$ (lines 12, 13).

If for any $i \in [n]$ adversary $\mathcal{A}_{so}$ queries $\mathrm{H}(k_i)$ or $\textsc{Open}(i)$, experiment Exp_2 programs $H(k_i) \leftarrow (\sigma_i^{sym} \oplus m_i, \sigma_i^{mac})$ in line 22, resp. in line 28.

Bear in mind that as from now $(k_i, \cdot) \notin L_H$ implies that $\textsc{Open}(i)$ was not called.

Claim 2.2.10 It holds $\Pr\left[\mathsf{Exp}_1^{\mathcal{A}_{so}}(n) \Rightarrow 1\right] = \Pr\left[\mathsf{Exp}_2^{\mathcal{A}_{so}}(n) \Rightarrow 1\right]$.

Proof of Claim 2.2.10. Assume that experiment Exp_2 does not abort. Then for all $i \in [n]$ keys k_i^{sym} and k_i^{mac} are uniformly random when $\mathcal{A}_{so,1}$ halts. Therefore for all $i \in [n]$ ciphertext $c_i^{(2)} = m_i \oplus k_i^{sym}$ is uniform and $c_i^{(3)}$ is a valid tag of a uniformly random message under a uniform key. Consequently, in experiment Exp_2 the ciphertexts $c_i^{(2)} \leftarrow \sigma_i^{sym}$ can be sampled uniformly and tags can be computed using a uniform MAC key σ_i^{mac} without changing the distribution of the encryptions c_i.

Although, $H(k_i)$ is not programmed immediately, H has to be kept consistently. This is done by letting $H(k_i) \leftarrow (\sigma_i^{sym} \oplus m_i, \sigma_i^{mac})$ once $\mathrm{H}(k_i)$ or $\textsc{Open}(i)$ is called. ■

It remains to treat cases 1 and 2 mentioned in the proof sketch of Theorem 2.2.7.

Experiment Exp_3. We introduce another abort condition: If $\mathcal{A}_{so,2}$ issues a decryption query $\langle c_i^{(1)}, c^{(2)}, c^{(3)} \rangle$ for which $H(k_i)$ has not been evaluated and it holds that $\mathsf{MAC.Vrfy}_{\sigma_i^{mac}}(c^{(2)}, c^{(3)}) = 1$, experiment Exp_3 aborts (see line 33).

Claim 2.2.11 There exists an adversary $\mathcal{A}_{cma}$ that breaks the $(\tau_{cma}, q_v, \varepsilon_{cma})$-sUF-OT-CMA security of MAC where

$$\tau_{cma} \approx \tau_{so\text{-}cca} \quad , \quad \varepsilon_{cma} \geq \frac{1}{n} \cdot \left| \Pr\left[\mathsf{Exp}_2^{\mathcal{A}_{so}}(n) \Rightarrow 1\right] - \Pr\left[\mathsf{Exp}_3^{\mathcal{A}_{so}}(n) \Rightarrow 1\right] \right| \quad , \quad q_v \geq q_d \ .$$

Proof of Claim 2.2.11. Experiments Exp_2 and Exp_3 are identical until the newly introduced Abort is executed in the latter experiment. Let ABORT denote the event that experiment Exp_3 aborts in line 33. It suffices to bound Pr[ABORT].

We construct adversary $\mathcal{A}_{cma} = (\mathcal{A}_{cma,1}, \mathcal{A}_{cma,2})$. Adversary $\mathcal{A}_{cma,1}$ samples $i^* \leftarrow_\$ [n]$ and runs adversary $\mathcal{A}_{so,1}$ as in experiment Exp_3. All hash queries by $\mathcal{A}_{so}$ are answered honestly. Receiving $\mathfrak{D}$, $\mathcal{A}_{cma,1}$ samples plaintexts and processes them as in Exp_3 except for the i^{*th} ciphertext. Here it outputs $\sigma_{i^*}^{sym}$ to its sUF-OT-CMA experiment and terminates.

When $\mathcal{A}_{cma,2}(t^*)$ is started it assembles $c_{i^*} \leftarrow \langle c_i^{(1)}, \sigma_i^{mac}, t^* \rangle$ and invokes $\mathcal{A}_{so,2}(\mathbf{c})$.

If $\mathcal{A}_{so,2}$ should call OPEN(i^*), $\mathcal{A}_{cma,2}$ aborts. For each decryption query of the form $\langle c_i^{(1)}, c_i^{(2)}, c_i^{(3)} \rangle)\rangle$, $\mathcal{A}_{cma,2}$ invokes its MAC.VRFY oracle to detect when $\mathcal{A}_{so,2}$ submits a valid ciphertext $\langle c_i^{(1)}, c^{(2)}, c^{(3)} \rangle$. Once it occurs, $\mathcal{A}_{cma,2}$ outputs $(c^{(2)}, c^{(3)})$ and halts.

ANALYSIS Assume that ABORT happens. Further, say $\mathcal{A}_{cma}$ guessed correctly where to embed its challenge. That is, ABORT happens as $\mathcal{A}_{so,2}$ submits $\langle c_{i^*}^{(1)}, \tilde{m}, \tilde{t} \rangle$ to decryption.

Then $(\tilde{m}, \tilde{t}) \neq (\sigma_i^{mac}, t^*)$ as otherwise $(c_{i^*}^{(1)}, \tilde{m}, \tilde{t}) \in \mathbf{c}$. Hence, $\tilde{t}$ is either a new valid tag $\tilde{t} \neq t^*$ for message σ_i^{mac} or $\tilde{t}$ is a valid tag for a new message $\tilde{m} \neq \sigma_i^{mac}$. Thus, $\mathcal{A}_{cma}$ wins the sUF-OT-CMA experiment by returning $(\tilde{m}, \tilde{t})$. As $i^* \leftarrow_\$ [n]$ we have

$$\varepsilon_{cma} \geq \frac{1}{n} \cdot \left| \Pr\left[\mathsf{Exp}_2^{\mathcal{A}_{so}}(n) \Rightarrow 1\right] - \Pr\left[\mathsf{Exp}_3^{\mathcal{A}_{so}}(n) \Rightarrow 1\right] \right| \ .$$

One easily verifies that the running time of $\mathcal{A}_{cma}$ is roughly the running time of $\mathcal{A}_{so}$. Further, $\mathcal{A}_{cma}$ will issue a query to MAC.VRFY per decryption query posed by $\mathcal{A}_{so}$. ■

Experiment Exp_4. Line 21 is added. That is, experiment Exp_4 aborts if for some $i \in [n]$ adversary $\mathcal{A}_{so,2}$ queries $\mathrm{H}(k_i)$ and did not call OPEN(i) before.

Claim 2.2.12 There exists an adversary $\mathcal{A}_{pca}$ that breaks the $(\tau_{pca}, q_c, \varepsilon_{pca})$-OW-PCA security of KEM where

$$\tau_{pca} \geq \tau_{so\text{-}cca} + \mathcal{O}(q_h \cdot q_d) \ , \quad \varepsilon_{pca} \geq \frac{1}{n} \cdot \left| \Pr\left[\mathsf{Exp}_3^{\mathcal{A}_{so}}(n) \Rightarrow 1\right] - \Pr\left[\mathsf{Exp}_4^{\mathcal{A}_{so}}(n) \Rightarrow 1\right]\right|$$

and $q_c \geq q_h \cdot (1 + 2q_d)$.

Interlude: Oracle Patching [CS03] Observe that adversary $\mathcal{A}_{pca}$ against the OW-PCA security of the KEM does not hold the secret key but has to answer decryption queries posed by $\mathcal{A}_{so}$. To solve the issue we employ the nifty *oracle patching technique* that we describe next.

Say, $\mathcal{A}_{so}$ submits a decryption query of the form $\langle c^{(1)}, c^{(2)}, c^{(3)} \rangle$. Now $\mathcal{A}_{pca}$ checks list L_H whether there is an entry $(s, \cdot)$ such that KEM.Check$(s, c^{(1)}) = 1$. If so, $\mathcal{A}_{pca}$ uses $(k^{sym}, k^{mac}) \leftarrow H(s)$ for decryption. If not, $\mathcal{A}_{pca}$ samples uniform random keys (k^{sym}, k^{mac}) for decryption. Note that the uniqueness of encapsulations permits sampling of fresh keys here. However, by processing the decryption query $\mathcal{A}_{pca}$ committed to hash value $H(k)$ for $k \leftarrow \mathsf{KEM.Dec}_{sk}(c^{(1)})$. In order to keep the simulation consistent, $\mathcal{A}_{pca}$ adds $(c^{(1)}, (k^{sym}, k^{mac}))$ to some dedicated list L_{patch}.

It remains to explain how hash queries are answered. For each fresh hash query H(s), $\mathcal{A}_{pca}$ has to check whether there is an entry $(c^{(1)}, k^{sym}, k^{mac})$ in L_{patch} such that KEM.Check$(s, c^{(1)}) = 1$. If there is such an entry, $\mathcal{A}_{pca}$ replies to H(s) by returning (k^{sym}, k^{mac}). If there is no such entry, $\mathcal{A}_{pca}$ replies with a uniformly random key pair from $\{0,1\}^\ell \times \mathcal{K}_{\mathsf{MAC}}$.

We proceed with the proof of Claim 2.2.12.

Proof of Claim 2.2.12. Let Abort denote the event that experiment Exp_4 aborts in line 21. As experiments Exp_3 and Exp_4 are identical until Abort happens, it suffices to bound Pr[Abort].

We describe how to construct adversary $\mathcal{A}_{pca}$. Attacker $\mathcal{A}_{pca}$ is started on input (pk, c^*). Let $k^* \leftarrow \mathsf{KEM.Dec}_{sk}(c^*)$. $\mathcal{A}_{pca}$ picks $i \leftarrow_\$ [n]$ and calls $\mathcal{A}_{so,1}(pk, n)$. Receiving $\mathfrak{D}$ from $\mathcal{A}_{so,1}$, $\mathcal{A}_{pca}$ samples plaintexts and encrypts them as in Exp_4 but lets $c_i^{(1)} \leftarrow c_{i^*}$.

If $\mathcal{A}_{so,2}$ submits a hash query H(s), $\mathcal{A}_{pca}$ checks whether $s \in [n] \setminus \{k_{i^*}\}$ in which case it aborts. If not, $\mathcal{A}_{pca}$ checks whether KEM.Check$(s, c_{i^*}^{(1)}) = 1$ holds. If so, it returns s to its $\mathsf{OW\text{-}PCA}$ experiment and halts, otherwise it processes the hash query by means of the oracle patching technique.

Let us consider the simulation of the decryption oracle. Importantly, $\mathcal{A}_{pca}$ still has to check whether it has to abort due to line 33 introduced in experiment Exp_3. To this end, for any query PKE.Dec$(\langle c^{(1)}, c^{(2)}, c^{(3)} \rangle)$ it checks if $c^{(1)} \in \{c_i^{(1)}\}_{i=1}^n$ and

$\mathsf{MAC.Vrfy}_{\sigma_i^{mac}}(c^{(2)}, c^{(3)}) = 1$ hold and aborts if so. Note that $\mathcal{A}_{pca}$ can use σ_i^{mac} for verification because $H(k_i)$ for $k_i \leftarrow \mathsf{KEM.Dec}_{sk}(c_i^{(1)})$ has not been evaluated; otherwise $\mathcal{A}_{pca}$ would have aborted earlier. Otherwise, decryption queries are answered via the oracle patching technique.

Opening queries are answered honestly unless $\mathcal{A}_{so,2}$ queries Open(i^*) where $\mathcal{A}_{pca}$ aborts.

Analysis Assume that Abort happens. Adversary $\mathcal{A}_{pca}$ can detect when Abort happens as it knows all k_i except for i^*. However, using its KEM.Check oracle it can detect if H(k^*) is queried. Otherwise, up to aborts happening, $\mathcal{A}_{so}$'s view in its interaction with $\mathcal{A}_{pca}$ are identical to its interaction in the Exp_4 experiment.

$\mathcal{A}_{pca}$ wins its OW-PCA experiment if Abort happens (the first time) due to $\mathcal{A}_{so,2}$ submitting $k^* = k_{i^*}$. As $i^* \leftarrow_\$ [n]$ we have

$$\varepsilon_{pca} \geq \frac{1}{n} \cdot \left| \Pr\left[\mathsf{Exp}_3^{\mathcal{A}_{so}}(n) \Rightarrow 1\right] - \Pr\left[\mathsf{Exp}_4^{\mathcal{A}_{so}}(n) \Rightarrow 1\right] \right| \ .$$

Further, for each hash query $\mathcal{A}_{pca}$ iterates over L_{patch} and issues at most $|L_{patch}| \leq q_d$ (oracle patching) calls to KEM.Check plus one more call to check whether it holds that KEM.Check$(s, c_{i^*}^{(1)}) = 1$.

For each decryption query, $\mathcal{A}_{pca}$ queries the KEM.Check oracle at most $|L_H| \leq q_h$ times while checking all entries in L_H due to the oracle patching technique.

Hence,

$$\tau_{pca} = \tau_{so\text{-}cca} + \mathcal{O}(q_h \cdot q_d) \ , \qquad q_c = q_h \cdot (1 + 2q_d) \ .$$

■

We construct a simulator to conclude the proof of Theorem 2.2.7.

Claim 2.2.13 There exists a simulator $\mathcal{S}$ with roughly the same running time as $\mathcal{A}_{so}$ such that

$$\left| \Pr\left[\mathsf{Exp}_4^{\mathcal{A}_{so}}(n) \Rightarrow 1\right] - \Pr\left[\mathsf{i\text{-}SO\text{-}CCA}^{\mathcal{S}}(n) \Rightarrow 1\right] \right| = 0 \ .$$

Proof of Claim 2.2.13. We describe $\mathcal{S} = (\mathcal{S}_1, \mathcal{S}_2)$. Simulator $\mathcal{S}_1(n)$ is run and invokes $(pk, sk) \leftarrow_\$ \mathsf{KEM.Gen}$. Then it runs $\mathcal{A}_{so,1}$ on (pk, n). Hash and decryption queries are answered as in experiment Exp_4. When $\mathcal{A}_{so,1}$ outputs $\mathfrak{D}$, $\mathcal{S}_1$ relays $\mathfrak{D}$ to the i-SO-CCA experiment and halts.

When $\mathcal{S}_2$ is started, it computes ciphertexts as in experiment Exp_4 and runs $\mathcal{A}_{so,2}$ on them. The simulator answers decryption and hash queries as before. If $\mathcal{A}_{so,2}$ queries Open(i), $\mathcal{S}_2$ relays the query to its i-SO-CCA experiment and receives m_i. $\mathcal{S}_2$ programs H accordingly (see line 28 in Figure 2.7) and sends (m_i, r_i) to $\mathcal{A}_{so,2}$. When $\mathcal{A}_{so,2}$ halts with output *out*, $\mathcal{S}_2$ outputs *out* and terminates.

If the simulator does not abort, it clearly perfectly simulates the r-SO-CCA experiment for adversary $\mathcal{A}_{so}$. ∎

Collecting Claims 2.2.8 to 2.2.13 yields the proof of Theorem 2.2.7. ∎

Observe that, here, the simulator $\mathcal{S}_2$'s input $(|m_1|, \ldots, |m_n|)$ is redundant as $\mathcal{M} = \{0,1\}^\ell$ for a-priori fixed ℓ.

2.2.4 Implications for Practical Encryption Schemes

We now give specific instantiations of SIM-SO-CCA secure schemes obtained via Construction 2.2.5. We focus on two well-known KEMs, namely the Diffie-Hellman KEM and the RSA KEM. Further, note that we only require a one-time secure MAC in Theorem 2.2.7. It is well-known that such MACs can be constructed information-theoretically [WC81].

DHIES Let $\mathbb{G}$ be a group of prime-order p, and let g be a generator. The Diffie-Hellman KEM $\mathsf{DH\text{-}KEM} = (\mathsf{DH.Gen}, \mathsf{DH.Enc}, \mathsf{DH.Dec})$ is defined as follows. The key generation algorithm $\mathsf{DH.Gen}$ picks $x \leftarrow_\$ \mathbb{Z}_p$ and defines $pk := X := g^x$ and $sk := x$; the encapsulation algorithm $\mathsf{DH.Enc}_{pk}$ picks $r \leftarrow_\$ \mathbb{Z}_p$ and returns $(c, k) \leftarrow (g^r, X^r)$; the decapsulation algorithm $\mathsf{KEM.Dec}_{sk}(c)$ returns $k \leftarrow c^x$.

OW-PCA security of the $\mathsf{DH\text{-}KEM}$ is equivalent to the *strong Diffie-Hellman (sDH)* assumption [ABR01]. The sDH assumption states that there is no efficient adversary $\mathcal{A}$ that, given two random group elements $U := g^u, V := g^v$ and a *restricted* DDH oracle $\mathcal{O}_v(\cdot,\cdot)$ where $\mathcal{O}_v(a,b) := (a^v =_? b)$, computes g^{uv} with high probability.

We obtain the DHIES scheme (instantiated with a one-time pad) by instantiating *Construction* 2.2.5 with the $\mathsf{DH\text{-}KEM}$; we denote the scheme with $\mathsf{DHIES}_\oplus$. Then we obtain the following informal corollary whose proof is a direct consequence of Theorem 2.2.7.

Corollary 2.2.14 $\mathsf{DHIES}_\oplus$ *is SIM-SO-CCA secure in the random oracle model, if* MAC *is sUF-OT-CMA secure and the sDH assumption holds in* $\mathbb{G}$.

RSA-KEM We obtain another selective-opening secure encryption scheme if we plug the $\mathsf{RSA\text{-}KEM}$ (RFC 5990) into the generic transformation given in Figure 2.4.

The $\mathsf{RSA\text{-}KEM} = (\mathsf{RSA.Gen}, \mathsf{RSA.Enc}, \mathsf{RSA.Dec})$ is defined as follows. $\mathsf{RSA.Gen}$ computes an RSA modulus $N = p \cdot q$ for some primes p, q. Then it lets e, d be such that

$e \cdot d \equiv 1 \mod \phi(N)$. $\mathsf{RSA.Gen}$ outputs $(pk, sk) \leftarrow ((N, e), d)$. The key encapsulation outputs $(r, r^e) \leftarrow_{\$} \mathsf{RSA.Enc}_{pk}$. Decapsulation $\mathsf{RSA.Dec}_{sk}(c)$ computes $k \leftarrow c^e \mod N$.

OW-PCA security of the RSA-KEM holds under the RSA assumption [Sho04a]. Hence, under the RSA assumption, the obtained PKE scheme (as described in ISO18033-2 [Sho04a]) is SIM-SO-CCA secure in the random oracle model.

Both reductions for the OW-PCA security of the DH-KEM and RSA-KEM respectively are tight.

2.2.5 Selective Opening Security of Hybrid PKE and KEMs

We conclude Section 2.2 with a rather short discussion. As mentioned in Observation 2.2.6, Construction 2.2.5 can be seen as a hybrid PKE where we implicitly construct an IND-CCA secure KEM and combine it with a OT-IND-CCA secure DEM. (While the DEM is assembled from the (OT-IND-CPA secure) one-time pad and a one-time secure MAC.)

DEMs Let us take a closer look at the employed DEM. Observe that key k^{sym} for the one-time pad is the output of a random oracle. Thus, allowing a simulator to efficiently open ciphertexts to any plaintext (by programming H accordingly).

However, the efficient openability comes at the price of a restricted plaintext space: The obtained PKE can only encrypt plaintexts at most as long as the output length of H. In Chapter 3 we define *simulatable* DEMs that allow for encapsulation of plaintext of arbitrary length while still ensuring efficient openability.

Hybrid PKE Consider a hybrid PKE that is exposed to an SO attack. Let $\langle c^{(1)}, c^{(2)} \rangle$ denote a hybrid ciphertext of a plaintext m where $c^{(1)}$ is contributed by the KEM encapsulation $(k, c^{(1)}) \leftarrow \mathsf{KEM.Enc}_{pk}(r)$ and $c^{(2)} \leftarrow \mathsf{DEM.Enc}_k(m)$. We observe that an SO attacker on the PKE scheme implicitly performs an SO attack on the KEM as well: Opening a ciphertext reveals (m, r). Thus, the attacker can compute k, 'the plaintext' encapsulated in $c^{(1)}$ by running $\mathsf{KEM.Enc}_{pk}(r)$.

The natural question that arises is as follows: Why do we not require IND-SO-CCA secure KEMs, rather than IND-CCA, to obtain SIM-SO-CCA secure PKE?

The answer is given by Theorem 1.5.8. Technically, we do require IND-SO-CCA security. However, IND-CCA security for KEMs tightly implies its IND-SO-CCA security. To this end, one may view KEM as a PKE scheme:[5] If we let $n \in \mathbb{N}$, then we see that n invocations (on fresh randomness r_i) of $(k_i, c_i) \leftarrow \mathsf{KEM.Enc}_{pk}(r_i)$ output 'plaintexts' $(k_1, \ldots, k_n)$ that come from a product distribution. More precisely, the distribution $\mathfrak{D}_n$ of $(k_1, \ldots, k_n)$ over $\mathcal{K}^n$ (induced by the randomness of KEM.Gen and r_i) can be written as n independent distributions over $\mathcal{K}$. Hence, the sequence of distributions $\{\mathfrak{D}_n\}_{n\in\mathbb{N}}$ is efficiently decomposable and Theorem 1.5.8 from Section 1.5 applies.

Note that in each step of the hybrid argument in the proof of Theorem 1.5.8 the reduction to IND-CCA security is tight. Thus, in all security proofs proving security under SO-CCA attacks, we can implicitly employ IND-SO-CCA security for a KEM while only losing the winning probability of some IND-CCA attacker.

2.3 The OAEP Transformation

The second construction of public-key encryption schemes that we consider is the well-known Optimal Asymmetric Encryption Padding (OAEP) transformation [BR95]. OAEP is a generic transformation for constructing public-key encryption schemes from trapdoor permutations that was proposed by Bellare and Rogaway. Since then, it has become an important ingredient in many security protocols and security standards like TLS [DR08, Res02], SSH [Har06], S/MIME [RT10, Hou03], EAP [CA06] and Kerberos [NSF05, Rae05]. We show that OAEP is SIM-SO-CCA secure when instantiated with a *partial-domain trapdoor permutation* (Section 2.3.1). In fact, our result holds not only for trapdoor permutations, but for *injective* trapdoor functions as well. Since SIM-SO-CCA security implies IND-CCA security, our proof also provides an alternative to the IND-CCA security proof of [FOPS01]. Interestingly, despite that we are analyzing security in a stronger security model, our proof seems to be somewhat simpler than the proof of [FOPS01], giving a more direct insight into which properties of the OAEP construction and the underlying trapdoor permutation make OAEP secure.

Provable Security of OAEP OAEP was initially published [BR95] with a flawed security proof discovered some years later by Shoup [Sho02]. Shoup showed furthermore that it is unlikely that the security of OAEP can be reduced to the security of the underlying trapdoor permutation alone. Moreover, Shoup described a modified construction, termed OAEP+, together with a corresponding security proof.

Boldyreva and Fischlin [BF06] studied the security of OAEP when only one of the two hash functions is modeled as a random oracle. The latter result was strengthened by

[5] For the matter of this discussion, we gloss over the syntactical differences.

Kiltz *et al.* [KOS10], who proved the IND-CPA security of OAEP without random oracles, when the underlying trapdoor permutation is *lossy* [PW08]. Since lossy encryption implies IND-SO-CPA security [BHY09], this immediately shows that OAEP is IND-SO-CPA secure in the standard model. However, we stress that prior to our work it was not clear if OAEP meets the stronger notion of SIM-SO-CCA security, neither in the standard model nor in the random oracle.

There also exist a number of negative results [Bro06, KP09] showing the impossibility of instantiating OAEP without random oracles. The latter showed that OAEP cannot be proven IND-CCA secure in the standard model. Thus, a proof of SIM-SO-CCA security in the random oracle model is the strongest result we can hope for, since SIM-SO-CCA security implies IND-CCA security.

2.3.1 Trapdoor Permutations and Partial-Domain One-wayness

Definition 2.3.1 (trapdoor permutation). A *trapdoor permutation* (TDP) over a finite set of bitstrings $\{0,1\}^k$ consists of an evaluation key space $\mathcal{EK}$, a trapdoor space $\mathcal{TD}$ and a triple of efficient algorithms $\mathcal{T} = (F.\mathsf{Gen}, F, F^{-1})$ where

$$F.\mathsf{Gen}\colon \to_\$ \mathcal{EK} \times \mathcal{TD} \quad F\colon \mathcal{EK} \times \{0,1\}^k \to \{0,1\}^k \quad F^{-1}\colon \mathcal{TD} \times \{0,1\}^k \to \{0,1\}^k$$

where for all $(ek, td) \in [F.\mathsf{Gen}]$ we have that F_{ek} is a permutation on $\{0,1\}^k$ and for all $(ek, td) \in [F.\mathsf{Gen}]$ and all $s \in \{0,1\}^k$ we have $F^{-1}_{td}(F_{ek}(s)) = s$.

For $k = \ell + k_1 + k_0$ we can write F as

$$F_{ek}\colon \{0,1\}^{\ell+k_1} \times \{0,1\}^{k_0} \to \{0,1\}^k \ .$$

Definition 2.3.2 (partial-domain secure trapdoor permutation). Let $\mathcal{T}$ be a TDF as given in Definition 2.3.1. We say $\mathcal{T}$ is (τ, ε)*-partial-domain one-way secure* if there exist $\ell, k_1, k_2 \in \mathbb{N}$ such that for all τ-time adversaries $\mathcal{A}$ that interact with the $\mathsf{PD\text{-}OW}$ experiment as given in Figure 2.8 we have

$$\Pr\left[\mathsf{PD\text{-}OW}^{\mathcal{A}} \Rightarrow 1\right] \leq \varepsilon \ .$$

Informally, a trapdoor permutation is *partial domain one-way secure* if ε is small for all efficient adversaries. For a partial domain secure TDF $\mathcal{T}$ we implicitly assume that values ℓ, k_1, k_2 as required in Definition 2.3.2 are part of $\mathcal{T}$'s description.

Exp $\mathsf{OW\text{-}PCA}^{\mathcal{A}}$

$(ek, td) \leftarrow_{\$} F.\mathsf{Gen}$
$(s, t) \leftarrow_{\$} \{0,1\}^{\ell+k_1} \times \{0,1\}^{k_0}$
$y \leftarrow F_{ek}((s,t))$
$s' \leftarrow_{\$} \mathcal{A}(ek, y)$
Stop with $(s =_? s')$

Figure 2.8: Experiment PD-OW as used in Definition 2.3.2.

2.3.2 The Optimal Asymmetric Encryption Padding (OAEP)

We begin by describing the OAEP transformation.

Construction 2.3.3 (OAEP transformation). *Let $\mathcal{T} = (F.\mathsf{Gen}, F, F^{-1})$ be a trapdoor permutation over $\{0,1\}^k$. For $\ell, k_0, k_1 \in \mathbb{N}$ s.t. $k = \ell + k_0 + k_1$ define hash functions*

$$G \colon \{0,1\}^{k_0} \to \{0,1\}^{\ell+k_1} \quad , \qquad H : \{0,1\}^{\ell+k_1} \to \{0,1\}^{k_0} \quad .$$

Then the procedures in Figure 2.9 form a PKE scheme for plaintexts in $\{0,1\}^{\ell}$. We refer to the PKE scheme as OAEP. *The OAEP padding process is illustrated in Figure 2.10.*

Proc $\mathsf{OAEP.Gen}$
$(ek, td) \leftarrow_{\$} F.\mathsf{Gen}$
$pk := ek,\ sk := td$
Return (pk, sk)

Proc $\mathsf{OAEP.Enc}_{pk}(m)$
$r \leftarrow_{\$} \{0,1\}^{k_0}$
$s \leftarrow m \,\|\, 0^{k_1} \oplus G(r)$
$t \leftarrow r \oplus H(s)$
$c \leftarrow F_{ek}((s,t))$
Return c

Proc $\mathsf{OAEP.Dec}_{sk}(c)$
$(s, t) \leftarrow F_{td}^{-1}(c)$
$r \leftarrow t \oplus H(s)$
$\mu \leftarrow s \oplus G(r)$
Parse μ as $m \,\|\, \rho \in \{0,1\}^{\ell} \times \{0,1\}^{k_1}$
If $\rho \neq 0^{k_1}$:
    Return $\perp$
Else:
    Return m

Figure 2.9: Construction of PKE OAEP from a TDP $\mathcal{T}$ and two hash functions G, H.

Note that, instead of employing a MAC as Construction 2.2.5 did, the OAEP transform relies on a 0-padding 0^{k_1} to serve as integrity protection.

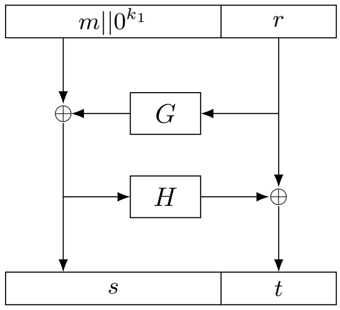

Figure 2.10: The OAEP padding process.

2.3.3 Selective Opening Security of OAEP

We prove that OAEP is SIM-SO-CCA secure in the random oracle model, assuming the partial-domain one-wayness of the trapdoor permutation $\mathcal{T}$.

Theorem 2.3.4 *Let $\mathcal{T} = (F.\mathsf{Gen}, F, F^{-1})$ be a TDP and G, H be hash functions as specified in Construction 2.3.3.*

If $\mathcal{T}$ is $(\tau_{pd\text{-}ow}, \varepsilon_{pd\text{-}ow})$-partial-domain one-way secure then PKE scheme OAEP *is $(\tau_{so\text{-}cca}, q_d, q_g, q_h, \varepsilon_{so\text{-}cca})$-SIM-SO-CCA secure where*

$$\tau_{so\text{-}cca} = \tau_{pd\text{-}ow} - \mathcal{O}(q_d \cdot (q_g + n) \cdot (q_h + n)) \ ,$$

$$\varepsilon_{so\text{-}cca}(n) \leq n \cdot \left(q_h \cdot \varepsilon_{pd\text{-}ow} + q_g \cdot \left(2^{-k_0} + 2^{-\ell-k_1}\right) + n \cdot 2^{-k_0}\right) + q_d \cdot \left(2^{-k_1} + q_g \cdot 2^{-k_0}\right) \ ,$$

and G, H are modeled as random oracles that an attacker may query at most q_g (resp. q_h) times.

Proof Sketch We prove Theorem 2.3.4 in a sequence of experiments, starting with the $\mathsf{r\text{-}SO\text{-}CCA_{OAEP}}$ experiment. We gradually modify the experiments, until a simulator run in the i-SO-CCA experiment can simulate the r-SO-CCA experiment for any SIM-SO-CCA adversary $\mathcal{A}_{so}$. Again, the simulator's strategy is to create 'non-committing' ciphertexts $c_1, \ldots, c_n$ which can then be opened to any plaintext m_i when $\mathcal{A}_{so}$ queries $\textsc{Open}(i)$.

We sketch the sequence of experiments that we employ in the proof of Theorem 2.3.4: In a first step, we replace the original decryption procedure that uses the real trapdoor td with an equivalent (up to a small error probability) decryption procedure. The new decryption procedure does not require td and is able to decrypt ciphertexts by examining the sequence of random oracle queries made by adversary $\mathcal{A}_{so}$. Here we use that $\mathcal{A}$ is not able (except for some small probability) to create a new valid ciphertext $c = F_{ek}((s,t))$, unless it queries $\mathrm{H}(s)$ and $\mathrm{G}(H(s) \oplus t)$. However, in this case the experiment is able to decrypt c by exhaustive search through all queries to H and G made by $\mathcal{A}$.

However, as we show, assuming that $\mathcal{T}$ is partial-domain one-way, it is unlikely to happen that $\mathcal{A}_{so}$ asks $H(s_i)$ before $\textsc{Open}(i)$.

Finally, we conclude with the observation that from $\mathcal{A}_{so}$'s point of view all values of $H(s_i)$ remain equally likely until $\textsc{Open}(i)$ is asked, which implies also that it is very unlikely that $\mathcal{A}_{so}$ ever queries $\mathrm{G}(t_i \oplus H(s_i))$ before $\textsc{Open}(i)$. This in turn means that the later simulator does not have to commit to a particular value of $G(t_i \oplus H(s_i))$, and thus not to a particular plaintext $m_i \,\|\, 0^{k_1} = s_i \oplus G(t_i \oplus H(s_i))$, before $\textsc{Open}(i)$ is asked.

We proceed with the detailed proof.

Proof of Theorem 2.3.4. Let $\mathcal{A}_{so} = (\mathcal{A}_{so,1}, \mathcal{A}_{so,2})$ be an adversary against the $(\tau_{so\text{-}cca}, q_d, q_g, q_h, \varepsilon_{so\text{-}cca})$-SIM-SO-CCA security of OAEP.

Experiment Exp_0. Experiment Exp_0 given in Figure 2.11 constitutes the $\mathsf{r\text{-}SO\text{-}CCA}$ experiment adjusted for PKE scheme OAEP. Note that random oracles G and H are implemented by lazy sampling. To this end, the experiment maintains four lists

$$L_G \subseteq \{0,1\}^{k_0} \times \{0,1\}^{\ell+k_1} \qquad L_H \subseteq \{0,1\}^{\ell+k_1} \times \{0,1\}^{k_0}$$
$$L_G^{\mathcal{A}} \subseteq \{0,1\}^{k_0} \qquad L_H^{\mathcal{A}} \subseteq \{0,1\}^{\ell+k_1}$$

which are initialized as empty in line 02 (resp. 03). Note that we explicitly updates these lists as queries are issued.

To simulate the random oracle G, the experiment uses the *internal* procedure $\mathrm{G}_{\mathsf{int}}$ (lines 29 – 32), which uses list L_G to ensure consistency of random oracle responses (see line 31). Adversary $\mathcal{A}_{so}$ does not have direct access to procedure $\mathrm{G}_{\mathsf{int}}$ but only via procedure G, which stores all values r queried by $\mathcal{A}_{so}$ in an additional list $L_G^{\mathcal{A}}$.

Random oracle H is implemented similarly, with procedures $\mathrm{H}_{\mathsf{int}}$ and H, using lists L_H and $L_H^{\mathcal{A}}$.

Clearly, we have $\Pr\left[\mathsf{r\text{-}SO\text{-}CCA}^{\mathcal{A}_{so}}(n) \Rightarrow 1\right] = \Pr\left[\mathsf{Exp}_0^{\mathcal{A}_{so}}(n) \Rightarrow 1\right]$.

Experiment Exp_1. Experiment Exp_1 proceeds exactly as Exp_0, except for decryption queries that are processed with a new oracle $\mathrm{OAEP.}\textsc{Dec}_1$ as given in Figure 2.13.

Claim 2.3.5 It holds

$$\left|\Pr\left[\mathsf{Exp}_0^{\mathcal{A}_{so}}(n) \Rightarrow 1\right] - \Pr\left[\mathsf{Exp}_1^{\mathcal{A}_{so}}(n) \Rightarrow 1\right]\right| \le q_d \cdot \left(2^{-k_1} + q_g \cdot 2^{-k_0}\right) \quad .$$

Exp $\mathsf{Exp}_0^{\mathcal{A}_{so}}(n)$
$\mathcal{I} \leftarrow \emptyset$; $\mathbf{c} \leftarrow \emptyset$
$L_G \leftarrow \emptyset$; $L_H \leftarrow \emptyset$
$L_G^{\mathcal{A}} \leftarrow \emptyset$; $L_H^{\mathcal{A}} \leftarrow \emptyset$
$(ek, td) \leftarrow_\$ F.\mathsf{Gen}$
$(\mathfrak{D}, st) \leftarrow_\$ \mathcal{A}_{so,1}^{\mathrm{G,H,OAEP.D{\scriptsize EC}}}(ek, n)$
$\mathbf{m} \leftarrow_\$ \mathfrak{D}$
For $i \leftarrow 1$ to n:
    $r_i \leftarrow_\$ \{0,1\}^{k_0}$
    $s_i \leftarrow m_i \| 0^{k_1} \oplus \mathrm{G}_{\mathsf{int}}(r_i)$
    $t_i \leftarrow r_i \oplus \mathrm{H}_{\mathsf{int}}(s_i)$
    $c_i \leftarrow F_{ek}((s_i, t_i))$
$\mathbf{c} \leftarrow (c_1, \ldots, c_n)$
$out \leftarrow_\$ \mathcal{A}_{so,2}^{\mathrm{G,H,O{\scriptsize PEN},OAEP.D{\scriptsize EC}}}(\mathbf{c})$
Stop with $\mathsf{Pred}(\mathfrak{D}, \mathbf{m}, \mathcal{I}, out)$

Oracle OAEP.Dec(c)
If $c \in \mathbf{c}$: Abort
$(s,t) \leftarrow F_{td}^{-1}(c)$
$r \leftarrow t \oplus \mathrm{H}_{\mathsf{int}}(s)$
$m \| \rho \leftarrow s \oplus \mathrm{G}_{\mathsf{int}}(r)$
If $\rho \neq 0^{k_1}$:
    Return $\perp$
Else:
    Return m

Oracle Open(i)
$\mathcal{I} \leftarrow \mathcal{I} \cup \{i\}$
Return (m_i, r_i)

Oracle G(r)
$L_G^{\mathcal{A}} \leftarrow L_G^{\mathcal{A}} \cup \{r\}$
Return $\mathrm{G}_{\mathsf{int}}(r)$

Oracle H(s)
$L_H^{\mathcal{A}} \leftarrow L_H^{\mathcal{A}} \cup \{s\}$
Return $\mathrm{H}_{\mathsf{int}}(s)$

Internal Proc $G_{\mathsf{int}}(r)$
If $(r, \cdot) \notin L_G$:
    $h_r \leftarrow_\$ \{0,1\}^{\ell+k_1}$
    $L_G \leftarrow L_G \cup (r, h_r)$
Return h_r

Internal Proc $\mathrm{H}_{\mathsf{int}}(s)$
If $(s, \cdot) \notin L_H$:
    $h_s \leftarrow_\$ \{0,1\}^{k_0}$
    $L_H \leftarrow L_H \cup (s, h_s)$
Return h_s

Figure 2.11: Procedures of experiment $\mathsf{Exp}_0(n)$ instantiating the r-SO-CCA experiment with PKE OAEP. We abstain from formally defining pk as ek and sk as td. Internal procedures $\mathrm{H}_{\mathsf{int}}$ and $\mathrm{G}_{\mathsf{int}}$ are only available to the experiment.

Oracle OAEP.Dec$_1(c)$ (Exp_1 – Exp_5)
If $c \in \mathbf{c}$: Abort
For all $(r, h_r, s, h_s) \in L_G \times L_H$:
    If $\begin{pmatrix} c = F_{ek}(s, r \oplus h_s) \\ \wedge\ s \oplus h_r = m \| 0^{k_1} \end{pmatrix}$:
        Return m
Return $\perp$

Figure 2.12: Replacement oracle OAEP.Dec$_1$ as used in Exp_1 – Exp_5.

For loop ($\mathsf{Exp}_2 - \mathsf{Exp}_5$)

42 For $i \leftarrow 1$ to n:

43 $\quad s_i \leftarrow_\$ \{0,1\}^{\ell+k_1}$, $t_i \leftarrow_\$ \{0,1\}^{k_0}$

44 $\quad c_i \leftarrow F_{ek}((s_i, t_i))$

45 $\quad r_i \leftarrow \mathrm{H}(s_i) \oplus t_i$

46 $\quad$ If $r_i \in L_G$: Abort

47 $\quad h_{r_i} \leftarrow s_i \oplus m_i \,\|\, 0^{k_1}$ // $\mathsf{Exp}_2 - \mathsf{Exp}_4$

48 $\quad L_G \leftarrow L_G \cup \{(r_i, h_{r_i})\}$ // $\mathsf{Exp}_2 - \mathsf{Exp}_4$

Figure 2.13: New For loop replacing lines 07 - 11 in Figure 2.11 in experiments Exp_2 – Exp_5. Lines 47 and 48 are removed in experiment Exp_5.

Proof of Claim 2.3.5. Experiment Exp_1 is indistinguishable from experiment Exp_0 unless $\mathcal{A}_{so}$ queries OAEP.DEC(c) where OAEP.DEC(c) $\neq$ OAEP.DEC$_1$(c). Observe that this can only hold if $\mathcal{A}_{so}$ queries OAEP.DEC$_1$(c). Now let $(s,t) \leftarrow F_{td}^{-1}(c)$, such that

$$\Big((s,\cdot) \notin L_H \quad \vee \quad (t \oplus H(s), \cdot) \notin L_G\Big) \quad \wedge \quad G(t \oplus H(s)) \oplus s = m \,\|\, 0^{k_1} \ .$$

Consider a single decryption query $c = F_{ek}((s,t))$. Assume $(s,\cdot) \notin L_H$. Then $H(s)$ is uniform and independent from $\mathcal{A}_{so}$'s view. Hence the probability that there exists $(r,\cdot) \in L_G$ such that $r = H(s) \oplus t$ is at most $q_g \cdot 2^{-k_0}$.

Assume $(r,\cdot) \notin L_G$. Then $G(r)$ is uniform and independent from $\mathcal{A}_{so}$'s view, thus the probability that $G(r) \oplus s = m || 0^{k_1}$ has the correct syntax is at most 2^{-k_1}.

Hence, taken over all at most q_d decryption queries we have

$$\left|\Pr\left[\mathsf{Exp}_0^{\mathcal{A}_{so}}(n) \Rightarrow 1\right] - \Pr\left[\mathsf{Exp}_1^{\mathcal{A}_{so}}(n) \Rightarrow 1\right]\right| \leq q_d \cdot \left(2^{-k_1} + q_g \cdot 2^{-k_0}\right) \ .$$

■

Note that procedure OAEP.DEC$_1$ does not require the trapdoor td to perform decryption.

Experiment Exp_2. In experiment Exp_2 we modify how the challenge ciphertexts $\mathbf{c}$ that are fed to $\mathcal{A}_{so,2}$ are computed. To this end, we replace the For loop in lines 07 - 11 in Figure 2.12 by the new instructions given in Figure 2.13. Note that this procedure first samples (s_i, t_i) uniformly random, then computes $c_i = F(ek, (s_i, t_i))$, and finally programs the random oracle G such that c_i decrypts to m_i.

Claim 2.3.6 It holds that

$$\left|\Pr\left[\mathsf{Exp}_1^{\mathcal{A}_{so}}(n) \Rightarrow 1\right] - \Pr\left[\mathsf{Exp}_2^{\mathcal{A}_{so}}(n) \Rightarrow 1\right]\right| \leq n \cdot (q_g + n) \cdot 2^{-k_0} \ .$$

Proof of Claim 2.3.6. Let Abort denote the event that Exp_2 aborts in line 46. We show that experiments Exp_1 and Exp_2 are identical until Abort. Note that the new encryption first defines $r_i \leftarrow \mathrm{H}(s_i) \oplus t_i$ for uniformly random $t_i \leftarrow_\$ \{0,1\}^{k_0}$. Thus, r_i is distributed uniformly over $\{0,1\}^{k_0}$, exactly as in experiment Exp_1.

Now, assume that it is not aborted in line 46. Hence $r_i \notin L_G$. It follows that hash function G is programmed such that $G(r_i) = h_{r_i} = s_i \oplus m_i \| 0^{k_1}$. Since s_i is uniformly distributed, so is $G(r_i)$, exactly as in experiment Exp_1. Thus, the new For loop is a perfect simulation of the old one conditioned on Abort not happening.

Note that the procedure terminates only if $r_i \in L_G$. Since for all $i \in [n]$ value s_i is uniform, so is r_i. Thus, Abort happens with probability at most $n \cdot (q_g + n) \cdot 2^{-k_0}$. ■

Experiment Exp_3. We add an abort condition to the Open oracle. See line 50 in Figure 2.14. That is, experiment Exp_3 proceeds exactly like Exp_2 but aborts if $\mathcal{A}_{so}$ for any $i \in [n]$ queries $\mathrm{H}(s_i)$ but did not query $\mathrm{Open}(i)$.

Claim 2.3.7 There exists an adversary $\mathcal{A}_{pd\text{-}ow}$ that breaks the $(\tau_{pd\text{-}ow}, \varepsilon_{pd\text{-}ow})$-partial domain one-way security of $\mathcal{T}$ where

$$\tau_{pd\text{-}ow} \approx \tau_{so\text{-}cca} + \mathcal{O}(q_d \cdot (q_g + n) \cdot (q_h + n)) \ ,$$

and

$$\varepsilon_{pd\text{-}ow} \geq \frac{1}{n \cdot q_h} \cdot \left| \Pr\left[\mathsf{Exp}_2^{\mathcal{A}_{so}}(n) \Rightarrow 1\right] - \Pr\left[\mathsf{Exp}_3^{\mathcal{A}_{so}}(n) \Rightarrow 1\right]\right| \ .$$

Proof of Claim 2.3.7. Let Abort denote the event that experiment Exp_3 aborts in line 50 (Figure 2.14). Clearly, experiments Exp_2 and Exp_3 are identical until Abort happens and it suffices to bound Pr[Abort].

We construct adversary $\mathcal{A}_{pd\text{-}ow}$ against the partial-domain one-wayness of $\mathcal{T}$. Adversary $\mathcal{A}_{pd\text{-}ow}$ is run on (ek, y) where $y = F_{ek}((s,t))$ for $(s,t) \leftarrow_\$ \{0,1\}^{\ell + k_1} \times \{0,1\}^{k_0}$. It samples indices $i^* \leftarrow_\$ [n]$, $q^* \leftarrow_\$ [q_h]$ and calls $\mathcal{A}_{so,1}(ek, n)$. When $\mathcal{A}_{so,1}$ outputs $\mathfrak{D}$, adversary $\mathcal{A}_{pd\text{-}ow}$ samples plaintexts from $\mathfrak{D}$ and encrypts them as in experiment Exp_3 except for c_{i^*} that is set to $c_{i^*} \leftarrow y$.

When $\mathcal{A}$ makes its q^{*th} query to H with input s^*, $\mathcal{A}_{pd\text{-}ow}$ outputs s^* and terminates.

Analysis Note that c_j is correctly distributed due to the changes introduced in experiment Exp_2. Assume that Abort happens. Then, at some point in its execution, $\mathcal{A}_{so}$ makes the *first* query $\mathrm{H}(s')$ such that $s' = s_i$ is a partial-domain preimage of some c_i. With probability $1/q_h$ it holds that $s^* = s_i$. Moreover, with probability $1/n$ we have $i = i^*$. In this case $\mathcal{A}_{pd\text{-}ow}$ obtains the partial preimage $s = s_j$ of $y = c_j$.

Oracle Open(i) (Exp$_3$ – Exp$_5$)

49 $\mathcal{I} \leftarrow \mathcal{I} \cup \{i\}$

50 If $s_i \in L_H^{\mathcal{A}}$: Abort // Exp$_3$ – Exp$_5$

51 If $r_i \in L_G^{\mathcal{A}}$: Abort // Exp$_4$ – Exp$_5$

52 $h_{r_i} \leftarrow s_i \oplus m_i \| 0^{k_1}$ // Exp$_5$

53 $L_G \leftarrow L_G \cup \{(r_i, h_{r_i})\}$ // Exp$_5$

54 Return (m_i, r_i)

Figure 2.14: New Open oracle used from experiment Exp$_3$ onwards.

Thus, if Abort happens and $\mathcal{A}_{pd\text{-}ow}$ guessed $i^* \in [n]$ and $q^* \in [q_h]$ correctly, then it breaks the partial-domain security of $\mathcal{T}$. Hence we obtain $\Pr[\text{Abort}] \leq n \cdot q_h \cdot \varepsilon_{pd\text{-}ow}$. The claim on $\varepsilon_{pd\text{-}ow}$ follows from rearranging.

The running time of $\mathcal{A}_{pd\text{-}ow}$ consists essentially of the running time of $\mathcal{A}_{so}$, plus the time needed to answer decryption queries, which is $\mathcal{O}((q_g + n) \cdot (q_h + n))$ per query. Thus, the total overhead in running time is $\mathcal{O}(q_d \cdot (q_g + n) \cdot (q_h + n))$. ■

Note that in experiment Exp$_3$ there is no $i \in \mathcal{I}$ such that $\mathcal{A}_{so}$ queries $\mathrm{H}(s_i)$ (as the experiment would abort in line 50).

Experiment Exp$_4$. We add another abort condition to the Open oracle. See line 51 in Figure 2.14. Experiment Exp$_4$ aborts if for any $i \in [n]$ adversary $\mathcal{A}_{so}$ queries $\mathrm{G}(r_i)$ before querying Open(i).

Claim 2.3.8 It holds $\left|\Pr\left[\mathsf{Exp}_3^{\mathcal{A}_{so}}(n) \Rightarrow 1\right] - \Pr\left[\mathsf{Exp}_4^{\mathcal{A}_{so}}(n) \Rightarrow 1\right]\right| \leq n \cdot q_g \cdot 2^{-\ell-k_1}$.

Proof of Claim 2.3.8. Due to the abort condition in line 50 introduced in the previous experiment Exp$_3$ adversary $\mathcal{A}_{so}$ never queries $\mathrm{H}(s_i)$ before querying Open(i). Let Abort denote the event that experiment Exp$_4$ aborts in line 51. Clearly, experiments Exp$_3$ and Exp$_4$ are identical until Abort happens. Thus, for all $i \notin \mathcal{I}$, $H(s_i)$ is uniformly random and independent of $\mathcal{A}_{so}$'s view. Therefore, all $r_i = t_i \oplus H(s_i)$ are uniformly random and independent of $\mathcal{A}_{so}$'s view. Because $\mathcal{A}_{so}$ issues at most q_g queries to G, and $1 \leq i \leq n$ we have $\Pr[\text{Abort}] \leq n \cdot q_g \cdot 2^{-\ell-k_1}$. ■

Experiment Exp$_5$. We move two lines of code. Precisely, lines 47 and 48 (see Figure 2.13) are removed from the encryption For loop to the Open oracle (see lines 52, 53 in Figure 2.14).

Claim 2.3.9 It holds $\Pr\left[\mathsf{Exp}_4^{\mathcal{A}_{so}}(n) \Rightarrow 1\right] = \Pr\left[\mathsf{Exp}_5^{\mathcal{A}_{so}}(n) \Rightarrow 1\right]$.

Proof of Claim 2.3.9. Note that experiment Exp_4 aborts if $\mathcal{A}_{so}$ queries $\mathrm{G}(r_i)$ before querying $\textsc{Open}(i)$. Thus, there is no need to define the hash value $G(r_i)$ before $\textsc{Open}(i)$ is asked. Therefore we can move the definition of $G(r_i)$ from the For loop to the $\textsc{Open}$ oracle.

This modification is completely oblivious to $\mathcal{A}_{so}$, which implies the claim. ∎

Claim 2.3.10 There exists a simulator $\mathcal{S} = (\mathcal{S}_1, \mathcal{S}_2)$ with roughly the same running time as $\mathcal{A}_{so}$ such that

$$\Pr\left[\mathsf{Exp}_5^{\mathcal{A}_{so}}(n) \Rightarrow 1\right] = \Pr\left[\mathsf{i\text{-}SO\text{-}CCA}^{\mathcal{S}}(n) \Rightarrow 1\right] \quad .$$

Proof of Claim 2.3.10. Note that in experiment Exp_5 plaintexts $(m_1, \ldots, m_n)$ are sampled after $\mathcal{A}_{so,1}$ outputs $\mathfrak{D}$ but only used in the $\textsc{Open}$ oracle. This allows us to construct a simulator, whose instructions are described in Figure 2.15. Note that the view of $\mathcal{A}_{so}$ when interacting with the simulator is *identical* to its view when interacting with experiment Exp_5. The claim follows.

∎

The claim follows from collecting the results from Claims 2.3.5 to 2.3.10. ∎

Selective Opening Security of RSA-OAEP The most important application of the OAEP scheme is clearly the RSA-OAEP encryption scheme, as described in the PKCS#1 standard [JK03]. Therefore an interesting question is whether the RSA trapdoor permutation is a partial-domain secure trapdoor permutation in the sense of Definition 2.3.2. By applying lattice reduction techniques, Fujisaki *et al.* [FOPS01] have shown that indeed the partial-domain one-wayness of the RSA permutation is equivalent to the one-wayness of RSA.

Simulator $\mathcal{S}_1(n)$

$\mathcal{I} \leftarrow \emptyset$; $\mathbf{c} \leftarrow \emptyset$
$L_G \leftarrow \emptyset$; $L_H \leftarrow \emptyset$
$L_G^{\mathcal{A}} \leftarrow \emptyset$; $L_H^{\mathcal{A}} \leftarrow \emptyset$
$(ek, td) \leftarrow_{\$} F.\mathsf{Gen}$
$(\mathfrak{D}, st) \leftarrow_{\$} \mathcal{A}_{so,1}^{\mathrm{G,H,OAEP.D{\scriptsize EC}}}(ek, n)$
Return $\mathfrak{D}$

Simulator $\mathcal{S}_2^{\mathrm{O{\scriptsize PEN}}_{\mathcal{S}}}(|m_1|, \dots, |m_n|)$

For $i \leftarrow 1$ to n:
  $s_i \leftarrow_{\$} \{0,1\}^{\ell + k_1}$
  $t_i \leftarrow_{\$} \{0,1\}^{k_0}$
  $c_i \leftarrow F_{ek}((s_i, t_i))$
  $r_i \leftarrow \mathrm{H}(s_i) \oplus t_i$
  If $r_i \in L_G$: Abort
$\mathbf{c} \leftarrow (c_1, \dots, c_n)$
$out \leftarrow_{\$} \mathcal{A}_{so,2}^{\mathrm{G,H,O{\scriptsize PEN},OAEP.D{\scriptsize EC}}}(\mathbf{c})$
Stop with $\mathsf{Pred}(\mathfrak{D}, \mathbf{m}, \mathcal{I}, out)$

Oracle $\mathrm{O{\scriptsize PEN}}(i)$

$m_i \leftarrow \mathrm{O{\scriptsize PEN}}_{\mathcal{S}}(i)$
$\mathcal{I} \leftarrow \mathcal{I} \cup \{i\}$
If $s_i \in L_H^{\mathcal{A}}$: Abort
If $r_i \in L_G^{\mathcal{A}}$: Abort
$h_{r_i} \leftarrow s_i \oplus m_i \,\|\, 0^{k_1}$
$L_G \leftarrow L_G \cup \{(r_i, h_{r_i})\}$
Return (m_i, r_i)

Oracle $\mathrm{OAEP.D{\scriptsize EC}}_1(c)$

If $c \in \mathbf{c}$: Abort
For all $(r, h_r, s, h_s) \in L_G \times L_H$:
  If $\begin{pmatrix} c = F(ek, (s, r \oplus h_s)) \\ \wedge\ s \oplus h_r = m || 0^{k_1} \end{pmatrix}$:
    Return m
Return $\perp$

Oracle $G(r)$

$L_G^{\mathcal{A}} \leftarrow L_G^{\mathcal{A}} \cup \{r\}$
Return $G_{\mathsf{int}}(r)$

Oracle $H(s)$

$L_H^{\mathcal{A}} \leftarrow L_H^{\mathcal{A}} \cup \{s\}$
Return $H_{\mathsf{int}}(s)$

Internal Proc $G_{\mathsf{int}}(r)$

If $(r, h_r) \notin L_G$:
  $h_r \leftarrow_{\$} \{0,1\}^{\ell + k_1}$
  $L_G \leftarrow L_G \cup (r, h_r)$
Return h_r

Internal Proc $H_{\mathsf{int}}(s)$

If $(s, h_s) \notin L_H$:
  $h_s \leftarrow_{\$} \{0,1\}^{k_0}$
  $L_H \leftarrow L_H \cup (s, h_s)$
Return h_s

Figure 2.15: Instructions of simulator $\mathcal{S}$ to implement the $\mathsf{r\text{-}SO\text{-}CCA}$ experiment for $\mathcal{A}_{so}$. We denote the open oracle provided by the ideal experiment for $\mathcal{S}_2$ with $\mathrm{O{\scriptsize PEN}}_{\mathcal{S}}$.

2.4 The Fujisaki-Okamoto Transformation

We move on to the last transformation covered in this chapter. The Fujisaki-Okamoto transformation was proposed in [FO99] and excels through its generality. In [Pei14] it has successfully been applied to construct an efficient lattice-based cryptosystem. Remarkably, other major transformations to IND-CCA secure PKE schemes like DHIES [BR97, ABR01], REACT [OP01], OAEP [BR95] could not be applied in this scenario [Pei14].

Provable Security of the FO Transformation The Fujisaki-Okamoto transformation employs two hash functions to strengthen a PKE scheme. If instantiated with one-way secure PKE scheme under with plaintexts 'spread well' (taken over the randomness (see Definition 2.4.2)), the transformed scheme is IND-CCA secure in the random oracle model [FO99, FO13].

In our analysis we consider a slightly modified transformation that was given in the journal version [FO13]. Further, the journal version clarifies that the two conditions on which the decryption algorithm aborts (see Figure 2.17) should trigger the same error symbol to be output. Joye, Quisquater, and Yung [JQY01] have shown that such a behavior is crucial for security. In fact, it has been practically exploited that in some implementations the output of the error symbol when generated by the first abort condition usually appears earlier than that of the second. [ST02]

2.4.1 One-wayness and Ciphertext Distribution

In this section PKE will always denote a PKE scheme ($\mathsf{PKE.Gen}, \mathsf{PKE.Enc}, \mathsf{PKE.Dec}$) for a finite plaintext space $\mathcal{M}$. The randomness space of $\mathsf{PKE.Enc}$ is denoted by $\mathcal{R}$.

Definition 2.4.1 (OW secure PKE). We say PKE is (τ, ε)-*OW secure* if for all τ-time adversaries that interact with the OW experiment from Figure 2.16 we have

$$\Pr\left[\mathsf{OW}^{\mathcal{A}} \Rightarrow 1\right] \leq \varepsilon \ .$$

Exp $\mathsf{OW}^{\mathcal{A}}$
01 $(pk, sk) \leftarrow_{\$} \mathsf{PKE.Gen}$
02 $m \leftarrow_{\$} \mathcal{M}$
03 $c \leftarrow_{\$} \mathsf{PKE.Enc}_{pk}(m)$
04 $m' \leftarrow_{\$} \mathcal{A}(pk, c)$
05 Stop with $(m =_? m')$

Figure 2.16: One-way experiment OW as used in Definition 2.4.1.

Informally, we say that PKE is *OW* (secure) if ε is small for all efficient adversaries.

Definition 2.4.2 (γ-spread PKE). Let $m \in \mathcal{M}$ and $(pk, sk) \in [\mathsf{PKE.Gen}]$. We define the *min-entropy* $\gamma_{pk}(m)$ of $\mathsf{PKE.Enc}_{pk}(m)$ as

$$\gamma_{pk}(m) := -\log \max_{c \in \{0,1\}^*} \left\{ \Pr_{r \leftarrow_{\$} \mathcal{R}}[c = \mathsf{PKE.Enc}_{pk}(m; r)] \right\} \ .$$

We say PKE is γ-*spread* if for all $(pk, sk) \in [\mathsf{PKE.Gen}]$ and all $m \in \mathcal{M}$ we have $\gamma_{pk}(m) \geq \gamma$.

Note that for any γ-spread PKE scheme appending γ' uniform random bits to the ciphertexts immediately ensures that the scheme is $(\gamma + \gamma')$-spread at the cost of longer ciphertexts as mentioned in [FO99]. Hence, as we show, SIM-SO-CCA secure PKE (in the ROM) exists assuming the existence of one-way secure PKE.

2.4.2 The Fujisaki-Okamoto Transformation

We describe the Fujisaki-Okamoto transformation as given in [FO13].

Construction 2.4.3 (Fujisaki-Okamoto transformation). *Let* PKE *be a PKE for finite plaintext space $\mathcal{M}$ and ciphertext space $\mathcal{C}$. Let $\mathcal{R}$ denote the finite randomness space of* PKE.Enc.

Let

$$\mathcal{M}_{\mathsf{FO}} := \{0,1\}^{\ell} \ , \qquad \mathcal{R}_{\mathsf{FO}} := \mathcal{M} \ , \qquad \mathcal{C}_{\mathsf{FO}} := \mathcal{C} \times \{0,1\}^{\ell} \ .$$

Let G, H be hash functions where

$$G\colon \mathcal{R}_{\mathsf{FO}} \to \{0,1\}^{\ell} \ , \qquad H\colon \mathcal{R}_{\mathsf{FO}} \times \{0,1\}^{\ell} \to \mathcal{R} \ .$$

Then the procedures in Figure 2.17 form a PKE scheme for plaintext space $\mathcal{M}_{\mathsf{FO}}$. We refer to the PKE scheme as FO.

Proc FO.Gen
01 $(pk, sk) \leftarrow \mathsf{PKE.Gen}$
02 Return (pk, sk)

Proc $\mathsf{FO.Enc}_{pk}(m)$
03 $r \leftarrow_{\$} \mathcal{R}_{\mathsf{FO}}$
04 $c^{(2)} \leftarrow m \oplus G(r)$
05 $h \leftarrow H(r, c^{(2)})$
06 $c^{(1)} \leftarrow \mathsf{PKE.Enc}_{pk}(r; h)$
07 Return $\langle c^{(1)}, c^{(2)} \rangle$

Proc $\mathsf{FO.Dec}_{sk}(\langle c^{(1)}, c^{(2)} \rangle)$
08 $\hat{r} \leftarrow \mathsf{PKE.Dec}_{sk}(c^{(1)})$
09 If $\hat{r} \notin \mathcal{R}_{\mathsf{FO}}$:
10 Return $\perp$
11 $\hat{h} \leftarrow H(\hat{r}, c^{(2)})$
12 $\hat{c}^{(1)} \leftarrow \mathsf{PKE.Enc}_{pk}(\hat{r}; \hat{h})$
13 If $c^{(1)} \neq \hat{c}^{(1)}$:
14 Return $\perp$
15 $m \leftarrow c^{(2)} \oplus G(\hat{r})$
16 Return m

Figure 2.17: Fujisaki-Okamoto Transformation for PKE scheme PKE.

For clarity, the FO encryption process is illustrated in Figure 2.18.

Note that the decryption procedure of the FO transformation performs a check of 'well-formedness' by re-encryption in line 12.

We instantiate the symmetric encryption with the one-time pad (see Section 2.2.5) and interpret the FO transformation as a transformation of PKE schemes. In its full generality, hash value $G(r)$ serves as key for a DEM.

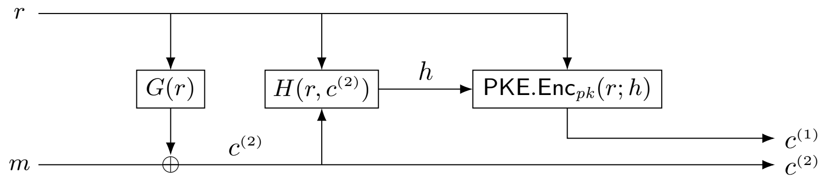

Figure 2.18: Structure of FO encryption. We have $\langle c^{(1)}, c^{(2)} \rangle \leftarrow \mathsf{FO.Enc}_{pk}(m; r)$.

2.4.3 Selective Opening Security of the Fujisaki-Okamoto Transformation

Theorem 2.4.4 *Let* PKE *be a PKE and let* FO *denote the PKE scheme obtained when instantiating Construction 2.4.3 with* PKE*.*

If PKE *is* $(\tau_{ow}, \varepsilon_{ow})$*-OW secure and* γ*-spread, then* FO *is* $(\tau_{so\text{-}cca}, q_d, q_{hash}, \varepsilon_{so\text{-}cca})$*-SIM-SO-CCA secure where*

$$\tau_{so\text{-}cca} = \tau_{ow} - \mathcal{O}(q_d \cdot q_{hash}), \quad \varepsilon_{so\text{-}cca}(n) \leq n \cdot \left(q_d \cdot 2^{-\gamma} + q_{hash} \cdot \left(\frac{1}{|\mathcal{R}| - q_{hash}} + \varepsilon_{ow} \right) \right),$$

where G and H are modeled as random oracles that may be queried jointly at most q_{hash} times.

Proof Sketch The idea is similar to the previous two proofs of SIM-SO-CCA secure PKE in Sections 2.2 and 2.3. Again, we proceed in a sequence of experiments:

After the first modification the experiment will be capable of answering (almost all) decryption queries without the secret key. Next, a statistical argument ensures that for all $i \in [n]$ hash functions $H(r_i, \cdot)$ and $G(r_i)$ were not evaluated when a SIM-SO-CCA adversary $\mathcal{A}_{so}$ outputs $\mathfrak{D}$ and expects encryptions of challenge plaintexts. Thus, we can rewrite the encryption of challenge plaintexts by moving the programming of G and the H to oracles G, H and Open procedure. Further, we use PKE's one-wayness to argue that $\mathcal{A}_{so,2}(c_1, \ldots, c_n)$ is unlikely to query $\mathrm{H}(r_i, c_i^{(2)})$ or $\mathrm{G}(r_i)$ for any $i \in [n]$.

As a last step, we construct a simulator $\mathcal{S}$ suitable to run $\mathcal{A}_{so}$ in a simulated r-SO-CCA experiment when $\mathcal{S}$ is run in the ideal experiment.

Proof of Theorem 2.4.4. Let $\mathcal{A}_{so} = (\mathcal{A}_{so,1}, \mathcal{A}_{so,2})$ denote an attacker against the $(\tau_{so\text{-}cca}, q_d, q_{hash}, \varepsilon_{so\text{-}cca})$-SIM-SO-CCA security of FO.

We continue with detailed descriptions of the experiments given in Figures 2.19 to 2.21.

Exp $\mathsf{Exp}_0(n) - \mathsf{Exp}_4(n)$

$\mathcal{I} \leftarrow \emptyset$; $\mathbf{c} \leftarrow \emptyset$
$L_G \leftarrow \emptyset$; $L_H \leftarrow \emptyset$
$(pk, sk) \leftarrow_\$ \mathsf{PKE.Gen}$
For $i \leftarrow 1$ to n:
  $r_i \leftarrow_\$ \mathcal{R}_{\mathsf{FO}}$
$(\mathfrak{D}, st) \leftarrow_\$ \mathcal{A}_{so,1}^{\mathrm{G,H,FO.Dec}}(pk, n)$
$\mathbf{m} \leftarrow_\$ \mathfrak{D}$
For $i \leftarrow 1$ to n:
  $c_i^{(2)} \leftarrow \mathrm{G}(r_i) \oplus m_i$ // $\mathsf{Exp}_0 - \mathsf{Exp}_2$
  $h_i \leftarrow \mathrm{H}(r_i, c_i^{(2)})$ // $\mathsf{Exp}_0 - \mathsf{Exp}_2$
  $c_i^{(1)} \leftarrow \mathsf{PKE.Enc}_{pk}(r_i; h_i)$ // $\mathsf{Exp}_0 - \mathsf{Exp}_2$
  $\sigma_i^g \leftarrow_\$ \{0,1\}^\ell$ // $\mathsf{Exp}_3 - \mathsf{Exp}_4$
  $c_i^{(2)} \leftarrow \sigma_i^g$ // $\mathsf{Exp}_3 - \mathsf{Exp}_4$
  $\sigma_i^h \leftarrow_\$ \mathcal{R}$ // $\mathsf{Exp}_3 - \mathsf{Exp}_4$
  $c_i^{(1)} \leftarrow \mathsf{PKE.Enc}_{pk}(r_i; \sigma_i^h)$ // $\mathsf{Exp}_3 - \mathsf{Exp}_4$
  $c_i \leftarrow \langle c_i^{(1)}, c_i^{(2)} \rangle$
$\mathbf{c} \leftarrow (c_1, \ldots, c_n)$
$out \leftarrow_\$ \mathcal{A}_{so,2}^{\mathrm{G,H,Open,FO.Dec}}(st, \mathbf{c})$
Stop with $\mathsf{Pred}(\mathfrak{D}, \mathbf{m}, \mathcal{I}, out)$

Oracle $\textsc{Open}(i)$

$\mathcal{I} \leftarrow \mathcal{I} \cup \{i\}$
$G(r_i) \leftarrow \sigma_i^g \oplus m_i$ // $\mathsf{Exp}_3 - \mathsf{Exp}_4$
$H(r_i, c_i) \leftarrow \sigma_i^h$ // $\mathsf{Exp}_3 - \mathsf{Exp}_4$
Return (m_i, r_i)

Figure 2.19: Sequence of experiments used in the proof of Theorem 2.4.4. Oracle FO.Dec is given in Figure 2.20. Hash oracles G and H are given in Figure 2.21.

Experiment Exp_0. Experiment Exp_0 constitutes the r-SO-CCA experiment adopted for PKE = FO. Further, hash functions G and H are implemented by lazy sampling. Additionally, we introduce a merely syntactical change: For all $i \in [n]$ the random coins r_i are sampled *before* $\mathcal{A}_{so,1}$ is started.

Clearly, we have $\Pr\left[\mathsf{r\text{-}SO\text{-}CCA}^{\mathcal{A}_{so}}(n) \Rightarrow 1\right] = \Pr\left[\mathsf{Exp}_0^{\mathcal{A}_{so}}(n) \Rightarrow 1\right]$.

Experiment Exp_1. We replace the decryption oracle $\mathrm{FO.Dec}_0$ by the new oracle $\mathrm{FO.Dec}_1$ given in Figure 2.20. For a decryption query $\langle c^{(1)} c^{(2)} \rangle$, instead of decrypting $c^{(1)}$ to obtain $\hat{r}$ and querying $\mathrm{H}(\hat{r}, c^{(2)})$, experiment Exp_1 aborts if $\mathcal{A}_{so}$ did not submit some tuple $(\hat{r}, c^{(2)})$ to H s.t. $c^{(1)} = \mathsf{PKE.Enc}_{pk}(\hat{r}; \mathrm{H}(\hat{r}, c^{(2)}))$. Otherwise the experiment retrieves $\hat{r}$ from L_H.

Oracle $\mathrm{FO.D\scriptstyle{EC}}_0(\langle c^{(1)}, c^{(2)}\rangle)$ (Exp_0)	**Oracle** $\mathrm{FO.D\scriptstyle{EC}}_1(\langle c^{(1)} c^{(2)}\rangle)$ ($\mathsf{Exp}_1 - \mathsf{Exp}_4$)
If $\langle c^{(1)}, c^{(2)}\rangle \in \mathbf{c}$: Abort	If $\langle c^{(1)}, c^{(2)}\rangle \in \mathbf{c}$: Abort
$\hat{r} \leftarrow \mathsf{PKE.Dec}_{sk}(c^{(1)})$	If $\nexists(\hat{r}, c^{(2)}, \hat{h}) \in L_{\mathrm{H}}$ s.t.
If $\hat{r} \notin \mathcal{R}_{\mathsf{FO}}$:	$c^{(1)} = \mathsf{PKE.Enc}_{pk}(\hat{r}; \hat{h})$:
Return $\perp$	Return $\perp$
$\hat{h} \leftarrow \mathrm{H}(\hat{r}, c^{(2)})$	Else:
If $c^{(1)} \neq \mathsf{PKE.Enc}_{pk}(\hat{r}; \hat{h})$:	Let $\hat{r}$ s.t. $(\hat{r}, c^{(2)}, \hat{h}) \in L_H$
Return $\perp$	$\wedge\ c^{(1)} = \mathsf{PKE.Enc}_{pk}(\hat{r}; \hat{h})$
$m \leftarrow c^{(2)} \oplus \mathrm{G}(\hat{r})$	$m \leftarrow c^{(2)} \oplus \mathrm{G}(\hat{r})$
Return m	Return m

Figure 2.20: Decryption oracles as used in the sequence of experiments given in Figure 2.19. Oracle $\mathrm{FO.D\scriptstyle{EC}}_0$ is used in experiment Exp_0, oracle $\mathrm{FO.D\scriptstyle{EC}}_1$ is used from experiment Exp_1 on.

Claim 2.4.5 It holds

$$\left|\Pr\left[\mathsf{Exp}_0^{\mathcal{A}_{so}}(n) \Rightarrow 1\right] - \Pr\left[\mathsf{Exp}_1^{\mathcal{A}_{so}}(n) \Rightarrow 1\right]\right| \leq n \cdot q_d \cdot 2^{-\gamma} \ .$$

Proof of Claim 2.4.5. Recall that a ciphertext $\langle c^{(1)}, c^{(2)}\rangle$ is *valid* if decryption does not result in $\perp$. That is, for $r \leftarrow \mathsf{PKE.Dec}_{sk}(c^{(1)})$ we have $r \in \mathcal{R}_{\mathsf{FO}}$ and $c^{(1)} = \mathsf{PKE.Enc}_{pk}(r; H(r, c^{(2)}))$.

Now consider decryption oracles $\mathrm{FO.D\scriptstyle{EC}}_0$ and $\mathsf{FO.Dec}_1$. We see that invalid ciphertexts are decrypted to $\perp$ in both procedures. Further, if a valid ciphertext is decrypted to $m \neq \perp$ in oracle $\mathrm{FO.D\scriptstyle{EC}}_1$ the same holds for $\mathrm{FO.D\scriptstyle{EC}}_0$. On the contrary, a valid ciphertext query $\langle c^{(1)}, c^{(2)}\rangle$ answered with $m \neq \perp$ by $\mathsf{FO.Dec}_0$ might result in a $\perp$ reply by $\mathrm{FO.D\scriptstyle{EC}}_1$. Precisely, it happens if $\mathcal{A}_{so}$ did not query $\mathrm{H}(\hat{r}, c)$ before calling $\mathrm{FO.D\scriptstyle{EC}}(\langle c^{(1)}, c^{(2)}\rangle)$, i.e., there is no entry $(\hat{r}, c^{(2)}, \cdot)$ in list L_{H}. Hence, the distance between experiments Exp_0 and Exp_1 is upper-bounded by the probability that $\mathcal{A}_{so}$ submits a *valid* $\langle c^{(1)}, c^{(2)}\rangle$ to $\mathrm{FO.D\scriptstyle{EC}}$ while $\mathrm{H}(\hat{r}, c)$ is still undefined.

We now show that $(r, c^{(2)}) \neq (r_i, c_i^{(2)})$ for all $i \in [n]$. $\mathcal{A}$ (without loss of generality) submits a ciphertext $\langle c^{(1)}, c^{(2)}\rangle \notin \{\langle c_i^{(1)}, c_i^{(2)}\rangle\}_{i\in[n]}$. If $c^{(2)} \neq c_i^{(2)}$ for all $i \in [n]$ the claim follows. Otherwise, assume that for some $i \in [n]$ we have $c^{(2)} = c_i^{(2)}$, then $c^{(1)} \neq c_i^{(1)}$. As $c^{(1)} \leftarrow \mathsf{PKE.Enc}_{pk}(r; H(r, c^{(2)}))$ it follows $(r, H(r, c^{(2)})) \neq (r_i, H(r_i, c_i^{(2)}))$. Thus, either $r \neq r_i$ or $H(r, c^{(2)}) \neq H(r_i, c_i^{(2)})$, implying $r \neq r_i$ since we assumed $c^{(2)} = c_i^{(2)}$. Hence, $H(r, c^{(2)})$ is independent of $H(r_i, c_i^{(2)})$ for all $i \in [n]$ and we can employ the γ-spreadness of PKE. Thereby, the probability of $\mathcal{A}_{so}$ submitting a valid decryption query $\langle c^{(1)}, c^{(2)}\rangle$ without querying $\mathrm{H}(\hat{r}, c^{(2)})$ is at most $n \cdot 2^{-\gamma}$ for a single decryption query.

The claim follows. ■

Note that $\mathrm{FO.D\scriptstyle{EC}}_1$ does not require knowledge of *sk* to process decryption queries.

Oracle $\mathrm{G}(t)$

40 If $t \in \{r_1, \ldots, r_n\}$: // $\mathcal{A}_{so,1}$: $\mathsf{Exp}_2 - \mathsf{Exp}_4$
41 Abort // $\mathcal{A}_{so,1}$: $\mathsf{Exp}_2 - \mathsf{Exp}_4$
42 Let $i \in [n]$ s.t. $t = r_i$ // $\mathcal{A}_{so,2}$: Exp_4
43 If $i \notin \mathcal{I}$: Abort // $\mathcal{A}_{so,2}$: Exp_4
44 $G(t) \leftarrow \sigma_i^g \oplus m_i$ // $\mathcal{A}_{so,2}$: $\mathsf{Exp}_3 - \mathsf{Exp}_4$
45 If $(t, \cdot) \notin L_G$:
46 $g_t \leftarrow_\$ \{0,1\}^\ell$
47 $G(t) \leftarrow g_t$
48 Return g_t

Oracle $\mathrm{H}(s_1, s_2)$

49 If $s_1 \in \{r_1, \ldots, r_n\}$: // $\mathcal{A}_{so,1}$: $\mathsf{Exp}_2 - \mathsf{Exp}_4$
50 Abort // $\mathcal{A}_{so,1}$: $\mathsf{Exp}_2 - \mathsf{Exp}_4$
51 Let $i \in [n]$ s.t. $s_1 = r_i$ // $\mathcal{A}_{so,2}$: $\mathsf{Exp}_3 - \mathsf{Exp}_4$
52 If $s_2 = c_i$: // $\mathcal{A}_{so,2}$: $\mathsf{Exp}_3 - \mathsf{Exp}_4$
53 If $i \notin \mathcal{I}$: Abort // $\mathcal{A}_{so,2}$: Exp_4
54 $H(s_1, s_2) \leftarrow \sigma_i^h$ // $\mathcal{A}_{so,2}$: $\mathsf{Exp}_3 - \mathsf{Exp}_4$
55 If $(s_1, s_2, \cdot) \notin L_H$:
56 $h_s \leftarrow_\$ \mathcal{R}$
57 $H(s_1, s_2) \leftarrow h_s$
58 Return h_s

Figure 2.21: Hash oracles G and H as part of the sequence of experiments from Figure 2.19.

Experiment Exp_2. We add abort conditions to the G and H oracles (see lines 40/41 and 49/50). If $\mathcal{A}_{so,1}$ queries $\mathrm{G}(r_i)$ or $\mathrm{H}(r_i, \cdot)$ for some $i \in [n]$, experiment Exp_2 aborts.

Claim 2.4.6 It holds

$$\left| \Pr\left[\mathsf{Exp}_0^{\mathcal{A}_{so}}(n) \Rightarrow 1\right] - \Pr\left[\mathsf{Exp}_1^{\mathcal{A}_{so}}(n) \Rightarrow 1\right] \right| \leq n \cdot \frac{q_{hash}}{|\mathcal{R}| - q_{hash}} \; .$$

Proof of Claim 2.4.6. Observe that for all $i \in [n]$ the value r_i is uniformly random from $\mathcal{A}_{so,1}$'s point of view. Experiment Exp_2 aborts if for any $i \in [n]$ adversary $\mathcal{A}_{so,1}$ queries $\mathrm{H}(r_i, \cdot)$ or $\mathrm{G}(r_i)$. Let us denote the respective event with ABORT. Now, $\Pr[\text{ABORT}]$ can be upper-bounded by the sum over the probability of aborting on the i^{th} hash query (to either G or H) conditioned on ABORT did not happen in the first $i-1$ hash queries. Hence,

$$\Pr[\text{ABORT}] \leq n \cdot \sum_{i=1}^{q_{hash}} \frac{1}{|\mathcal{R}_{\mathsf{FO}}| - (i-1)} \leq \frac{n \cdot q_{hash}}{|\mathcal{R}| - q_{hash}} \; .$$

■

Experiment Exp_3. We modify the encryption of challenge plaintexts. Instead of querying $\mathrm{H}(r_i, c_i^{(2)})$ (resp. $\mathrm{G}(r_i)$) we pick $\sigma_i^h \leftarrow \mathcal{R}$ (resp. $\sigma_i^g \leftarrow_\$ \{0,1\}^\ell$) uniformly at random (see lines 12, 14). The challenge ciphertexts are computed as $(c_i^{(1)}, c_i^{(2)}) = (\mathsf{PKE.Enc}_{pk}(r_i; \sigma_i^h), \sigma_i^g)$ (see lines 13, 15).

We accordingly program the hash functions (if $\mathcal{A}_{so}$ should query them) as $\mathrm{G}(r_i) \leftarrow \sigma_i^g \oplus m_i$ (line 44) and $\mathrm{H}(r_i, c_i^{(2)}) \leftarrow \sigma_i^h$ (lines 51, 52, 54). The same programming is performed when $\mathcal{A}_{so}$ queries $\textsc{Open}(i)$ (lines 21 and 22).

Claim 2.4.7 It holds

$$\Pr\left[\mathsf{Exp}_2^{\mathcal{A}_{so}}(n) \Rightarrow 1\right] = \Pr\left[\mathsf{Exp}_3^{\mathcal{A}_{so}}(n) \Rightarrow 1\right] .$$

Proof of Claim 2.4.7. Fix $i \in [n]$ and observe that during the For loop (line 08) values $H(r_i, c_i)$ and $G(r_i)$ are uniformly random. Thus, we can choose some uniform σ_i^h for encryption instead of evaluating $\mathrm{H}(r_i, c_i)$.

The same argument applies for G. Thus, value $G(r_i) \oplus m_i$ is uniform and we can replace it by some uniform σ_i^g. The additional instructions within G, H and $\textsc{Open}$ ensure that for all $i \in [n]$ values $\mathrm{H}(r_i, c_i)$ and $G(r_i)$ are programmed consistently. ∎

Observe that, from experiment Exp_3 on, for all $i \in [n]$ ciphertext $c_i = \langle c_i^{(1)}, c_i^{(2)} \rangle$ is independent of plaintext m_i when $\mathcal{A}_{so,2}$ is run on $\mathbf{c}$.

We now ensure that for all $i \in [n]$ the ciphertext $\langle c_i^{(1)}, c_i^{(2)} \rangle$ *remains* independent of m_i unless $\mathcal{A}_{so,2}$ queries $\textsc{Open}(i)$

Experiment Exp_4. We add abort conditions to the hash functions. Experiment Exp_4 aborts $\mathcal{A}_{so,2}$ if for any $i \in [n]$ it queries $\mathrm{H}(r_i, c_i^{(2)})$ or $\mathrm{G}(r_i)$ and did not call $\textsc{Open}(i)$. See lines 42/43 and 53.

Claim 2.4.8 There exists an adversary $\mathcal{A}_{ow}$ that breaks the $(\tau_{ow}, \varepsilon_{ow})$-OW security of PKE where $\tau_{ow} \approx \tau_{so\text{-}cca} + \mathcal{O}(q_d \cdot q_h)$ and

$$\varepsilon_{ow} \geq \frac{1}{n \cdot q_{hash}} \cdot \left| \Pr\left[\mathsf{Exp}_3^{\mathcal{A}_{so}}(n) \Rightarrow 1\right] - \Pr\left[\mathsf{Exp}_4^{\mathcal{A}_{so}}(n) \Rightarrow 1\right] \right| .$$

Proof of Claim 2.4.8. Let $\textsc{Abort}$ denote the event that a newly introduced Abort happens in experiment Exp_4. As experiments Exp_3 and Exp_4 are identical until $\textsc{Abort}$, we have $|\Pr[\mathsf{Exp}_3^{\mathcal{A}_{so}}] - \Pr[\mathsf{Exp}_4^{\mathcal{A}_{so}}]| \leq \Pr[\textsc{Abort}]$.

We construct adversary $\mathcal{A}_{ow}$. It is run on (pk, c^*). It samples $i^* \leftarrow_\$ [n]$, $q^* \leftarrow_\$ [q_{hash}]$ and invokes $\mathcal{A}_{so,1}(pk, n)$. On $\mathcal{A}_{so}$'s q^{*th} hash query, either $\mathrm{G}(t)$ or $\mathrm{H}(s_1, \cdot)$, adversary $\mathcal{A}_{ow}$ outputs t (resp. s_1) and halts.

When $\mathcal{A}_{so,1}$ outputs $\mathfrak{D}$, $\mathcal{A}_{ow}$ processes them as in experiment Exp_4 except for ciphertext $c_{i^*} \leftarrow (c^*, \sigma_i^g)$.

$\mathcal{A}_{ow}$ answers opening queries honestly unless $\mathcal{A}_{so,2}$ queries $\textsc{Open}(i^*)$ where $\mathcal{A}_{ow}$ aborts.

$\textsc{Analysis}$ Assume that $\textsc{Abort}$ happens. Then, with probability $1/n$ it will happen for $i = i^*$. In particular, $\mathcal{A}_{so}$ will not call $\textsc{Open}(i^*)$ and will query $\mathrm{G}(r_{i^*})$ or $\mathrm{H}(r_{i^*}, \cdot)$ where $r_{i^*} = \mathsf{PKE.Dec}_{sk}(c^{(2)})$. With probability $1/q_{hash}$, adversary $\mathcal{A}_{so}$ will make that query as its j^{*th}. Clearly, $\mathcal{A}_{ow}$ breaks OW security by returning r_i. The claim on ε_{ow} follows.

The running time of $\mathcal{A}_{ow}$ is at least the running time of $\mathcal{A}_{so}$. Further, $\mathcal{A}_{ow}$ simulates the decryption oracle as in experiment Exp_4. To this end, for each decryption query, it iterates over all entries in L_H, $|L_H| \leq q_{hash}$. That is, the overall overhead for answering decryption queries is $\mathcal{O}(q_d \cdot q_{hash})$. ■

Note that from now on for all $i \in [n]$ ciphertext c_i remains independent of the sampled plaintexts until $\mathcal{A}_{so}$ queries $\textsc{Open}(i)$.

Claim 2.4.9 There exists a simulator $\mathcal{S} = (\mathcal{S}_1, \mathcal{S}_2)$ such that

$$\Pr\left[\mathsf{Exp}_4^{\mathcal{A}_{so}}(n) \Rightarrow 1\right] = \Pr\left[\mathsf{i\text{-}SO\text{-}CCA}^{\mathcal{S}}(n) \Rightarrow 1\right] .$$

Proof of Claim 2.4.9. We describe $\mathcal{S}$ run in the i-SO-CCA experiment assuming that $\mathcal{A}_{so}$ does not cause abort to happen.

$\mathcal{S}_1(n)$ runs PKE.Gen on its own to obtain (pk, sk). $\mathcal{S}_1$ invokes $\mathcal{A}_{so,1}(pk, n)$. Hash and decryption queries by $\mathcal{A}_{so}$ are answered as in experiment Exp_4. Once $\mathcal{A}_{so,1}$ halts with $\mathfrak{D}$, simulator $\mathcal{S}_1$ halts with output $\mathfrak{D}$ as well.

Once $\mathcal{S}_2$ is run, it computes ciphertexts as in experiment Exp_4 and runs $\mathcal{A}_{so,2}(\mathbf{c})$. If $\mathcal{A}_{so,2}$ queries $\textsc{Open}(i)$, simulator $\mathcal{S}_2$ relays the query to its ideal experiment to obtain m_i. Then $\mathcal{S}_2$ programs the hash functions as in experiment Exp_4 and forwards (r_i, m_i) to $\mathcal{A}_{so,2}$. When $\mathcal{A}_{so,2}$ halts with output *out*, $\mathcal{S}_2$ outputs *out* and terminates. ■

The claim from Theorem 2.4.4 follows from collecting the results of Claims 2.4.5 to 2.4.9. ■

Chapter 3

Selective Opening Security of Hybrid Encryption

We already established results on widely standardized PKE schemes in Chapter 2. However, they remain of little use to obtain *practical* PKE resisting selective opening attacks: Recall that in all covered transformations the symmetric encryption consists of one-time-padding the plaintext with the output of a random oracle to ensure efficient openability. In the case of the OAEP transform (Construction 2.3.3) by design, in the cases of Constructions 2.2.5 and 2.4.3 by our choice. This severely limits the results of Chapter 2 to plaintexts that are not longer than the output lengths of the used random oracle, e.g., less than 512 bits when using SHA-3 [Dwo15].

PKE in practice is usually composed of a KEM employed to transport a short (symmetric) key, while a highly efficient data encapsulation mechanism is used to encrypt the plaintext. Thus, one might consider using $\mathsf{DHIES}_{\oplus}$ (Corollary 2.2.14) and the PKE obtain by using the $\mathsf{RSA\text{-}KEM}$ in Construction 2.2.5 as a KEM. Unfortunately, SO security of a KEM, generally, does not carry over to a PKE built following the KEM/DEM-approach [BDWY11].

In this chapter we study the selective opening security of hybrid PKE schemes as employed in practice. Contrary to previous approaches (e.g. [LP15]) we focus on properties of a DEM rather than the KEM that render the whole hybrid PKE scheme selective opening secure. To this end, we introduce the notion of *simulatability* for DEMs built around blockciphers. If a DEM offers simulatabilty and one-time integrity protection, we may combine it with any IND-CCA secure KEM to obtain an SIM-SO-CCA secure PKE in the ideal cipher model (see [CPS08]).

We recall some important cryptographic notions next.

3.1 Preliminaries

We define partial permutations and blockciphers. In our proofs, the former play an important role for the abstraction of the latter.

3.1.1 Symmetric Primitives

Definition 3.1.1 ((partial permutation), blockcipher). For a finite domain $\mathcal{D}$ we denote the set of all permutations on $\mathcal{D}$ with $\mathcal{P}(\mathcal{D})$ and the set of all partial permutations on $\mathcal{D}$ with $\mathcal{PP}(\mathcal{D})$. Precisely, a relation $R \subseteq \mathcal{D} \times \mathcal{D}$ is a *partial permutation* if $\alpha R \beta, \alpha' R \beta \Rightarrow \alpha = \alpha'$ and $\alpha R \beta, \alpha R \beta' \Rightarrow \beta = \beta'$; relation R is a *permutation* if in addition $|R| = |\mathcal{D}|$ holds. A *blockcipher* with key space $\mathcal{K}$ and domain $\mathcal{D}$ is a family $(E_k)_{k \in \mathcal{K}}$ of permutations $E_k \in \mathcal{P}(\mathcal{D})$.

Definition 3.1.2 (ideal cipher). A blockcipher $(E_k)_{k \in \mathcal{K}}$ with key space $\mathcal{K}$ and domain $\mathcal{D}$ obtained by for all $k \in \mathcal{K}$ letting $E_k \leftarrow_{\$} \mathcal{P}(\mathcal{D})$ is called *ideal cipher*.

We associate with a partial permutation $R \in \mathcal{PP}(\mathcal{D})$ the partial functions $R^+ \colon \mathcal{D} \to \mathcal{D}$ and $R^- \colon \mathcal{D} \to \mathcal{D}$ that evaluate R left-to-right and right-to-left, respectively. For instance, if $(\alpha, \beta) \in R$ then $R^+(\alpha) = \beta$ and $R^-(\beta) = \alpha$. We write $\mathrm{Dom}(R)$ and $\mathrm{Rng}(R)$ for the domain and range of R^+, i.e., for the sets $\{\alpha \in \mathcal{D} \mid \exists \beta : (\alpha, \beta) \in R\}$ and $\{\beta \in \mathcal{D} \mid \exists \alpha : (\alpha, \beta) \in R\}$, respectively. If $\alpha \notin \mathrm{Dom}(R)$ and $\beta \notin \mathrm{Rng}(R)$ we denote with $R \leftarrow R \cup \{(\alpha, \beta)\}$ the operation of 'programming' R such that $R^+(\alpha) = \beta$ and $R^-(\beta) = \alpha$ for the updated R, which is again a partial permutation. Note that any partial permutation can be completed to a (full) permutation by adding sufficiently many such pairs (α, β) to it. More importantly, if a partial permutation is selected according to the uniform distribution over some subset of $\mathcal{PP}(\mathcal{D})$, it can be extended to a permutation uniformly distributed in $\mathcal{P}(\mathcal{D})$ by adding random such pairs (α, β) to it.

Definition 3.1.3 (keyed hash function). A *keyed hash function* for a message space $\mathcal{M}$ consists of a key space $\mathcal{K}$, a tag space $\mathcal{T}$, and an efficient function $\mathrm{khf} \colon \mathcal{K} \times \mathcal{M} \to \mathcal{T}$.

We proceed with specifying the syntax and functionality of DEMs. As a corresponding notion of authenticity we define integrity of ciphertexts [BN00]. In a nutshell, a DEM offers this feature if no adversary with access to an encapsulation oracle can find a fresh ciphertext that corresponds to a valid message, i.e., is not rejected by the decapsulation algorithm. Relevant to our work is in particular the corresponding one-time notion where the adversary can pose at most one encapsulation query.

Definition 3.1.4 (data encapsulation mechanism). A *data encapsulation mechanism* (DEM) for a plaintext space $\mathcal{M}$ consists of a finite key space $\mathcal{K}$, a ciphertext space $\mathcal{C}$, and a pair of efficient algorithms $\mathsf{DEM} = (\mathsf{DEM.Enc}, \mathsf{DEM.Dec})$ of the form

$$\mathsf{DEM.Enc}\colon \mathcal{K} \times \mathcal{M} \to \mathcal{C} \qquad \mathsf{DEM.Dec}\colon \mathcal{K} \times \mathcal{C} \to \mathcal{M} \cup \{\bot\} \ ,$$

where symbol '$\bot$' may be used to indicate errors. Correctness requires that for all $k \in \mathcal{K}$ and $m \in \mathcal{M}$, if $\mathsf{DEM.Enc}(k, m) = c$ then $\mathsf{DEM.Dec}(k, c) = m$.

Recall from Section 2.2 that we combined the one-time pad with a OT-secure MAC to obtain integrity protection. As for practical DEMs, the latter might be realized in other ways than employing a MAC, we introduce the notion of (one-time) integrity of ciphertexts [BN00].

Definition 3.1.5 (OT-INT-CTXT secure DEM). A data encapsulation mechanism is (τ, q_d, ε)-OT-INT-CTXT secure if for all τ-time adversaries $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ that interact in the $\mathsf{OT\text{-}INT\text{-}CTXT}$ experiment from Figure 3.1 and issue at most q_d queries to the DEM.DEC oracle we have

$$\Pr\left[\mathsf{OT\text{-}INT\text{-}CTXT}^{\mathcal{A}} \Rightarrow 1\right] \leq \varepsilon \ .$$

Exp $\mathsf{OT\text{-}INT\text{-}CTXT}^{\mathcal{A}}$	**Oracle** DEM.DEC(c)
$c^* \leftarrow_\$ \emptyset$	If $c = c^*$: Abort
$k \leftarrow_\$ \mathcal{K}$	$m \leftarrow \mathsf{DEM.Dec}(k, c)$
$(m, st) \leftarrow_\$ \mathcal{A}_1^{\text{DEM.DEC}}$	If $m \neq \bot$:
$c^* \leftarrow \mathsf{DEM.Enc}(k, m)$	Stop with 1
$() \leftarrow_\$ \mathcal{A}_2^{\text{DEM.DEC}}(st, c^*)$	Return $\bot$
Stop with 0	

Figure 3.1: Experiment for defining OT-INT-CTXT security of DEMs.

3.1.2 Hybrid Encryption

KEMs While we assumed KEMs to sample keys uniformly in Section 2.2, we drop the requirement in the following.

We define IND-CCA security for KEMs next.

Definition 3.1.6 (IND-CCA secure KEM). A key encapsulation mechanism KEM is (τ, q_d, ε)-IND-CCA secure if all τ-time adversaries $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ that interact in the $\mathsf{IND\text{-}CCA}_b$ experiments from Figure 3.2 and issue at most q_d queries to the KEM.DEC oracle we have

$$\left|\Pr\left[\mathsf{IND\text{-}CCA}_0^{\mathcal{A}} \Rightarrow 1\right] - \Pr\left[\mathsf{IND\text{-}CCA}_1^{\mathcal{A}} \Rightarrow 1\right]\right| \leq \varepsilon \ .$$

Exp $\mathsf{IND\text{-}CCA}_b^{\mathcal{A}}$	**Oracle** KEM.DEC(c)
$c^* \leftarrow \emptyset$	If $c = c^*$: Abort
$(pk, sk) \leftarrow_\$ \mathsf{KEM.Gen}$	$k \leftarrow \mathsf{KEM.Dec}_{sk}(c)$
$st \leftarrow_\$ \mathcal{A}_1^{\mathrm{KEM.DEC}}(pk)$	Return k
$(k_0^*, c^*) \leftarrow_\$ \mathsf{KEM.Enc}_{pk}$	
$k_1^* \leftarrow_\$ \mathcal{K}$	
$b' \leftarrow_\$ \mathcal{A}_2^{\mathrm{KEM.DEC}}(st, c^*, k_b^*)$	
Stop with b'	

Figure 3.2: Security experiments $\mathsf{IND\text{-}CCA}_b$ for defining IND-CCA security of KEMs.

In most applications a DEM is combined with a KEM (see Definition 2.2.1) to obtain (hybrid) PKE [CS03] as follows:

Construction 3.1.7 (hybrid encryption). *Take a DEM* (DEM.Enc, DEM.Dec) *for a plaintext space* $\mathcal{M}$ *and a KEM* (KEM.Gen, KEM.Enc, KEM.Dec) *for the key space of the DEM. Let the randomness space of* PKE.Enc *be defined as the randomness space of* KEM.Enc*. Then the algorithms in Figure 3.3 form the hybrid PKE scheme.*

Proc PKE.Gen	**Proc** $\mathsf{PKE.Enc}_{pk}(m; r)$	**Proc** $\mathsf{PKE.Dec}_{sk}(\langle c^{(1)}, c^{(2)}\rangle)$
$(pk, sk) \leftarrow_\$ \mathsf{KEM.Gen}$	$(k, c^{(1)}) \leftarrow \mathsf{KEM.Enc}_{pk}(r)$	$k \leftarrow \mathsf{KEM.Dec}_{sk}(c^{(1)})$
Return (pk, sk)	$c^{(2)} \leftarrow \mathsf{DEM.Enc}(k, m)$	If $k = \perp$: Return $\perp$
	Return $\langle c^{(1)}, c^{(2)}\rangle$	$m \leftarrow \mathsf{DEM.Dec}(k, c^{(2)})$
		Return m

Figure 3.3: Hybrid construction of PKE from a KEM and a DEM.

3.2 Simulatable DEMs and our Main Result

In this section we present our main result on hybrid public key encryption. We define a combinatorial property of a DEM called *simulatability*. Then we show that any KEM and any DEM satisfying standard security notions yield a SIM-SO-CCA secure hybrid PKE (in the ideal cipher model) if the DEM is simulatable. [CPS08, EM93, KR01].

3.2.1 Simulatable DEMs

Many practical DEMs are constructed from blockciphers, possibly in combination with further symmetric building blocks like universal hash functions or MACs. We formalize next what it means for a DEM to make use of a blockcipher in a black-box way. Virtually all blockcipher-based DEMs, and in particular those specified by the major standardization bodies, are of this type. In our definition, $\mathcal{K}$ denotes the key space of the blockcipher and $\mathcal{K}'$ denotes the cartesian product of the key spaces of the remaining cryptographic primitives used by the scheme. For instance, in an encrypt-then-MAC construction, $\mathcal{K}'$ would be the key space of the message authentication code; if the construction requires no further keyed primitive, $\mathcal{K}'$ would be the trivial set containing a single element.

Recall from Definition 3.1.1 that $\mathcal{P}(\mathcal{D})$ and $\mathcal{PP}(\mathcal{D})$ denote the sets of all permutations and partial permutations, respectively, on domain $\mathcal{D}$.

The next two definitions provide the syntactical requirements we impose on DEMs. The first definitions establishes how a DEM may be built around a blockcipher. The second definition specifies how a DEM shall employ its key material.

Definition 3.2.1 (oracle DEM). An *oracle data encapsulation mechanism* (oDEM) for a domain $\mathcal{D}$ and a plaintext space $\mathcal{M}$ consists of a finite key space $\mathcal{K}'$, a ciphertext space $\mathcal{C}$, and efficient algorithms $\mathsf{O.Enc}$ and $\mathsf{O.Dec}$ that have oracle access to a permutation π on $\mathcal{D}$ (in both directions) and are of the form

$$\mathsf{O.Enc}^{\pi} \colon \mathcal{K}' \times \mathcal{M} \to \mathcal{C} \qquad \mathsf{O.Dec}^{\pi} \colon \mathcal{K}' \times \mathcal{C} \to \mathcal{M} \mathbin{\dot\cup} \{\perp\} \ ,$$

where symbol '$\perp$' may be used to indicate errors. Correctness requires that for all $\pi \in \mathcal{P}(\mathcal{D})$, $k' \in \mathcal{K}'$, and $m \in \mathcal{M}$, if $\mathsf{O.Enc}^{\pi}(k', m) = c$ then $\mathsf{O.Dec}^{\pi}(k', c) = m$.

Definition 3.2.2 (permutation-driven DEM). A DEM for plaintext space $\mathcal{M}$ with keyspace $\mathcal{K}'' = \mathcal{K} \times \mathcal{K}'$ is $(\mathcal{K}, \mathcal{D})$*-permutation-driven* if there exists an oracle DEM for $\mathcal{D}$ and $\mathcal{M}$ with algorithms $\mathsf{O.Enc}^{\pi} \colon \mathcal{K}' \times \mathcal{M} \to \mathcal{C}$ and $\mathsf{O.Dec}^{\pi} \colon \mathcal{K}' \times \mathcal{C} \to \mathcal{M} \mathbin{\dot\cup} \{\perp\}$ and a blockcipher $(E_k)_{k \in \mathcal{K}}$ on domain $\mathcal{D}$ such that for all $k' \in \mathcal{K}'$ and $m \in \mathcal{M}$ and $c \in \mathcal{C}$ we have

$$\mathsf{DEM.Enc}((k, k'), m) = \mathsf{O.Enc}^{E_k}(k', m) \quad \text{and} \quad \mathsf{DEM.Dec}((k, k'), c) = \mathsf{O.Dec}^{E_k}(k', c) \ . \tag{3.1}$$

According to this definition, for any specific permutation-driven DEM many corresponding oracle DEMs, i.e., $\mathsf{O.Enc}$ and $\mathsf{O.Dec}$ algorithms, and blockciphers E might exist. In practice, however, a single canonical specification of these algorithms will stick out. In particular, this holds, as we will see, for the standardized DEMs studied in Section 3.3. For the sake of a concise notation, we thus assume that suitable $\mathsf{O.Enc}$, $\mathsf{O.Dec}$, and E algorithms are always uniquely given.

We next define a combinatorial property called *simulatability* that holds for an oracle DEM if, in principle, the encapsulation algorithm could commit to a ciphertext before seeing the corresponding plaintext; intuitively, this is only possible if the permutation in the oracle is 'flexible enough', i.e., can be 'programmed'. We formalize this idea by splitting the encapsulation routine into two components, Fake and Make.

First Fake outputs a ciphertext c without seeing the plaintext m (but it length $|m|$), then Make, on input m, is meant to find a possible (partial) permutation instance $\tilde{\pi}$ under which indeed m would be encapsulated to c. To be useful in our later selective opening related proofs where we want to embed $\tilde{\pi}$ into an ideal cipher, $\tilde{\pi}$ is further required to be uniformly distributed (conditioned on the formulated requirements).

Definition 3.2.3 (simulatable oracle DEM). Consider an oracle DEM for a domain $\mathcal{D}$ and a plaintext space $\mathcal{M}$ that has an encapsulation algorithm of the form $\mathsf{O.Enc}^{\Pi} \colon \mathcal{K}' \times \mathcal{M} \to \mathcal{C}$. Consider algorithms Fake and Make of the form

$$\mathsf{Fake} \colon \mathcal{K}' \times \mathbb{N} \to_{\$} \mathcal{C} \times \Sigma \quad \text{and} \quad \mathsf{Make} \colon \Sigma \times \mathcal{M} \to_{\$} \mathcal{PP}(\mathcal{D}) \ ,$$

where Σ is a state space shared between the two algorithms. We say that the oracle DEM is *ε-simulatable* (by Fake and Make) if for all $k' \in \mathcal{K}'$ and $m \in \mathcal{M}$, for the random variable (defined over the coins of Fake and Make)

$$\Pi_{k'}^{m} = \{\tilde{\pi} : (c, st) \leftarrow_{\$} \mathsf{Fake}(k', |m|); \tilde{\pi} \leftarrow_{\$} \mathsf{Make}(st, m)\}$$

we have

(1) the partial permutation $\Pi_{k'}^{m}$ can be extended to a uniformly distributed permutation on $\mathcal{D}$, i.e., by 'filling up' $\Pi_{k'}^{m}$ with random pairs one obtains a permutation uniformly distributed in $\mathcal{P}(\mathcal{D})$;

(2) the ciphertext output by Fake deviates from the one that would be output by $\mathsf{O.Enc}$ if invoked with an extension of the partial permutation output by Make with probability at most ε. More precisely, for any uniformly distributed extension $\pi \in \mathcal{P}(\mathcal{D})$ of $\Pi_{k'}^{m}$ we have $\Pr[c \neq \mathsf{O.Enc}^{\pi}(k', m)] \leq \varepsilon$ (where the probability is also taken over the random extension of $\Pi_{k'}^{m}$ to π);

(3) the joint running time of $\mathsf{Fake}(k', |m|)$ and $\mathsf{Make}(st, m)$ does not exceed the running time of $\mathsf{O.Enc}(k', m)$, not counting the latter's oracle queries.

In informal discussions, when we say that a data encapsulation mechanism is *simulatable* we mean that it is permutation-driven and $\mathsf{Fake}, \mathsf{Make}$ algorithms exist for which the corresponding oracle DEM ε-simulatable with a small value ε.

Remark 3.2.4 We note that simulatability is a purely information-theoretic property of an oracle DEM.

Concerning the above definition it is important to understand that the random coins of Fake and Make, and the coins used to extend the partial permutation in items (1) and (2), belong to the same probability space.

In line with a comment made above, for all practical DEMs that are simulatable, corresponding specifications for the Fake and Make algorithms emerge canonically. For the sake of notational clarity, from now on we thus assume uniqueness.

Proving Simulatability. We discuss a general technique for proving the simulatability of an oracle DEM. The Fake and Make algorithms are typically explicitly provided in the proof. Fake's strategy is to mimic the behavior of O.Enc by executing it and answering blockcipher queries with random elements from $\mathcal{D}$. Make constructs a partial permutation $\tilde{\pi}$ that fits this random assignment by starting with the empty relation $\tilde{\pi} = \emptyset$ and iteratively adding pairs $(\alpha, \beta) \in \mathcal{D} \times \mathcal{D}$ to $\tilde{\pi}$ that help meeting the $\mathsf{O.Enc}^{\tilde{\pi}}(k', m) = c$ goal, always taking care that also the $\alpha\tilde{\pi}\beta, \alpha'\tilde{\pi}\beta \Rightarrow \alpha = \alpha'$ and $\alpha\tilde{\pi}\beta, \alpha\tilde{\pi}\beta' \Rightarrow \beta = \beta'$ requirements from Definition 3.1.1 are not violated (Make aborts if simultaneously reaching these conditions turns out to be impossible). Simulatability requirement (1) is achieved by ensuring that for each addition of (α, β) to $\tilde{\pi}$ either α or β are uniformly distributed, conditioned on the prior state of $\tilde{\pi}$. Proving the bound from condition (2) typically requires a combinatorial argument that assesses the probability of collisions. Requirement (3) follows by inspection of the specifications of Fake and Make.

3.2.2 Selective Opening Security from Simulatable DEMs

Our main result is on the SO security of public-key encryption obtained by combining an arbitrary KEM with a permutation-driven DEM. Our analysis is conducted in the ideal cipher model for the blockcipher underlying the DEM. We give an informal version of our main theorem and an outline of the proof. We caution that some technical preconditions are omitted in the statement as we give it here. See Section 3.4 for the full theorem statement and proof.

Exp $\mathsf{r\text{-}SO\text{-}CCA}^{\mathcal{A}}(n)$

For all $k \in \mathcal{K}$: $E_k \leftarrow \emptyset$
$\mathcal{I} \leftarrow \emptyset$; $\mathbf{c} \leftarrow \emptyset$
$(pk, sk) \leftarrow_\$ \mathsf{KEM.Gen}$
$(\mathfrak{D}, st) \leftarrow_\$ \mathcal{A}_1^{\text{PKE.DEC},\mathrm{E}}(pk, n)$
$\mathbf{m} \leftarrow_\$ \mathfrak{D}$
For $i \leftarrow 1$ to n:
  $r_i \leftarrow_\$ \mathcal{R}$
  $(k_i'', c_i^{(1)}) \leftarrow \mathsf{KEM.Enc}_{pk}(r_i)$
  $(k_i, k_i') \leftarrow k_i''$
  $c_i^{(2)} \leftarrow \mathsf{O.Enc}^{\mathrm{E}(k_i;\cdot)}(k_i', m_i)$
  $c_i \leftarrow \langle c_i^{(1)}, c_i^{(2)} \rangle$
$\mathbf{c} \leftarrow (c_1, \ldots, c_n)$
$out \leftarrow_\$ \mathcal{A}_2^{\text{OPEN},\text{PKE.DEC},\mathrm{E}}(st, \mathbf{c})$
Stop with $\mathsf{Pred}(\mathfrak{D}, \mathbf{m}, \mathcal{I}, out)$

Oracle OPEN(i)

$\mathcal{I} \leftarrow \mathcal{I} \cup \{i\}$
Return (m_i, r_i)

Oracle PKE.DEC$(\langle c^{(1)}, c^{(2)} \rangle)$

If $\langle c^{(1)}, c^{(2)} \rangle \in \mathbf{c}$: Abort
$k'' \leftarrow \mathsf{KEM.Dec}_{sk}(c^{(1)})$
If $k'' = \perp$: Return $\perp$
$(k, k') \leftarrow k''$
$m \leftarrow \mathsf{O.Dec}^{\mathrm{E}(k;\cdot)}(k', c^{(2)})$
Return m

Oracle $\mathrm{E}^+(k, \alpha)$

If $\alpha \notin \mathrm{Dom}(E_k)$:
  $\beta \leftarrow_\$ \mathcal{D} \setminus \mathrm{Rng}(E_k)$
  $E_k \leftarrow E_k \cup \{(\alpha, \beta)\}$
Return $E_k^+(\alpha)$

Oracle $\mathrm{E}^-(k, \beta)$

If $\beta \notin \mathrm{Rng}(E_k)$:
  $\alpha \leftarrow_\$ \mathcal{D} \setminus \mathrm{Dom}(E_k)$
  $E_k \leftarrow E_k \cup \{(\alpha, \beta)\}$
Return $E_k^-(\beta)$

Figure 3.4: Experiment $\mathsf{r\text{-}SO\text{-}CCA}$ adapted towards the analysis of a PKE scheme constructed following the KEM/DEM paradigm using a permutation-driven DEM with corresponding oracle DEM algorithms $\mathsf{O.Enc}$ and $\mathsf{O.Dec}$, in the ideal cipher model. We further abbreviate the pair $\mathrm{E}^+, \mathrm{E}^-$ of ideal cipher oracles with just E.

Theorem 3.2.5 (informal). *Combine any KEM and any permutation-driven DEM to obtain a PKE scheme. If the KEM is IND-CCA secure, the DEM is OT-INT-CTXT secure and the corresponding oracle DEM is simulatable, then the combined PKE scheme is SIM-SO-CCA secure, in the ideal cipher model.*

PROOF SKETCH In Figure 3.4 we reproduce the $\mathsf{r\text{-}SO\text{-}CCA}$ experiment from Figure 2.1 with the hybrid construction of the encryption scheme, the oracle DEM underlying the DEM, and the ideal cipher model made explicit. (In the $\mathsf{i\text{-}SO\text{-}CCA}$ experiment there is nothing to be adapted.) We correspondingly equip adversary $\mathcal{A}$ and the DEM algorithms with oracles E^+ and E^- that implement an ideal blockcipher on domain $\mathcal{D}$. In particular, for each key k, oracles $\mathrm{E}^+(k;\cdot)$ and $\mathrm{E}^-(k;\cdot)$ are inverses of each other. For a concise notation, we typically just write E for the pair consisting of E^+ and E^-. We implement ideal cipher E via lazy sampling and keep track of made assignments using an experiment internal family $(E_k)_{k\in\mathcal{K}}$ of partial permutations $E_k \in \mathcal{PP}(\mathcal{D})$. Note that we do not provide the KEM algorithms with access to E, meaning we assume the KEM does not use the same blockcipher as the DEM. See Section 3.4 for a discussion.

When it comes to constructing $\mathcal{S}$ from $\mathcal{A}$, the strategy is to let the former run the

Simulator $\mathcal{S}_1(n)$

01 For all $k \in \mathcal{K}$: $E_k \leftarrow \emptyset$

02 $\mathbf{c} \leftarrow \emptyset$

03 $(pk, sk) \leftarrow_\$ \mathsf{KEM.Gen}$

04 $\mathfrak{D} \leftarrow_\$ \mathcal{A}_1^{\mathrm{E},\mathrm{PKE.D{\scriptsize EC}}}(pk, n)$

05 Return $\mathfrak{D}$

Simulator $\mathcal{S}_2^{\mathrm{O{\scriptsize PEN}}_\mathcal{S}}(|m_1|, \ldots, |m_n|)$

06 For $i \leftarrow 1$ to n:

07 $\quad r_i \leftarrow_\$ \mathcal{R}$

08 $\quad (k_i'', c_i^{(1)}) \leftarrow \mathsf{KEM.Enc}_{pk}(r_i)$

09 $\quad (k_i, k_i') \leftarrow k_i''$

10 $\quad (c_i^{(2)}, st_i) \leftarrow_\$ \mathsf{Fake}(k_i', |m_i|)$

11 $\quad c_i \leftarrow \langle c_i^{(1)}, c_i^{(2)} \rangle$

12 $\mathbf{c} \leftarrow (c_1, \ldots, c_n)$

13 $out \leftarrow_\$ \mathcal{A}_2^{\mathrm{E},\mathrm{O{\scriptsize PEN}}_\mathcal{A},\mathrm{PKE.D{\scriptsize EC}}}(\mathbf{c})$

14 Return out

Oracle Open(i)

15 $m_i \leftarrow \mathrm{O{\scriptsize PEN}}_\mathcal{S}(i)$

16 $\tilde{\pi} \leftarrow_\$ \mathsf{Make}(st_i, m_i)$

17 $E_{k_i} \leftarrow E_{k_i} \cup \tilde{\pi}$

18 Return (m_i, r_i)

Oracle PKE.Dec$(\langle c^{(1)}, c^{(2)} \rangle)$
as in Figure 3.4

Oracle $\mathrm{E}^+(k, \alpha)$
as in Figure 3.4

Oracle $\mathrm{E}^-(k, \beta)$
as in Figure 3.4

Figure 3.5: Simplified version of simulator $\mathcal{S} = (\mathcal{S}_1, \mathcal{S}_2)$, constructed from adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$. We write $\mathrm{O{\scriptsize PEN}}_\mathcal{S}$ for the opening oracle provided to $\mathcal{S}_2$. For simplicity we do not annotate the state information passed from $\mathcal{A}_1$ to $\mathcal{A}_2$ and from $\mathcal{S}_1$ to $\mathcal{S}_2$.

latter as a subroutine: Simulator $\mathcal{S}$ converts its own input to an input for $\mathcal{A}$, uses the output of $\mathcal{A}$ as the own output, and answers, and in some cases relays, oracle queries posed by $\mathcal{A}$. We give the footprint of a universal such simulator that leverages on the simulatability of the (permutation-driven) DEM in Figure 3.5. For the sake of clarity, we simplified the specifications of algorithms $\mathcal{S}_1$ and $\mathcal{S}_2$ quite a bit, removing many technicalities. While we briefly discuss the missing parts below, for the full details of the simulator and a formal analysis we refer to Section 3.4.

$\mathcal{S}$ will only benefit from internally running $\mathcal{A}$, if $\mathcal{A}$ is in the r-SO-CCA experiment. As already mentioned in 'Proving SIM-SO-CCA Security' on page 69, $\mathcal{S}$ has to:

(a) generate and provide a public key for $\mathcal{A}_1$, (b) prove ciphertexts to $\mathcal{A}_2$ that correspond to plaintext $m_1, \ldots, m_n$, (c) prove adequate randomness when processing opening queries of $\mathcal{A}_2$, and (d) handle decryption queries of $\mathcal{A}_1$ and $\mathcal{A}_2$.

Further, ideal cipher queries of $\mathcal{A}_1$ and $\mathcal{A}_2$ have to be taken care of. The latter is straight-forward when deploying lazy sampling, i.e., using the mechanisms of the r-SO-CCA version from Figure 3.4. Also (a) and (d) are easy to deal with: The public key pk provided to $\mathcal{A}_1$ is a regular KEM key generated by $\mathcal{S}_1$ (lines 03, 04); in particular, secret key sk is known to $\mathcal{S}$ and can be used to process decryption queries. Concerning (b), creating ciphertexts $c_1, \ldots, c_n$ for $\mathcal{A}_2$ consists, in principle, of two parts: letting the KEM establish session keys and encapsulating plaintext with the DEM. Component $\mathcal{S}_2$ of our simulator does the former according to the specification, i.e., by invoking algorithm KEM.Enc with fresh randomness (lines 07, 08), while for the latter, as it cannot invoke DEM.Enc (or, more precisely, O.Enc) for not knowing the plaintexts it needs to encapsulate, it leverages on the simulatability of the DEM and obtains the corresponding ciphertext from an execution of the Fake algorithm (line 10). How $\mathcal{S}_2$ deals with (c) is now immediate: for each created ciphertext it knows the randomness used, so it can release it in an opening query (line 18). Note, however, that knowledge of this randomness brings $\mathcal{A}_2$ into the position to verify the DEM ciphertext components generated by Fake (e.g., by decapsulating or re-encapsulating them); correspondingly, the OPEN oracle in addition runs the Make algorithm and embeds the partial permutation proposed by it into ideal cipher E (lines 16, 17). By the definition of simulatability of a DEM, this fixes the ideal cipher such that overall consistency is established.

As announced earlier, in Figure 3.5 we leave out some details of our simulator. These are related to situations in which $\mathcal{S}$ cannot uphold a proper environment for $\mathcal{A}$ and has to abort its execution. This is the case when Fake and Make fail to properly simulate O.Enc (the definition of simulatability considers a small probability of failure), or if the partial permutation output by Make cannot be embedded into the ideal cipher (line 17). The latter condition can result from various actions of adversary $\mathcal{A}$, for instance (explicitly) from queries to the E oracles, or (implicitly) from evaluations of E during the processing of a decryption query. In the full proof given in Section 3.4 we show that if the KEM is IND-CCA secure and the DEM is OT-INT-CTXT secure, then the probability is small that any of these conditions is met. (Very briefly speaking, we use the KEM notion for bounding the probability of explicit queries, and we use the DEM notion for bounding the probability of implicit ones.)

Classifying the Result We briefly describe how our result relates to standard results on the IND-CCA security of hybrid PKE. To obtain IND-CCA secure hybrid encryption we require[1] an IND-CCA secure KEM to be combined with an IND-OT-CCA secure KEM. Thereby, IND-OT-CCA security of a DEM follows from its IND-OT-CPA and OT-INT-CTXT security [BN00]. Observe that in Theorem 3.2.5 we do require the KEM to be IND-CCA and the DEM to be OT-INT-CTXT secure while we assume the (corresponding oracle) DEM to be simulatable instead of IND-OT-CPA secure. One easily verifies that simulatability implies IND-OT-CPA security in the ideal cipher model.[2] That is, simulatability is the key property of the DEM lifting the security of the hybrid PKE from IND-CCA to SIM-SO-CCA security (in the ideal cipher model).

[1]See [HK07] for an exception.
[2]In a nutshell, it allows the IND-OT-CPA experiment to compute an attacker's challenge plaintext-independently.

See Section 2.2.5 for a discussion on why employing an IND-CCA (rather than an IND-SO-CCA) secure KEM is sufficient for our results.

3.3 Simulatability of practical DEMs

We prove that all blockcipher-based DEMs that were standardized by the National Institute of Standards and Technology (NIST) are permutation-driven and simulatable. Concretely we analyze the CTR and CBC modes of operation (SP 800-38A [Dwo01]), a CBC variant with ciphertext stealing (CTS) (Addendum to SP 800-38A [Dwo10]), the CCM mode (SP 800-38C [Dwo07a]), and the GCM mode (SP 800-38D [Dwo07b]). More precisely, as for our results on selective opening security only those DEMs are relevant that offer ciphertext integrity (see Definition 3.1.5), instead of plain CTR, CBC, and CBC/CTS encryption we actually analyze their encrypt-then-MAC variants, where we assume arbitrary strongly unforgeable MACs. Further, as CCM and GCM are authenticated encryption schemes with associated data (AEAD [Rog02]), we turn them into DEMs by using them with a fixed nonce N_0 and an empty associated data string A_0. As the four named modes follow different design principles, some of which might be incompatible with simulatability, analyzing all of them is more than just a matter of due diligence. For instance, GCM is an encrypt-then-MAC and CCM is a MAC-then-encrypt design. Further, while CTR mode encrypts by xoring blockcipher outputs with the plaintext, CBC mode encrypts by pushing plaintexts blocks through the cipher, and CCM combines both approaches.

In the following we specify the mentioned DEMs in their oracle DEM form, assuming that the underlying blockcipher $(E_k)_{k \in \mathcal{K}}$ is over domain $\mathcal{D} = \{0,1\}^\ell$. We show their simulatability by proposing and analyzing corresponding Fake and Make algorithms, following the general strategy suggested at the end of Section 3.2.1.

Notation For $n \in \mathbb{N}$, we let $[1 \mathinner{..} n] := \{1, \ldots, n\}$. For a bitstring x of length at least ℓ we write $\mathrm{msb}_\ell(x)$ for its left-most ℓ bits and $\mathrm{lsb}_\ell(x)$ for its right-most ℓ bits ('most/least significant bits').

3.3.1 CTR-then-MAC

We analyze the DEM obtained by first encrypting the provided plaintext with the CTR0 mode of operation of a blockcipher (counter mode with fixed initial counter value) and then appending a deterministic MAC tag to the ciphertext.

We specify the O.Enc and O.Dec algorithms of CTR0-DEM in Figure 3.6, where we assume that $G\colon [1\,..\,V] \to \mathcal{D}$ denotes a fixed injective function (a 'counter generator') for some sufficiently large value V. The MAC is represented by a keyed hash function $\mathrm{khf}\colon \mathcal{K}' \times \{0,1\}^* \to \{0,1\}^T$. The plaintext space of CTR0-DEM is $\mathcal{M} = \{0,1\}^*$ and the ciphertext space is $\mathcal{C} = \{0,1\}^{\geq T}$.

$\mathsf{O.Enc}^{\pi}(k', m)$	$\mathsf{O.Dec}^{\pi}(k', c)$
01 Write $\lvert m\rvert$ as $(l-1)\ell + l^*$	13 If $\lvert c\rvert < T$: Return $\perp$
02 Split m into $m_1 \dots m_{l-1} m_l^*$	14 Split c into $\bar{c}t$
03 $m_l \leftarrow m_l^* \,\Vert\, 0^{\ell - l^*}$	15 If $t \neq \mathrm{khf}(k', \bar{c})$:
04 For $i \leftarrow 1$ to l:	16 Return $\perp$
05 $u_i \leftarrow G(i)$	17 Write $\lvert\bar{c}\rvert$ as $(l-1)\ell + l^*$
06 $v_i \leftarrow \pi(u_i)$	18 Split $\bar{c}$ into $c_1 \dots c_{l-1} c_l^*$
07 $c_i \leftarrow m_i \oplus v_i$	19 $c_l \leftarrow c_l^* \,\Vert\, 0^{\ell - l^*}$
08 $c_l^* \leftarrow \mathrm{msb}_{l^*}(c_l)$	20 For $i \leftarrow 1$ to l:
09 $\bar{c} \leftarrow c_1 \dots c_{l-1} c_l^*$	21 $u_i \leftarrow G(i)$
10 $t \leftarrow \mathrm{khf}(k', \bar{c})$	22 $v_i \leftarrow \pi(u_i)$
11 $c \leftarrow \bar{c}t$	23 $m_i \leftarrow c_i \oplus v_i$
12 Return c	24 $m_l^* \leftarrow \mathrm{msb}_{l^*}(m_l)$
	25 $m \leftarrow m_1 \dots m_{l-1} m_l^*$
	26 Return m

Figure 3.6: CTR0-DEM. Lines 01 and 17 uniquely identify quantities l and l^* such that $l \in \mathbb{N}^{\geq 1}$ and $0 \leq l^* < \ell$, and $\lvert m\rvert = (l-1)\ell + l^*$ and $\lvert\bar{c}\rvert = (l-1)\ell + l^*$, respectively. Correspondingly, line 02 assumes $\lvert m_1\rvert = \ldots = \lvert m_{l-1}\rvert = \ell$ and $\lvert m_l^*\rvert = l^*$, and line 18 assumes $\lvert c_1\rvert = \ldots = \lvert c_{l-1}\rvert = \ell$ and $\lvert c_l^*\rvert = l^*$. Further, line 14 assumes $\lvert t\rvert = T$.

Lemma 3.3.1 *CTR0-DEM is ε-simulatable with $\varepsilon = (\lceil L/\ell\rceil^2 - \lceil L/\ell\rceil)/2^{\ell+1}$, where L is the maximum plaintext length (in bits).*

Proof. Consider algorithms Fake and Make from Figure 3.7. The idea of Fake is to compute intermediate ciphertext $\bar{c}$ on basis of uniformly distributed blockcipher outputs (see how line 02 in Fake replaces l-many iterations of line 07 in O.Enc), but to compute the MAC tag on $\bar{c}$ faithfully. Note that the correct length of $\bar{c}$ is known to Fake as it coincides with the length of m. Inspection shows that, given m, algorithm Make finds a minimal partial permutation $\tilde{\pi}$ such that Fake and Make jointly mimic the behavior of O.Enc (see here how lines 16–19 of Make arrange the entries of $\tilde{\pi}$ such that they are consistent with lines 06–07 of O.Enc). In some invocations of the algorithms, the described process might fail (lines 17, 18), namely when partial permutation $\tilde{\pi}$ would become inconsistent (i.e., the updated $\tilde{\pi}$ would stop being an element of $\mathcal{PP}$). In such cases Make aborts, outputting the empty partial permutation $\tilde{\pi} = \emptyset$.

We next show that the conditions from Definition 3.2.3 are met. Observe that, as Fake picks values $c_1, \ldots, c_l$ uniformly and independently of each other, the same holds for the values $v_1, \ldots, v_l$ computed in line 16. That is, in each iteration of line 19 a value v_i is added to $\mathrm{Rng}(\tilde{\pi})$ that is uniform conditioned on the (then) current state of $\mathrm{Rng}(\tilde{\pi})$. Thus condition (1) holds. To establish the correctness bound of condition (2) we analyze the probability that Make aborts. By the injectivity of function G the u_i-values from line 15 are pairwise distinct, so the abort condition of line 17 is never met. Further, as values v_i computed in line 16 are uniformly distributed and independent of each other, the abort condition of line 18 is met with probability $\varepsilon = (0 + \ldots + (l-1))/|\mathcal{D}| = ((l^2 - l)/2)/|\mathcal{D}|$ (accumulated over all iterations of the loop). Plugging in the maximum value $l = \lceil L/\ell \rceil$ gives the bound claimed in the statement. Condition (3) is clear. ∎

$\mathsf{Fake}(k', \lvert m \rvert)$	$\mathsf{Make}(st, m)$
01 Write $\lvert m \rvert$ as $(l-1)\ell + l^*$	09 $\tilde{\pi} \leftarrow \emptyset$
02 $c_1, \ldots, c_l \leftarrow_\$ \mathcal{D}$	10 Write $\lvert m \rvert$ as $(l-1)\ell + l^*$
03 $c_l^* \leftarrow \mathrm{msb}_{l^*}(c_l)$	11 Parse st as $(c_1, \ldots, c_l)$
04 $\bar{c} \leftarrow c_1 \ldots c_{l-1} c_l^*$	12 Split m into $m_1 \ldots m_{l-1} m_l^*$
05 $t \leftarrow \mathrm{khf}(k', \bar{c})$	13 $m_l \leftarrow m_l^* \,\Vert\, 0^{\ell - l^*}$
06 $c \leftarrow \bar{c}t$	14 For $i \leftarrow 1$ to l:
07 $st \leftarrow (c_1, \ldots, c_l)$	15 $u_i \leftarrow G(i)$
08 Return c, st	16 $v_i \leftarrow m_i \oplus c_i$
	17 If $u_i \in \mathrm{Dom}(\tilde{\pi})$: Abort
	18 If $v_i \in \mathrm{Rng}(\tilde{\pi})$: Abort
	19 $\tilde{\pi} \leftarrow \tilde{\pi} \cup \{(u_i, v_i)\}$
	20 Return $\tilde{\pi}$

Figure 3.7: Fake and Make for CTR0-DEM. We write 'Abort' as an abbreviation for 'Return $\emptyset$'.

3.3.2 CBC-then-MAC

We consider the DEM obtained by encrypting the plaintext with CBC0 mode (cipher block chaining with initialization vector zero) and appending a MAC tag to the ciphertext. As a variant we also look at CBC0-CTS (CBC0 with 'ciphertext stealing') that supports a complementary plaintext space.

We specify the O.Enc and O.Dec algorithms of CBC-DEM in Figure 3.8 and of CBC-CTS-DEM in Figure 3.9. Similarly as for CTR0-DEM, the MAC is represented by a keyed hash function of the form $\mathrm{khf}\colon \mathcal{K}' \times \{0,1\}^* \to \{0,1\}^T$. The plaintext space of CBC-DEM consists of all plaintexts that have a length that is a multiple of the blocklength ℓ, i.e., $\mathcal{M} = \bigcup_{\lambda \geq \ell, \ell | \lambda} \{0,1\}^\lambda$; the ciphertext space is $\mathcal{C} = \bigcup_{\lambda \geq \ell, \ell | \lambda} \{0,1\}^{\lambda+T}$. In contrast, CBC-CTS-DEM supports all plaintexts lengths that are not a multiple of ℓ, with a minimum value of $\ell+1$; formally, $\mathcal{M} = \bigcup_{\lambda \geq \ell, \ell \nmid \lambda} \{0,1\}^\lambda$ and $\mathcal{C} = \bigcup_{\lambda \geq \ell, \ell \nmid \lambda} \{0,1\}^{\lambda+T}$. Together, CBC-DEM and CBC-CTS-DEM can handle plaintexts of any length not smaller than ℓ.[3]

$\mathsf{O.Enc}^\pi(k', m)$	$\mathsf{O.Dec}^\pi(k', c)$
01 Write $\lvert m \rvert$ as $l\ell$	11 If $\lvert c \rvert < T$: Return $\bot$
02 Split m into $m_1 \dots m_l$	12 Split c into $\bar{c}t$
03 $c_0 \leftarrow 0^\ell$	13 If $t \neq \mathrm{khf}(k', \bar{c})$:
04 For $i \leftarrow 1$ to l:	14 Return $\bot$
05 $u_i \leftarrow m_i \oplus c_{i-1}$	15 Write $\lvert \bar{c} \rvert$ as $l\ell$
06 $c_i \leftarrow \pi(u_i)$	16 Split $\bar{c}$ into $c_1 \dots c_l$
07 $\bar{c} \leftarrow c_1 \dots c_l$	17 $c_0 \leftarrow 0^\ell$
08 $t \leftarrow \mathrm{khf}(k', \bar{c})$	18 For $i \leftarrow 1$ to l:
09 $c \leftarrow \bar{c}t$	19 $u_i \leftarrow \pi^{-1}(c_i)$
10 Return c	20 $m_i \leftarrow u_i \oplus c_{i-1}$
	21 $m \leftarrow m_1 \dots m_l$
	22 Return m

Figure 3.8: CBC-DEM (for multi-block plaintext). Lines 01 and 15 identify quantity $l \in \mathbb{N}^{\geq 0}$ such that $|m| = l\ell$ and $|\bar{c}| = l\ell$, respectively. Correspondingly, line 02 assumes $|m_1| = \ldots = |m_l| = \ell$ and line 16 assumes $|c_1| = \ldots = |c_l| = \ell$. Further, line 12 assumes $|t| = T$.

Lemma 3.3.2 *CBC-DEM is ε-simulatable where $\varepsilon = ((L/\ell)^2 - (L/\ell))/2^\ell$, and CBC-CTS-DEM is ε-simulatable with $\varepsilon = (\lfloor L/\ell \rfloor^2 + \lfloor L/\ell \rfloor)/2^\ell$, where L is the maximum plaintext length (in bits).*

Proof. The proof is similar to the one of Lemma 3.3.1. Consider algorithms Fake and Make from Figure 3.10. The idea of Fake is to compute intermediate ciphertext $\bar{c}$ on basis of uniformly distributed blockcipher outputs (see how line 02 of Fake replaces l-many iterations of line 06 of O.Enc), but to compute the MAC tag on $\bar{c}$ faithfully. Note that the correct length of $\bar{c}$ is known to Fake as it coincides with the length of m. Inspection shows that, given m, algorithm Make finds a minimal partial permutation $\tilde{\pi}$ such that Fake and Make jointly mimic the behaviour of O.Enc (see here how lines 14–17 of Make arrange the entries of $\tilde{\pi}$ such that they are consistent with lines 05, 06 of O.Enc). In some invocations of the algorithms, the described process might fail (lines 15, 16), namely when partial permutation $\tilde{\pi}$ would become inconsistent. In such cases Make aborts, outputting the empty partial permutation $\tilde{\pi} = \emptyset$.

[3] Instead of specifying different algorithms for different classes of plaintext length, one could also join them together into a single, more general algorithm. This is usually done in standards [Dwo10], but we abstain from doing so in this thesis to avoid rather obstructive case distinctions in the analysis.

$\mathsf{O.Enc}^{\pi}(k', m)$
01 Write $|m|$ as $l\ell + l^*$
02 Split m into $m_1 \ldots m_l m^*_{l+1}$
03 $m_{l+1} \leftarrow m^*_{l+1} \,\|\, 0^{\ell - l^*}$
04 $c_0 \leftarrow 0^\ell$
05 For $i \leftarrow 1$ to $l+1$:
06 $\quad u_i \leftarrow m_i \oplus c_{i-1}$
07 $\quad c_i \leftarrow \pi(u_i)$
08 $c^*_l \leftarrow \mathrm{msb}_{l^*}(c_l)$
09 $\bar{c} \leftarrow c_1 \ldots c_{l-1} c^*_l c_{l+1}$
10 $t \leftarrow \mathrm{khf}(k', \bar{c})$
11 $c \leftarrow \bar{c}t$
12 Return c

$\mathsf{O.Dec}^{\pi}(k', c)$
13 If $|c| < T$: Return $\perp$
14 Split c into $\bar{c}t$
15 If $t \neq \mathrm{khf}(k', \bar{c})$:
16 $\quad$ Return $\perp$
17 Write $|\bar{c}|$ as $l\ell + l^*$
18 Split $\bar{c}$ into $c_1 \ldots c_{l-1} c^*_l c_{l+1}$
19 $u_{l+1} \leftarrow \pi^{-1}(c_{l+1})$
20 $m^*_{l+1} \leftarrow \mathrm{msb}_{l^*}(u_{l+1}) \oplus c^*_l$
21 $c_l \leftarrow c^*_l \,\|\, \mathrm{lsb}_{\ell - l^*}(u_{l+1})$
22 $c_0 \leftarrow 0^\ell$
23 For $i \leftarrow 1$ to l:
24 $\quad u_i \leftarrow \pi^{-1}(c_i)$
25 $\quad m_i \leftarrow u_i \oplus c_{i-1}$
26 $m \leftarrow m_1 \ldots m_l m^*_{l+1}$
27 Return m

Figure 3.9: CBC-CTS-DEM (for plaintext that require padding). Lines 01 and 17 uniquely identify quantities l and l^* such that $l \in \mathbb{N}^{\geq 1}$ and $1 \leq l^* < \ell$, and $|m| = l\ell + l^*$ and $|\bar{c}| = l\ell + l^*$, respectively. Correspondingly, line 02 assumes $|m_1| = \ldots = |m_l| = \ell$ and $|m^*_{l+1}| = l^*$, and line 18 assumes $|c_1| = \ldots = |c_{l-1}| = \ell$ and $|c^*_l| = l^*$ and $|c_{l+1}| = \ell$. Further, line 14 assumes $|t| = T$.

We next show that the conditions from Definition 3.2.3 are met. Observe that, as Fake picks values $c_1, \ldots, c_l$ uniformly and independently of each other, in each iteration of line 17 a value c_i is added to $\mathrm{Rng}(\tilde{\pi})$ that is uniform conditioned on the then current state of $\mathrm{Rng}(\tilde{\pi})$. Thus condition (1) holds. To establish the correctness bound of condition (2) we analyze the probability that Make aborts. With values $c_1, \ldots, c_{l-1}$, also the values $u_2, \ldots, u_l$ computed in line 14 are uniformly distributed and independent of each other, so the abort condition of line 15 is met with probability $(0 + \ldots + (l-1))/|\mathcal{D}| = ((l^2 - l)/2)/|\mathcal{D}|$ (accumulated over all iterations of the loop). The same bound holds for line 16. Plugging in the maximum value $l = L/\ell$ gives the bound claimed in the statement. Condition (3) is clear.

Algorithms Fake and Make for CBC-CTS-DEM are given in Figure 3.11. The analysis is similar. Here, however, we have $l = \lfloor L/\ell \rfloor$ and for lines 17 and 18 the accumulated probabilities of abort amount to $(0 + \ldots + l)/|\mathcal{D}|$ each. ∎

$\mathsf{Fake}(k', |m|)$
Write $|m|$ as $l\ell$
$c_1, \ldots, c_l \leftarrow_\$ \mathcal{D}$
$\bar{c} \leftarrow c_1 \ldots c_l$
$t \leftarrow \mathrm{khf}(k', \bar{c})$
$c \leftarrow \bar{c}t$
$st \leftarrow (c_1, \ldots, c_l)$
Return c, st

$\mathsf{Make}(st, m)$
$\tilde{\pi} \leftarrow \emptyset$
Write $|m|$ as $l\ell$
Parse st as $(c_1, \ldots, c_l)$
Split m into $m_1 \ldots m_l$
$c_0 \leftarrow 0^\ell$
For $i \leftarrow 1$ to l:
  $u_i \leftarrow m_i \oplus c_{i-1}$
  If $u_i \in \mathrm{Dom}(\tilde{\pi})$: Abort
  If $c_i \in \mathrm{Rng}(\tilde{\pi})$: Abort
  $\tilde{\pi} \leftarrow \tilde{\pi} \cup \{(u_i, c_i)\}$
Return $\tilde{\pi}$

Figure 3.10: Fake and Make for CBC-DEM. We write 'Abort' as an abbreviation for 'Return $\emptyset$'.

$\mathsf{Fake}(k', |m|)$
Write $|m|$ as $l\ell + l^*$
$c_1, \ldots, c_{l+1} \leftarrow_\$ \mathcal{D}$
$c_l^* \leftarrow \mathrm{msb}_{l^*}(c_l)$
$\bar{c} \leftarrow c_1 \ldots c_{l-1} c_l^* c_{l+1}$
$t \leftarrow \mathrm{khf}(k', \bar{c})$
$c \leftarrow \bar{c}t$
$st \leftarrow (c_1, \ldots, c_{l+1})$
Return c, st

$\mathsf{Make}(st, m)$
$\tilde{\pi} \leftarrow \emptyset$
Write $|m|$ as $l\ell + l^*$
Parse st as $(c_1, \ldots, c_{l+1})$
Split m into $m_1 \ldots m_l m_{l+1}^*$
$m_{l+1} \leftarrow m_{l+1}^* \,\|\, 0^{\ell - l^*}$
$c_0 \leftarrow 0^\ell$
For $i \leftarrow 1$ to $l+1$:
  $u_i \leftarrow m_i \oplus c_{i-1}$
  If $u_i \in \mathrm{Dom}(\tilde{\pi})$: Abort
  If $c_i \in \mathrm{Rng}(\tilde{\pi})$: Abort
  $\tilde{\pi} \leftarrow \tilde{\pi} \cup \{(u_i, c_i)\}$
Return $\tilde{\pi}$

Figure 3.11: Fake and Make for CBC-CTS-DEM. We write 'Abort' as an abbreviation for 'Return $\emptyset$'.

3.3.3 CCM

We analyze the CCM mode of operation ('CTR mode with CBC-MAC') with fixed nonce and associated data field; we call this mode CCM0-DEM. CCM is parameterized by an authentication tag length T, a formatting function $F\colon \mathcal{N} \times \mathcal{A} \times \mathcal{M} \to \mathcal{D}^+$ (where $\mathcal{N}$ and $\mathcal{A}$ denote the nonce space and the associated data space, respectively), and a counter generation function $G\colon \mathcal{N} \times [0 \mathinner{..} V] \to \mathcal{D}$, where V is a sufficiently large value. While only one set of instantiations of F and G is suggested in SP 800-38C (and if it is chosen the resulting version of CCM is the one used in wireless encryption standard IEEE 802.11), the specification is explicitly modular in the sense that it works with any F and G that meet certain conditions. Amongst others, the conditions listed in [Dwo07a] imply that for all $N \in \mathcal{N}$ the function $G(N;\cdot)$ is injective and that for all $(N, A, m) \in \mathcal{N} \times \mathcal{A} \times \mathcal{M}$ and $z_0 \ldots z_r = F(N, A, m)$ we have that $z_0 \notin G(N, [0 \,..\, V])$. Now, if we fix any nonce N_0 and any associated data string A_0 (e.g., the all-zero string for N_0 and the empty string for A_0) and define the restrictions $F_0 \colon \mathcal{M} \to \mathcal{D}^+;\ m \mapsto F(N_0, A_0, m)$ and $G_0 \colon [0 \,..\, V] \to \mathcal{D};\ i \mapsto G(N_0, i)$, then the algorithms of the resulting oracle DEM associated with CCM are given in Figure 3.12. The plaintext space of CCM0-DEM is $\mathcal{M} = \{0,1\}^*$ and the ciphertext space is $\mathcal{C} = \{0,1\}^{\geq T}$.

$\mathsf{O.Enc}^\pi(k', m)$	$\mathsf{O.Dec}^\pi(k', c)$
01 $z_0 \ldots z_r \leftarrow F_0(m)$	20 If $\lvert c\rvert < T$: Return $\perp$
02 $y_0 \leftarrow \pi(z_0)$	21 Write $\lvert c\rvert$ as $(l-1)\ell + l^* + T$
03 For $i \leftarrow 1$ to r:	22 Split c into $c_1 \ldots c_{l-1} c_l^* t^*$
04 $\quad x_i \leftarrow z_i \oplus y_{i-1}$	23 $c_l \leftarrow c_l^* \,\|\, 0^{\ell - l^*}$
05 $\quad y_i \leftarrow \pi(x_i)$	24 For $j \leftarrow 1$ to l:
06 $u_0 \leftarrow G_0(0)$	25 $\quad u_j \leftarrow G_0(j)$
07 $v_0 \leftarrow \pi(u_0)$	26 $\quad v_j \leftarrow \pi(u_j)$
08 $t \leftarrow y_r \oplus v_0$	27 $\quad m_j \leftarrow c_j \oplus v_j$
09 $t^* \leftarrow \mathrm{msb}_T(t)$	28 $m_l^* \leftarrow \mathrm{msb}_{l^*}(m_l)$
10 Write $\lvert m\rvert$ as $(l-1)\ell + l^*$	29 $m \leftarrow m_1 \ldots m_{l-1} m_l^*$
11 Split m into $m_1 \ldots m_{l-1} m_l^*$	30 $z_0 \ldots z_r \leftarrow F_0(m)$
12 $m_l \leftarrow m_l^* \,\|\, 0^{\ell - l^*}$	31 $y_0 \leftarrow \pi(z_0)$
13 For $j \leftarrow 1$ to l:	32 For $i \leftarrow 1$ to r:
14 $\quad u_j \leftarrow G_0(j)$	33 $\quad x_i \leftarrow z_i \oplus y_{i-1}$
15 $\quad v_j \leftarrow \pi(u_j)$	34 $\quad y_i \leftarrow \pi(x_i)$
16 $\quad c_j \leftarrow m_j \oplus v_j$	35 $u_0 \leftarrow G_0(0)$
17 $c_l^* \leftarrow \mathrm{msb}_{l^*}(c_l)$	36 $v_0 \leftarrow \pi(u_0)$
18 $c \leftarrow c_1 \ldots c_{l-1} c_l^* t^*$	37 $t \leftarrow y_r \oplus v_0$
19 Return c	38 If $t^* \neq \mathrm{msb}_T(t)$: Return $\perp$
	39 Return m

Figure 3.12: CCM0-DEM. Lines 10 and 21 uniquely identify quantities l and l^* such that $l \in \mathbb{N}^{\geq 1}$ and $0 \leq l^* < \ell$, and $\lvert m\rvert = (l-1)\ell + l^*$ and $\lvert c\rvert = (l-1)\ell + l^* + T$, respectively. Correspondingly, line 11 assumes $\lvert m_1\rvert = \ldots = \lvert m_{l-1}\rvert = \ell$ and $\lvert m_l^*\rvert = l^*$, and line 22 assumes $\lvert c_1\rvert = \ldots = \lvert c_{l-1}\rvert = \ell$ and $\lvert c_l^*\rvert = l^*$ and $\lvert t^*\rvert = T$.

Lemma 3.3.3 *CCM0-DEM is ε-simulatable with $\varepsilon \leq \lfloor L/\ell \rfloor^2 / 2^{\ell-2}$, where L is the maximum plaintext length (in bits).*

Proof. Consider algorithms Fake and Make from Figure 3.13. The idea of Fake is to compute the visible ciphertext components on basis of uniformly distributed blockcipher outputs while completely ignoring the blockcipher invocations of CCM's internal CBC-MAC computation (see how line 08 and l-many iterations of line 16 of $\mathsf{O.Enc}$ (in Figure 3.12) are replaced by lines 40 and 43 of Fake, while lines 02 and 05 of $\mathsf{O.Enc}$

have no counterpart). Inspection shows that, given m, algorithm Make finds a minimal partial permutation $\tilde{\pi}$ such that Fake and Make jointly mimic the behaviour of O.Enc (see here how lines 52 – 55, 58 – 61, 63 – 66, 71 – 74 of Make arrange the entries of $\tilde{\pi}$ such that they are consistent with lines 02, 05, 07/08, 15/16 of O.Enc). In some invocations of the algorithms, the described process might fail (in lines 53/54, 59/60, 64/65, 72/73), namely when partial permutation $\tilde{\pi}$ would become inconsistent. In such cases Make aborts, outputting the empty partial permutation $\tilde{\pi} = \emptyset$.

Fake$(k', |m|)$
40 $t \leftarrow_{\$} \mathcal{D}$
41 $t^* \leftarrow \mathrm{msb}_T(t)$
42 Write $|m|$ as $(l-1)\ell + l^*$
43 $c_1, \ldots, c_l \leftarrow_{\$} \mathcal{D}$
44 $c_l^* \leftarrow \mathrm{msb}_{l^*}(c_l)$
45 $c \leftarrow c_1 \ldots c_{l-1} c_l^* t^*$
46 $st \leftarrow (t, c_1, \ldots, c_l)$
47 Return c, st

Make(st, m)
48 $\tilde{\pi} \leftarrow \emptyset$
49 Write $|m|$ as $(l-1)\ell + l^*$
50 Parse st as $(t, c_1, \ldots, c_l)$
51 $z_0 \ldots z_r \leftarrow F_0(m)$
52 $y_0 \leftarrow_{\$} \mathcal{D}$
53 If $z_0 \in \mathrm{Dom}(\tilde{\pi})$: Abort
54 If $y_0 \in \mathrm{Rng}(\tilde{\pi})$: Abort
55 $\tilde{\pi} \leftarrow \tilde{\pi} \cup \{(z_0, y_0)\}$
56 For $i \leftarrow 1$ to r:
57 $\quad x_i \leftarrow z_i \oplus y_{i-1}$
58 $\quad y_i \leftarrow_{\$} \mathcal{D}$
59 $\quad$ If $x_i \in \mathrm{Dom}(\tilde{\pi})$: Abort
60 $\quad$ If $y_i \in \mathrm{Rng}(\tilde{\pi})$: Abort
61 $\quad \tilde{\pi} \leftarrow \tilde{\pi} \cup \{(x_i, y_i)\}$
62 $u_0 \leftarrow G_0(0)$
63 $v_0 \leftarrow y_r \oplus t$
64 If $u_0 \in \mathrm{Dom}(\tilde{\pi})$: Abort
65 If $v_0 \in \mathrm{Rng}(\tilde{\pi})$: Abort
66 $\tilde{\pi} \leftarrow \tilde{\pi} \cup \{(u_0, v_0)\}$
67 Split m into $m_1 \ldots m_{l-1} m_l^*$
68 $m_l \leftarrow m_l^* \,\|\, 0^{\ell - l^*}$
69 For $j \leftarrow 1$ to l:
70 $\quad u_j \leftarrow G_0(j)$
71 $\quad v_j \leftarrow m_j \oplus c_j$
72 $\quad$ If $u_j \in \mathrm{Dom}(\tilde{\pi})$: Abort
73 $\quad$ If $v_j \in \mathrm{Rng}(\tilde{\pi})$: Abort
74 $\quad \tilde{\pi} \leftarrow \tilde{\pi} \cup \{(u_j, v_j)\}$
75 Return $\tilde{\pi}$

Figure 3.13: Fake and Make for CCM0-DEM. We write ‘Abort’ as an abbreviation for ‘Return $\emptyset$’.

We next show that the requirements from Definition 3.2.3 are met. To see that condition (1) holds, observe that in Make the values y_0, y_i, v_0, and v_j are uniformly distributed and independent of each other at the point they are added to $\mathrm{Rng}(\tilde{\pi})$ in lines 55, 61, 66, 74. To establish the correctness bound of condition (2) we assess the probability that Make aborts. Using a similar analysis as in the proof of Lemma 3.3.1 we obtain the following (accumulated) probabilities: The abort conditions in lines 53 and 54 are never met; for lines 59 and 60 the probabilities are $(1 + \ldots + r)/|\mathcal{D}|$ each;

by the properties of CCM's functions F_0 and G_0, for lines 64 and 65 the probabilities are $r/|\mathcal{D}|$ and $(r+1)/|\mathcal{D}|$; for line 72 the probability is $lr/|\mathcal{D}|$; finally, for line 73 the probability is $((r+2)+\ldots+(r+l+1))/|\mathcal{D}|$. If we assume reasonable behavior of function F_0 and let $r=l$, we obtain quantity $4l^2/|\mathcal{D}|$ as an upper bound for the sum of these probabilities. This establishes the claimed bound. Condition (3) is clear. ∎

3.3.4 GCM

The GCM mode of operation ('Galois/Counter Mode') is a nonce-based AEAD parameterized by an authentication tag length T. To deploy GCM as a DEM we use it with a fixed nonce and an empty associated data field and call this version GCM0-DEM. Internally, GCM combines CTR mode encryption with a Wegman-Carter-style MAC [WC81]. The former uses an injective counter generation function $G\colon [0\mathinner{..}V] \to \mathcal{D}\setminus\{0^\ell\}$, where V is a sufficiently large value, and the latter is built around a polynomial-based universal hash function named GHASH defined over finite field $\mathrm{GF}(2^\ell)$. For our purposes it suffices to represent the MAC by a keyed hash function of the form $\mathrm{khf}\colon \mathcal{D}\times\{0,1\}^* \to \mathcal{D}$. The algorithms of GCM0-DEM, in the abstraction of an oracle DEM, are specified in Figure 3.14. The supported plaintext space is $\mathcal{M}=\{0,1\}^*$, and the ciphertext space is $\mathcal{C}=\{0,1\}^{\geq T}$.

Lemma 3.3.4 *GCM0-DEM is ε-simulatable with $\varepsilon \leq (\lceil L/\ell\rceil^2 + 4\lceil L/\ell\rceil)/2^{\ell-1}$, where L is the maximum plaintext length (in bits).*

Proof. The structure of GCM0-DEM is quite similar to the one of CTR0-DEM: both modes first encrypt the plaintext using CTR mode, then they append a MAC tag to the ciphertext. Two potentially interesting differences are that (a) in GCM0-DEM, the MAC key is derived by enciphering the value 0^ℓ under the blockcipher, and (b) in GCM0-DEM, the MAC tag is a GHASH value that is blinded with a blockcipher output (as is standard for Wegman-Carter MACs). Despite these differences, extending the proof of Lemma 3.3.1 to the GCM setting is straight-forward. The corresponding Fake and Make algorithms are given in Figure 3.15 and do not require further explanation.

We show that the requirements from Definition 3.2.3 are met. To see that condition (1) holds, observe that in Make the values v_i, v, and v_0 are uniformly distributed and independent of each other at the point they are added to $\mathrm{Rng}(\tilde{\pi})$ in lines 20, 27, 33. To establish the correctness bound of condition (2) we assess the probability that Make aborts. The analysis is particularly simple: the conditions in lines 18, 25, 31 are never met by construction, and the conditions in lines 19, 26, 32 are met with a total

$\mathsf{O.Enc}^{\pi}(k', m)$

01 Write $|m|$ as $(l-1)\ell + l^*$
02 Split m into $m_1 \ldots m_{l-1} m_l^*$
03 $m_l \leftarrow m_l^* \,\|\, 0^{\ell - l^*}$
04 For $i \leftarrow 1$ to l:
05 $\quad u_i \leftarrow G(i)$
06 $\quad v_i \leftarrow \pi(u_i)$
07 $\quad c_i \leftarrow m_i \oplus v_i$
08 $c_l^* \leftarrow \mathrm{msb}_{l^*}(c_l)$
09 $\bar{c} \leftarrow c_1 \ldots c_{l-1} c_l^*$
10 $u \leftarrow 0^{\ell}$
11 $v \leftarrow \pi(u)$
12 $h \leftarrow \mathrm{khf}(v, \bar{c})$
13 $u_0 \leftarrow G(0)$
14 $v_0 \leftarrow \pi(u_0)$
15 $t \leftarrow h \oplus v_0$
16 $t^* \leftarrow \mathrm{msb}_T(t)$
17 $c \leftarrow \bar{c} t^*$
18 Return c

$\mathsf{O.Dec}^{\pi}(k', c)$

19 If $|c| < T$: Return $\perp$
20 Write $|c|$ as $(l-1)\ell + l^* + T$
21 Split c into $\bar{c} t^*$
22 $u \leftarrow 0^{\ell}$
23 $v \leftarrow \pi(u)$
24 $h \leftarrow \mathrm{khf}(v, \bar{c})$
25 $u_0 \leftarrow G(0)$
26 $v_0 \leftarrow \pi(u_0)$
27 $t \leftarrow h \oplus v_0$
28 If $t^* \neq \mathrm{msb}_T(t)$: Return $\perp$
29 Split $\bar{c}$ into $c_1 \ldots c_{l-1} c_l^*$
30 $c_l \leftarrow c_l^* \,\|\, 0^{\ell - l^*}$
31 For $i \leftarrow 1$ to l:
32 $\quad u_i \leftarrow G(i)$
33 $\quad v_i \leftarrow \pi(u_i)$
34 $\quad m_i \leftarrow c_i \oplus v_i$
35 $m_l^* \leftarrow \mathrm{msb}_{l^*}(m_l)$
36 $m \leftarrow m_1 \ldots m_{l-1} m_l^*$
37 Return m

Figure 3.14: GCM0-DEM. Lines 01 and 20 uniquely identify quantities l and l^* such that $l \in \mathbb{N}^{\geq 1}$ and $0 \leq l^* < \ell$, and $|m| = (l-1)\ell + l^*$ and $|c| = (l-1)\ell + l^* + T$, respectively. Correspondingly, line 02 assumes $|m_1| = \ldots = |m_{l-1}| = \ell$ and $|m_l^*| = l^*$, line 21 assumes $|t^*| = T$, and line 29 assumes $|c_1| = \ldots = |c_{l-1}| = \ell$ and $|c_l^*| = l^*$.

probability of $(0 + \ldots + (l+1))/|\mathcal{D}|$. This establishes the claimed bound. Condition (3) is clear. ∎

$\mathsf{Fake}(k', \lvert m \rvert)$	$\mathsf{Make}(st, m)$

Write $\lvert m \rvert$ as $(l-1)\ell + l^*$
$c_1, \ldots, c_l \leftarrow_\$ \mathcal{D}$
$c_l^* \leftarrow \mathrm{msb}_{l^*}(c_l)$
$\bar{c} \leftarrow c_1 \ldots c_{l-1} c_l^*$
$t \leftarrow_\$ \mathcal{D}$
$t^* \leftarrow \mathrm{msb}_T(t)$
$c \leftarrow \bar{c} t^*$
$st \leftarrow (c_1, \ldots, c_l, t)$
Return c, st
	$\tilde{\pi} \leftarrow \emptyset$
	Write $\lvert m \rvert$ as $(l-1)\ell + l^*$
	Parse st as $(c_1, \ldots, c_l, t)$
	Split m into $m_1 \ldots m_{l-1} m_l^*$
	$m_l \leftarrow m_l^* \,\\|\, 0^{\ell - l^*}$
	For $i \leftarrow 1$ to l:
	    $u_i \leftarrow G(i)$
	    $v_i \leftarrow m_i \oplus c_i$
	    If $u_i \in \mathrm{Dom}(\tilde{\pi})$: Abort
	    If $v_i \in \mathrm{Rng}(\tilde{\pi})$: Abort
	    $\tilde{\pi} \leftarrow \tilde{\pi} \cup \{(u_i, v_i)\}$
	$c_l^* \leftarrow \mathrm{msb}_{l^*}(c_l)$
	$\bar{c} \leftarrow c_1 \ldots c_{l-1} c_l^*$
	$u \leftarrow 0^\ell$
	$v \leftarrow_\$ \mathcal{D}$
	If $u \in \mathrm{Dom}(\tilde{\pi})$: Abort
	If $v \in \mathrm{Rng}(\tilde{\pi})$: Abort
	$\tilde{\pi} \leftarrow \tilde{\pi} \cup \{(u, v)\}$
	$h \leftarrow \mathrm{khf}(v, \bar{c})$
	$u_0 \leftarrow G(0)$
	$v_0 \leftarrow h \oplus t$
	If $u_0 \in \mathrm{Dom}(\tilde{\pi})$: Abort
	If $v_0 \in \mathrm{Rng}(\tilde{\pi})$: Abort
	$\tilde{\pi} \leftarrow \tilde{\pi} \cup \{(u_0, v_0)\}$
	Return $\tilde{\pi}$

Figure 3.15: Fake and Make for GCM0-DEM. We write 'Abort' as an abbreviation for 'Return $\emptyset$'.

3.4 Selective Opening Secure Hybrid Encryption

We anticipated the main result of this chapter in Section 3.2: A PKE scheme constructed from any KEM and a permutation-driven DEM offers SIM-SO-CCA security in the ideal cipher model, if the KEM provides confidentiality (IND-CCA), the DEM provides authenticity (OT-INT-CTXT), and the DEM is simulatable (see Definition 3.2.3). Prerequisites like IND-CCA and OT-INT-CTXT on the KEM and DEM, respectively, are standard for proofs of the IND-CCA security of hybrid encryption, so the important finding is that the added constraint of simulatability suffices to lift security to the stronger notion of SIM-SO-CCA security.[4]

We discussed an informal version of our result in Section 3.2.2. Recall from the

[4] We note that a typical proof of IND-CCA security of hybrid PKE requires the DEM to also offer some kind of confidentiality (e.g., OT-IND-CCA). A corresponding notion appears only implicitly in our theorem statement, as it follows from the DEM's simulatability (in the ideal cipher model).

included proof sketch that an important subgoal was bounding the probability of the ideal cipher being evaluated on input a key established by the KEM before a corresponding Open query is posed. (If the cipher is evaluated earlier, the partial permutation found by Fake and Make cannot be smoothly embedded into it any more.) In the following we argue that without putting further restrictions on the KEM, bounding this probability to any small value is in general impossible. Indeed, consider for a moment a KEM where $\mathsf{KEM.Enc}$, before outputting a key k and a ciphertext c, evaluates the blockcipher used by $\mathsf{DEM.Enc}$ on input key k and a value d_0, where the latter is any fixed element $d_0 \in \mathcal{D}$ in the cipher's domain, and assume $\mathsf{KEM.Enc}$ completely ignores the result. Even though this blockcipher evaluation is completely pointless and should not affect security of the overall design, for such a KEM our arguments would not work. Below, in the formal version of our theorem statement, we correspondingly restrict the set of considered KEMs to those that do not evaluate the blockcipher at all. This admittedly is a limitation of our result, but we believe it is a mild one. Indeed, all practical KEMs we are aware of do not (internally) invoke blockcipher operations at all. This holds in particular for Hashed Elgamal, PSEC-KEM, Cramer-Shoup KEM, and RSA-KEM. In the following theorem statement, if E is a blockcipher, we say a KEM is *E-independent* if no KEM algorithm evaluates E^+ or E^-.

We proceed with the statement and proof of our main theorem.

Theorem 3.4.1 *Let* DEM *be a* $(\mathcal{K}, \mathcal{D})$*-permutation-driven DEM with corresponding oracle DEM* oDEM *and blockcipher* E*. Let* KEM *denote an* E*-independent KEM for the key space of the DEM. Let* PKE *denote the hybrid PKE scheme obtained when instantiating Construction 3.1.7 in Figure 3.3 with* KEM *and* DEM*.*

Let DEM *be* $(\tau_{ctxt}, q_{d,ctxt}, \varepsilon_{ctxt})$*-OT-INT-CTXT secure and* KEM $(\tau_{cca}, q_{d,cca}, \varepsilon_{cca})$*-IND-CCA secure.*

If oDEM *is* ε_{sim}*-simulatable, then* PKE *is* $(\tau_{so\text{-}cca}, q_{d,so\text{-}cca}, q_{ic}, \varepsilon_{so\text{-}cca})$*-SIM-SO-CCA secure where*

$$\tau_{so\text{-}cca} \leq \min\{\tau_{ctxt}, \tau_{cca}\} \ , \qquad \varepsilon(n) \leq n \cdot \left(3 \cdot \varepsilon_{cca} + \varepsilon_{ctxt} + \varepsilon_{sim} + 2 \cdot \frac{n + q_{ic} + q_d}{|\mathcal{K}|}\right) \ ,$$

and $q_{d,so\text{-}cca} \leq \min\{q_{d,ctxt}, q_{d,cca}\}$*. Further,* E *is modeled as an ideal cipher that may be queried at most* q_{ic} *times by an adversary.*

See Section 3.2.2 for a proof sketch including the high-level ideas. We proceed with a detailed proof of Theorem 3.4.1.

Proof of Theorem 3.4.1. For the keys $(k_i, k'_i) \leftarrow k''_i$ output by the n iterations of $\mathsf{KEM.Enc}$, and $\mathcal{J} \subseteq [n]$ let $K_{\mathcal{J}}$ denote the set $\{k_j \mid j \in \mathcal{J}\}$ of blockcipher keys k_i

$\mathcal{I} \leftarrow \emptyset$

01 For all $k \in \mathcal{K}$: $E_k \leftarrow \emptyset$
02 $\mathcal{I} \leftarrow \emptyset$; $\mathbf{c} \leftarrow \emptyset$
03 $(pk, sk) \leftarrow_{\$} \mathsf{KEM.Gen}$
04 $(\mathfrak{D}, st) \leftarrow_{\$} \mathcal{A}_1^{\mathrm{E},\mathrm{PKE.DEC}}(pk, n)$

$(m_1, \ldots, m_n) \leftarrow_{\$} \mathfrak{D}$

05 For $i \leftarrow 1$ to n:
06 $\quad r_i \leftarrow_{\$} \mathcal{R}$
07 $\quad (k''_i, c_i^{(1)}) \leftarrow \mathsf{KEM.Enc}_{pk}(r_i)$
08 $\quad (k_i, k'_i) \leftarrow k''_i$
09 $\quad$ If $k_i \in K_{[i-1]} \cup \mathrm{supp}(E)$: Abort
10 $\quad (c_i^{(2)}, st_i) \leftarrow_{\$} \mathsf{Fake}(k'_i, |m_i|)$
11 $\quad c_i \leftarrow \langle c_{i,1}, c_{i,2} \rangle$
12 $\mathbf{c} \leftarrow (c_1, \ldots, c_n)$
13 $out \leftarrow_{\$} \mathcal{A}_2^{\mathrm{E},\mathrm{OPEN},\mathrm{PKE.DEC}}(st, \mathbf{c})$

Stop with $\mathsf{Pred}(\mathfrak{D}, m_1, \ldots, m_n, \mathcal{I}, out)$

Oracle OPEN(i)
14 $\mathcal{I} \leftarrow \mathcal{I} \cup \{i\}$
15 If $k_i \in K_{[i-1]} \cup \mathrm{supp}(E)$: Abort
16 $\tilde{\pi} \leftarrow_{\$} \mathsf{Make}(st_i, m_i)$
17 $E_{k_i} \leftarrow \tilde{\pi}$
18 If $c_i^{(2)} \neq \mathsf{O.Enc}^{\mathrm{E}(k_i;\cdot)}(k'_i, m_i)$: Abort
19 Return (m_i, r_i)

Oracle PKE.DEC($\langle c^{(1)}, c^{(2)} \rangle$)
20 If $\langle c^{(1)}, c^{(2)} \rangle \in \mathbf{c}$: Abort
21 If $c_1 \in \mathbf{c}^{(1)}_{[n]\setminus\mathcal{I}}$: Return $\perp$
22 $k'' \leftarrow \mathsf{KEM.Dec}_{sk}(c^{(1)})$
23 If $k'' = \perp$: Return $\perp$
24 $(k, k') \leftarrow k''$
25 $m \leftarrow \mathsf{O.Dec}^{\mathrm{E}(k,\cdot)}(k', c^{(2)})$
26 Return m

Oracle $\mathrm{E}^+(k, \alpha)$
27 If $k \in K_{[n]\setminus\mathcal{I}}$: Abort
28 If $\alpha \notin \mathrm{Dom}(E_k^+)$:
29 $\quad \beta \leftarrow_{\$} \mathcal{D} \setminus \mathrm{Rng}(E_k^+)$
30 $\quad E_k \leftarrow E_k \cup \{(\alpha, \beta)\}$
31 Return β

Oracle $\mathrm{E}^-(k, \beta)$
32 If $k \in K_{[n]\setminus\mathcal{I}}$: Abort
33 If $\beta \notin \mathrm{Dom}(E_k^-)$:
34 $\quad \alpha \leftarrow_{\$} \mathcal{D} \setminus \mathrm{Rng}(E_k^-)$
35 $\quad E_k \leftarrow E_k \cup \{(\alpha, \beta)\}$
36 Return α

Figure 3.16: Proposed simulator $\mathcal{S} = (\mathcal{S}_1, \mathcal{S}_2)$ inlined into the i-SO-CCA experiment. $\mathcal{S}_1$ in lines 01 – 04, $\mathcal{S}_2$ given in lines 05 – 13. Instructions in gray boxes are executed by the ideal experiment. The whole code corresponds to the last experiment Exp_6 in our proof. For $\mathcal{J} \subseteq [n]$ we denote $K_{\mathcal{J}} := \{k_j \mid j \in \mathcal{J}\}$. Further, we denote $\mathrm{supp}(E) := \{k \in \mathcal{K} \mid E_k \neq \emptyset\}$.

for $i \in \mathcal{J}$. For the family of partial permutations $(E_k)_{k \in \mathcal{K}}$ maintained by $\mathcal{S}$ to implement ideal cipher E, let $\mathrm{supp}(E) := \{k \in \mathcal{K} \mid E_k \neq \emptyset\}$ denote the set of keys $k \in \mathcal{K}$ where partial permutation E_k is not empty.

Let $\mathcal{A}_{so} = (\mathcal{A}_{so,1}, \mathcal{A}_{so,2})$ denote an attacker against the $(\tau_{so\text{-}cca}, q_{d,so\text{-}cca}, q_{ic}, \varepsilon_{so\text{-}cca})$-SIM-SO-CCA security of PKE.

We define a simulator $(\mathcal{S}_1, \mathcal{S}_2)$ by giving its pseudocode in Figure 3.16. Simulator $\mathcal{S}_1$ consists of lines 01 – 04, $\mathcal{S}_2$ consists of lines 05 – 13. Their code is enhanced by bookkeeping and abort events, while the explicit invocation of $\mathcal{S}_1$, $\mathcal{S}_2$ and their input/output behaviour is merged into the ideal experiment. Instructions in gray boxes are performed by the ideal experiment.

We show that $\mathcal{S}$, when run in the ideal experiment, can simulate the real experiment for $\mathcal{A}_{so}$. To this end we proceed in a sequence of experiments tracing how likely it is for $\mathcal{A}_{so}$ to distinguish two consecutive experiments. The sequence interpolates between the real experiment ($\mathsf{Exp}_0 = \text{r-SO-CCA}$, see Figure 3.4) and a simulated real experiment (Exp_6, see Figure 3.16) provided by the simulator $\mathcal{S}$ inlined into the ideal experiment.

We proceed with detailed descriptions of the experiments given in Figures 3.17 to 3.19.

Exp $\mathsf{Exp}_0^{\mathcal{A}_{so}}(n) - \mathsf{Exp}_6^{\mathcal{A}_{so}}(n)$

01 For all $k \in \mathcal{K}$: $E_k \leftarrow \emptyset$
02 $\mathcal{I} \leftarrow \emptyset$; $C \leftarrow \emptyset$
03 $\textsc{Bad} \leftarrow \mathit{false}$ // Exp_4
04 $(pk, sk) \leftarrow_\$ \mathsf{KEM.Gen}$
05 $(\mathfrak{D}, st) \leftarrow_\$ \mathcal{A}_{so,1}^{\mathrm{E},\textsc{PKE.Dec}}(pk, n)$
06 $(m_1, \ldots, m_n) \leftarrow_\$ \mathfrak{D}$
07 For $i \leftarrow 1$ to n:
08 $\quad r_i \leftarrow_\$ \mathcal{R}$
09 $\quad (k_i'', c_i^{(1)}) \leftarrow \mathsf{KEM.Enc}_{pk}(r_i)$
10 $\quad (k_i, k_i') \leftarrow k_i''$
11 $\quad$ If $k_i \in K_{[i-1]} \cup \operatorname{supp}(E)$: Abort // $\mathsf{Exp}_2 - \mathsf{Exp}_6$
12 $\quad c_i^{(2)} \leftarrow \mathsf{O.Enc}^{\mathrm{E}(k_i;\cdot)}(k_i', m_i)$ // $\mathsf{Exp}_0 - \mathsf{Exp}_2$
13 $\quad (c_i^{(2)}, st_i) \leftarrow_\$ \mathsf{Fake}(k_i', |m_i|)$ // $\mathsf{Exp}_3 - \mathsf{Exp}_6$
14 $\quad \tilde{\pi} \leftarrow_\$ \mathsf{Make}(st_i, m_i)$ // $\mathsf{Exp}_3 - \mathsf{Exp}_5$
15 $\quad E_{k_i} \leftarrow \tilde{\pi}$ // $\mathsf{Exp}_3 - \mathsf{Exp}_5$
16 $\quad$ If $c_i^{(2)} \neq \mathsf{O.Enc}^{\mathrm{E}(k_i;\cdot)}(k_i', m_i)$: Abort // $\mathsf{Exp}_3 - \mathsf{Exp}_5$
17 $\quad c_i \leftarrow \langle c_i^{(1)}, c_i^{(2)} \rangle$
18 $\mathbf{c} \leftarrow (c_1, \ldots, c_n)$
19 $out \leftarrow_\$ \mathcal{A}_{so,2}^{\textsc{Open},\textsc{PKE.Dec},\mathrm{E}}(st, \mathbf{c})$
20 If $\textsc{Bad}$: Abort // Exp_4
21 Stop with $\mathsf{Pred}(\mathfrak{D}, m_1, \ldots, m_n, \mathcal{I}, out)$

Figure 3.17: Experiments Exp_0 – Exp_6 used in the proof of Theorem 3.4.1. Oracles Open, PKE.Dec, E^+ and E^- are given in Figure 3.18.

Experiment Exp_0. The r-SO-CCA experiment as given in Figure 3.4.

Experiment Exp_1. Line 23 is added: Any decryption query of the form $\langle c^{(1)}, c^{(2)} \rangle$ is answered with $\perp$ if $c^{(1)} \in \mathbf{c}^{(1)}_{[n]\setminus\mathcal{I}}$. That is, there exists $i \in [n]$ such that $c^{(1)} = c_i^{(1)}$ and $\mathcal{A}_{so,2}$ did not query $\textsc{Open}(i)$.

Claim 3.4.2 There exists an adversary $\mathcal{A}^{(1)}_{cca}$ that $(\tau^{(1)}_{cca}, q^{(1)}_{d,cca}, \varepsilon^{(1)}_{cca})$-breaks the IND-CCA security of KEM and an adversary $\mathcal{A}_{ctxt}$ that $(\tau_{ctxt}, q_{d,ctxt}, \varepsilon_{ctxt})$-breaks the OT-

Oracle $\text{PKE.D}\textsc{ec}(\langle c^{(1)}, c^{(2)} \rangle)$
If $\langle c^{(1)}, c^{(2)} \rangle \in \mathbf{c}$: Abort
If $c^{(1)} \in \mathbf{c}^{(1)}_{[n]\setminus\mathcal{I}}$: Return $\perp$ // $\mathsf{Exp}_1 - \mathsf{Exp}_6$
$k'' \leftarrow \mathsf{KEM.Dec}_{sk}(c^{(1)})$
If $k'' = \perp$: Return $\perp$
$(k, k') \leftarrow k''$
$m \leftarrow \mathsf{O.Dec}^{\mathrm{E}(k,\cdot)}(k', c^{(2)})$
Return m

Oracle $\textsc{Open}(i)$
$\mathcal{I} \leftarrow \mathcal{I} \cup \{i\}$
If $k_i \in K_{[i-1]} \cup \mathrm{supp}(E)$: Abort // Exp_6
$\tilde{\pi} \leftarrow_{\$} \mathsf{Make}(st_i, m_i)$ // Exp_6
$E_{k_i} \leftarrow \tilde{\pi}$ // Exp_6
If $c_i^{(2)} \neq \mathsf{O.Enc}^{\mathrm{E}(k_i;\cdot)}(k'_i, m_i)$: Abort // Exp_6
Return (m_i, r_i)

Figure 3.18: Provided Oracles in experiments $\mathsf{Exp}_0 - \mathsf{Exp}_6$ as given in Figure 3.17.

INT-CTXT security of DEM with

$$\tau^{(1)}_{cca} \approx \tau_{so\text{-}cca} \approx \tau_{ctxt} \quad , \qquad q^{(1)}_{d,cca} \geq q_{d,so\text{-}cca} \quad , \qquad q_{d,ctxt} \geq q_{d,so\text{-}cca},$$

$$\varepsilon^{(1)}_{cca} + \varepsilon_{ctxt} \geq \frac{1}{n} \cdot \left| \Pr\left[\mathsf{Exp}_0^{\mathcal{A}_{so}}(n) \Rightarrow 1\right] - \Pr\left[\mathsf{Exp}_1^{\mathcal{A}_{so}}(n) \Rightarrow 1\right] \right| \quad .$$

Proof of Claim 3.4.2. Experiments Exp_0 and Exp_1 proceed identically, until $\mathcal{A}_{so}$ submits a ciphertext $\langle c^{(1)}, c^{(2)} \rangle$ to decryption where $c^{(1)} \in \mathbf{c}^{(1)}_{[n]\setminus\mathcal{I}}$ and $\mathsf{PKE.Dec}_{sk}(\langle c_1, c_2 \rangle) \neq \perp$.

We fix some $i \in [n]$ and analyze the probability that $\mathcal{A}_{so}$ submits a ciphertext $\langle c^{(1)}, c^{(2)} \rangle$ where $c^{(1)} \in \mathbf{c}^{(1)}_{\{i\}\setminus\mathcal{I}}$ and $\mathsf{PKE.Dec}(sk, \langle c^{(1)}, c^{(2)} \rangle) \neq \perp$; we denote this event by '$\langle c_i^{(1)}, c^{(2)} \rangle \nrightarrow \perp$'.

We perform a preparational modification before bounding $\Pr[\langle c_i^{(1)}, c^{(2)} \rangle \nrightarrow \perp]$. To this end, we replace k''_i as output by the i^{th} invocation of $\mathsf{KEM.Enc}_{pk}$ with a uniformly random key.

We lose an additional summand of ε_{cca} in the bound on $\Pr[\langle c_{i,1}, c_2 \rangle \nrightarrow \perp]$ as shown by the following reduction run by adversary $\mathcal{A}^{(1)}_{cca} = (\mathcal{A}^{(1)}_{cca,1}, \mathcal{A}^{(1)}_{cca,2})$: Adversary $\mathcal{A}^{(1)}_{cca,1}$ is started on pk and invokes $\mathcal{A}_{so,1}(pk, n)$. It uses its decapsulation oracle to answer decryption queries from $\mathcal{A}_{so,1}$. When $\mathcal{A}_{so,1}$ outputs $\mathfrak{D}$, $\mathcal{A}^{(1)}_{cca,1}$ halts. When $\mathcal{A}^{(2)}_{cca,2}(c^*, k_b^*)$ is started, it parses $(k_b, k'_b) \leftarrow k_b^*$ and computes all ciphertexts faithfully except for $c_i \leftarrow \langle c^*, \mathsf{O.Enc}^{\mathrm{E}(k_b;\cdot)}(k'_b, m_i) \rangle$. Adversary $\mathcal{A}^{(1)}_{cca,2}$ calls $\mathcal{A}_{so,2}(c_1, \ldots, c_n)$. Decryption queries $\langle c^{(1)}, c^{(2)} \rangle$ by $\mathcal{A}_{so,2}$ are answered employing the decapsulation oracle for $c^{(1)} \neq c^*$ and using key k_b^* otherwise.

Analysis The reduction perfectly simulates Exp_1 until $\mathcal{A}_{so,2}$ queries Open(i) which the reduction cannot answer. However, to bound the probability of event '$\langle c_i^{(1)}, c^{(2)} \rangle \nrightarrow \bot$' it suffices to make sure that the reduction simulates $\mathcal{A}_{so}$'s interface as expected in the r-SO-CCA experiment as long as the event can occur. Note that '$\langle c_i^{(1)}, 2 \rangle \nrightarrow \bot$' cannot happen after query Open(i).

We now show how to break the OT-INT-CTXT security of the DEM assuming '$\langle c_i^{(1)}, c^{(2)} \rangle \nrightarrow \bot$' happens. We construct adversary $\mathcal{A}_{ctxt} = (\mathcal{A}_{ctxt,1}, \mathcal{A}_{ctxt,2})$. When $\mathcal{A}_{ctxt,1}$ is started, it runs $\mathsf{KEM.Gen}$ and starts $\mathcal{A}_{so,1}(pk, n)$. Decryption queries are answered using sk. Once $\mathcal{A}_{so,1}$ outputs $\mathfrak{D}$, $\mathcal{A}_{ctxt,1}$ samples plaintext $\mathbf{m} \leftarrow_\$ \mathfrak{D}$, outputs m_i and halts. Then $\mathcal{A}_{ctxt,2}(c_2^*)$ is started whereby $c^{(2)*} \leftarrow \mathsf{DEM.Enc}(k''_\$, m_i)$ constitutes a data encapsulation of m_i under a random key $k''_\$$. Additionally, $\mathcal{A}_{ctxt,2}$ runs $\mathsf{KEM.Enc}_{pk}$ to obtain $(k, c^{(1)*})$ and invokes $\mathcal{A}_{so,2}(c_1, \dots, c_{i-1}, \langle c^{(1)*}, c^{(2)*} \rangle, \dots, c_n)$. Adversary $\mathcal{A}_{ctxt,2}$ answers all further decryption queries on its own, unless the ciphertext is of the form $\langle c^{(1)*}, c^{(2)} \rangle$ where it submits $c^{(2)}$ to its decapsulation oracle DEM.Dec of the OT-INT-CTXT experiment and relays the reply to $\mathcal{A}_{so,2}$.

Analysis Clearly, $\mathcal{A}_{ctxt}$ wins the OT-INT-CTXT experiment when $\mathcal{A}_{so}$ submits a ciphertext that causes '$\langle c_i^{(1)}, c^{(2)} \rangle \nrightarrow \bot$' to happen.

We obtain $\Pr[\langle c_i^{(1)}, c^{(2)} \rangle \nrightarrow \bot] \leq \varepsilon'_{cca} + \varepsilon_{ctxt}$. Adversaries $\mathcal{A}_{cca}^{(1)}$ and $\mathcal{A}_{ctxt}$ (roughly) have the same running time and may have to issue a query to the KEM.Dec (resp. DEM.Dec) oracle when receiving a decryption query from $\mathcal{A}_{so}$. The claim follows from the union-bound over all $i \in [n]$. ■

Note that, ideally, one would wish to employ IND-CCA security of the KEM once to replace a key output by $\mathsf{KEM.Enc}_{pk}$ by a uniform key. However, once done, opening queries by $\mathcal{A}_{so,2}$ cannot be answered anymore.

The next modification ensures that (if it is not aborted) the i^{th} invocation of the oracle data encapsulation, i.e., $\mathsf{O.Enc}^{E(k_i;\cdot)}$, has access to an empty partial permutation E_{k_i}. This is a preparational step to ensure that later, when O.Enc is replaced with Fake and Make, the partial permutation output by Make can be embedded into E_{k_i}.

Experiment Exp_2. Line 11 is added. That is, Exp_2 aborts if the i^{th} iteration of O.Enc would have oracle access to a non-empty permutation $E(k_i; \cdot)$.[5]

[5] As of now, in the i^{th} iteration of the For loop, we have $K_{[i-1]} \subseteq \mathrm{supp}(E)$ as the invocation of $\mathsf{O.Enc}^{\mathrm{E}(k_i;\cdot)}$ adds elements to E_{k_i}. Later, in experiment Exp_6, we do not invoke code that (implicitly) adds elements to E_{k_i} and rely on set $K_{[i-1]}$ to detect collisions amongst the (blockcipher) keys.

Claim 3.4.3 There exists an adversary $\mathcal{A}_{cca}^{(2)}$ that $(\tau_{cca}^{(2)}, q_{d,cca}^{(2)}, \varepsilon_{cca}^{(2)})$-breaks the IND-CCA security of KEM where

$$\tau_{cca}^{(2)} \approx \tau_{so\text{-}cca} \ , \qquad q_{d,cca}^{(2)} \geq q_{d,so\text{-}cca} \ ,$$

and

$$\varepsilon_{cca}^{(2)} \geq \frac{1}{n} \cdot \left| \Pr\left[\mathsf{Exp}_1^{\mathcal{A}_{so}}(n) \Rightarrow 1\right] - \Pr\left[\mathsf{Exp}_2^{\mathcal{A}_{so}}(n) \Rightarrow 1\right] \right| - \frac{n + q_{ic} + q_d}{|\mathcal{K}|} \ .$$

Proof of Claim 3.4.3. We bound $\Pr[k_i \in K_{[i-1]} \cup \mathrm{supp}(E)]$ for fixed $i \in [n]$. Again, we replace k_i'' output by the i^{th} invocation of $\mathsf{KEM.Enc}_{pk}$ with a uniform key first. We show how to break KEM's IND-CCA security if the two experiments should differ noticeably.

We construct adversary $\mathcal{A}_{cca}^{(2)} = (\mathcal{A}_{cca,1}^{(2)}, \mathcal{A}_{cca,2}^{(2)})$. It is executed on pk and starts $\mathcal{A}_{so,1}(pk, n)$. Decryption queries are answered using the decapsulation oracle. When $\mathcal{A}_{so,1}$ halts, $\mathcal{A}_{cca,1}^{(2)}$ halts as well. Then $\mathcal{A}_{cca,2}^{(2)}(c^*, k_b^*)$ is started. Let $(k_b, k_b') \leftarrow k_b^*$. Next, $\mathcal{A}_{cca,2}^{(2)}$ runs the For loop from line 08. In the i^{th} iteration $\mathcal{A}_{cca,2}^{(2)}$ aborts $\mathcal{A}_{so}$ and returns 1 iff $k_b \in K_{[i-1]} \cup \mathrm{supp}(E)$.

Analysis The simulation is perfect until $\mathcal{A}_{cca,2}^{(2)}$ halts. Further we have

$$\varepsilon_{cca}^{(2)} \geq |\Pr[k_i \in K_{[i-1]} \cup \mathrm{supp}(E)] - \Pr[k_\$ \in K_{[i-1]} \cup \mathrm{supp}(E)]| \ ,$$

where $k_\$ \leftarrow_\$ \mathcal{K}$.

Note that each decryption query or query to the ideal cipher oracles adds at most one element to $\mathrm{supp}(E)$, hence $|K_{[i-1]} \cup \mathrm{supp}(E)| \leq n + q_{ic} + q_d$. Thus, we obtain

$$\Pr[k_\$ \in K_{[i-1]} \cup \mathrm{supp}(E)] \leq (n + q_{ic} + q_d) \ / \ |\mathcal{K}| \ ,$$

and

$$\Pr[k_i \in K_{[i-1]} \cup \mathrm{supp}(E)] \leq \varepsilon_{cca} + (n + q_{ic} + q_d) \ / \ |\mathcal{K}| \ .$$

The claim follows from the union-bound over $i \in [n]$ and rearranging. One easily checks that $\mathcal{A}_{cca}^{(2)}$ runs roughly as long as $\mathcal{A}_{so}$ and the bound on $q_{d,cca}^{(2)}$ holds. ∎

Experiment Exp_3. The faithful data encapsulation is replaced by algorithms Fake and Make. More precisely, for each iteration of the For loop (line 07) we replace the invocation $\mathsf{O.Dec}^{E(k_i;\cdot)}(k_i', m_i)$ (line 12) with running $\mathsf{Fake}(k_i', |m_i|)$ and $\mathsf{Make}(m_i)$ back to back (lines 13,14). E_{k_i} gets assigned partial permutation $\tilde{\pi}$ as output by Make (see line 15) and a check is performed whether E_{k_i} has been programmed 'consistently'; if

Oracle $\mathrm{E}^+(k, \alpha)$		**Oracle** $\mathrm{E}^-(k, \beta)$	
35 If $k \in K_{[n]\setminus\mathcal{I}}$:	// $\mathsf{Exp}_4 - \mathsf{Exp}_6$	42 If $k \in K_{[n]\setminus\mathcal{I}}$:	// $\mathsf{Exp}_4 - \mathsf{Exp}_6$
36 $\quad$ BAD $\leftarrow$ *true*	// Exp_4	43 $\quad$ BAD $\leftarrow$ *true*	// Exp_4
37 $\quad$ Abort	// $\mathsf{Exp}_5 - \mathsf{Exp}_6$	44 $\quad$ Abort	// $\mathsf{Exp}_5 - \mathsf{Exp}_6$
38 If $\alpha \notin \mathrm{Dom}(E_k^+)$:		45 If $\beta \notin \mathrm{Dom}(E_k^-)$:	
39 $\quad \beta \leftarrow_\$ \mathcal{D} \setminus \mathrm{Rng}(E_k^+)$		46 $\quad \alpha \leftarrow_\$ \mathcal{D} \setminus \mathrm{Rng}(E_k^-)$	
40 $\quad E_k \leftarrow E_k \cup \{(\alpha, \beta)\}$		47 $\quad E_k \leftarrow E_k \cup \{(\alpha, \beta)\}$	
41 Return β		48 Return α	

Figure 3.19: Ideal cipher oracles E^+, E^- provided in Exp_0 – Exp_6 as given in Figure 3.17.

not, experiment Exp_3 aborts (line 16).

Claim 3.4.4 $\left|\Pr\left[\mathsf{Exp}_2^{\mathcal{A}_{so}}(n) \Rightarrow 1\right] - \Pr\left[\mathsf{Exp}_3^{\mathcal{A}_{so}}(n) \Rightarrow 1\right]\right| \le n \cdot \varepsilon_{sim}$.

Proof of Claim 3.4.4. Fix $i \in [n]$. Due to the modifications in experiments Exp_1 and Exp_2, partial permutation E_{k_i} is empty at the time of invoking O.Enc. Hence, once we replace O.Enc by Fake and Make, the partial permutation as output by Make can always be embedded into E_{k_i}. Particularly, partial permutations E_{k_i} accessed by O.Enc and $\tilde{\pi}$ output by Make are identically distributed when randomly extended to a full permutation on $\mathcal{D}$. We conclude that the abort in line 16 happens with probability at most ε_{sim} as oDEM is ε_{sim}-simulatable. The claim follows from the union-bound over all $i \in [n]$. ■

Recall from the proof outline that, eventually, Make shall be run as part of the OPEN procedure. The upcoming modifications ensure that partial permutation E_{k_i} remains empty until OPEN(i) is queried.

Experiment Exp_4. Line 03 is added to initialize a flag BAD as *false*. Lines (35, 36) are added to the E^+ oracle, lines (42, 43) are added to the E^- oracle and line 20 is added. That is, if E^+ or E^- is queried on (k_i, z) for any z and $i \notin \mathcal{I}$, BAD is set to *true* and the experiment aborts *after* the execution of $\mathcal{A}_2$ (in line 20).

Claim 3.4.5 There exists an adversary $\mathcal{A}_{cca}^{(3)}$ that $(\tau_{cca}^{(3)}, q_{d,cca}^{(3)}, \varepsilon_{cca}^{(3)})$-breaks the IND-CCA security of KEM where $\tau_{cca}^{(3)} \approx \tau_{so\text{-}cca}$, $q_{d,cca}^{(3)} \ge q_{d,so\text{-}cca}$ and

$$\varepsilon_{cca}^{(3)} \ge \frac{1}{n} \cdot \left|\Pr\left[\mathsf{Exp}_3^{\mathcal{A}_{so}}(n) \Rightarrow 1\right] - \Pr\left[\mathsf{Exp}_4^{\mathcal{A}_{so}}(n) \Rightarrow 1\right]\right| - \frac{q_{ic} + q_d}{|\mathcal{K}|} .$$

Proof of Claim 3.4.5. Fix $i \in [n]$ and let '$k \in K_{\{i\}\setminus\mathcal{I}}$' denote the event that E^+ or E^- is queried on (k, z) where $k \in K_{\{i\}\setminus\mathcal{I}}$. (That is, the condition in lines 35 or 42 holds, even when replacing $K_{[n]\setminus\mathcal{I}}$ with $K_{\{i\}\setminus\mathcal{I}}$). Again, we replace key k_i'' output in the i^{th} invocation of $\mathsf{KEM.Enc}$ with a uniform key $(k_\$, k'_\$) \leftarrow k''_\$$. The reduction run by $\mathcal{A}^{(3)}_{cca} = (\mathcal{A}^{(3)}_{cca,1}, \mathcal{A}^{(3)}_{cca,2})$ proceeds as in the proof of Claim 3.4.3 to bridge Exp_0 and Exp_1. However, here, $\mathcal{A}^{(3)}_{cca,2}$ halts after $\mathcal{A}_{so,2}$'s execution and outputs 1 iff BAD is true.

ANALYSIS The reduction is perfect unless $\mathcal{A}_{so,2}$ queries OPEN(i) which cannot be answered. Similarly to before, it suffices to guarantee the correctness of the simulation as long as the abort in line 20 can potentially happen. Note that after query OPEN(i), BAD cannot be set to *true* as $K_{\{i\}\setminus\mathcal{I}} = \emptyset$. Hence, $|\Pr[k \in K_{\{i\}\setminus\mathcal{I}}] - \Pr[k \in \{k_\$\} \setminus \mathcal{I}]| \leq \varepsilon_{cca}$ for uniform $k_\$ \leftarrow_\$ \mathcal{K}$.

Further, $k_\$$ is uniform from $\mathcal{A}_{so}$'s view: Only ciphertext $\langle c_i^{(1)}, c_i^{(2)} \rangle$ might contain information on $k_\$$. However, $c^{(1)}$ is independent of $k_\$$ as it is sampled after $\mathsf{KEM.Enc}_{pk}$ outputs $c^{(1)}$ and data encapsulation $c_i^{(2)}$ is independent of $k_\$$ as we run $\mathsf{Fake}(k'_i, m_i)$ to compute $c_i^{(2)}$. Thus, $\Pr[k \in \{k_\$\} \setminus \mathcal{I}] \leq (q_{ic} + q_d)/|\mathcal{K}|$ and collecting the probabilities and applying the union-bound gives the desired bound.

One easily verifies the statements on the running time and decapsulation queries by $\mathcal{A}^{(3)}_{cca}$. ■

Experiment Exp_5. Lines 37 and 44 are added. Instead of aborting after the execution of $\mathcal{A}_2$ if BAD $=$*true*, experiment Exp_5 aborts as soon as BAD (as introduced in Exp_4) is set to *true*. Now obsolete lines 03, 20, 36 and 43 are removed for clarity. Note that this step is purely cosmetic since we condition our analysis on 'Abort does not happen' anyway.

Claim 3.4.6 $\Pr\left[\mathsf{Exp}_4^{\mathcal{A}_{so}}(n) \Rightarrow 1\right] = \Pr\left[\mathsf{Exp}_5^{\mathcal{A}_{so}}(n) \Rightarrow 1\right]$.

Proof of Claim 3.4.6. The claim follows from observing that experiment Exp_5 aborts in lines 37 or 44 if and only if experiment Exp_4 aborts in line 20. ■

Experiment Exp_6. An abort event is added in line 30. The invocation of Make, the embedding of a partial permutation and the consistency check are moved from the For loop in lines 14 – 16 to the OPEN oracle (lines 31 – 32).

Claim 3.4.7 $\Pr\left[\mathsf{Exp}_5^{\mathcal{A}_{so}}(n) \Rightarrow 1\right] = \Pr\left[\mathsf{Exp}_6^{\mathcal{A}_{so}}(n) \Rightarrow 1\right]$.

Proof of Claim 3.4.7. The abort event in line 30 is solely added for clarity but is never met: Assume that line 30 would cause an abort, then the condition in line 11, or lines 35/42 would have been satisfied earlier. Hence, for all $i \in [n]$: a) in experiment Exp_5 partial permutation $E_{k_i} \leftarrow \tilde{\pi}$ as output by Make in line 14 is information-theoretically hidden from $\mathcal{A}$ until it queries OPEN, and b) in Exp_6 partial permutation E_{k_i} remains empty until $\mathcal{A}$ queries OPEN. Thus, embedding partial permutation $\tilde{\pi}$ into E_{k_i} always succeeds. Further, moving the invocation of Make, the embedding and checking to the OPEN oracle is completely oblivious to $\mathcal{A}$. ■

We observe that the code as given in experiment Exp_6 in Figure 3.17 matches the code of the simulator as given in Figure 3.16. Further, observe that all IND-CCA adversaries $\mathcal{A}_{cca}^{(1)}$, $\mathcal{A}_{cca}^{(2)}$, $\mathcal{A}_{cca}^{(3)}$ have roughly the same running time and pose the same number of decryption queries. Further, for their winning probabilities we have

$$\max\{\varepsilon'_{cca}, \varepsilon''_{cca}, \varepsilon''_{cca}\} \leq \varepsilon_{cca} \ .$$

The claim of Theorem 3.4.1 follows by collecting the results from Claims 3.4.2 to 3.4.7.[6] ■

[6]Note that we obtain a slightly better bound than given in Theorem 3.4.1 that happens to be slightly messier.

Conclusion & Open Problems

In this thesis we presented contributions to the understanding of selective opening attacks, security against the former, respectively. Our results fall into two categories: Results in the standard model, and results in the random oracle (resp. ideal cipher) model.

Our standard model results in Part I are the first non-trivial implications results on the relation of IND-CPA and IND-SO-CPA security. Motivated by an observation on 'memomoryless' distributions we developed a new reduction. We could show that IND-CPA securtiy entails IND-SO-CPA security for a class of distributions strictly larger than what was previously known. However, the conditions imposed on distributions covered by our positive result are quite significant and restrict the distribution to be chain-like. Interestingly, we exploit the lack of dependencies while the separation result of [HRW16] relies on 'distributions with many dependencies'. As already mentioned in the introduction, the latter negative result and our positive result leaves an uncharted territory of distributions for which we do not know whether IND-CPA implies IND-SO-CPA security.

For our results in idealized models we first concentrated on well-known transformations that are known to obtain IND-CCA security in the random oracle model. Surprisingly, we could show that all transformations do obtain the (strictly) stronger notion of SIM-SO-CCA security. Yet, for these transformations we required the plaintext to be one-time padded with the output of random oracle in order to ensure efficient openability. Thus, the schemes covered by our results are of rather limited use in practice. However, we could generalize the concept to *simulatable* DEMs allowing for efficient openability for arbitrary plaintexts in the ideal cipher model. When combined with a standard IND-CCA secure KEM we showed that the obtained hybrid encryption scheme achieves the strong notion of SIM-SP-CCA as well. Note that for all results in Part II the use of idealized primitives allowed us to circumvent the negative result of [BDWY11] as no 'committing' PKE scheme can obtain SIM-SO security.

Bibliography

[ABR01] Michel Abdalla, Mihir Bellare, and Phillip Rogaway. The oracle Diffie-Hellman assumptions and an analysis of DHIES. In David Naccache, editor, *Topics in Cryptology – CT-RSA 2001*, volume 2020 of *Lecture Notes in Computer Science*, pages 143–158, San Francisco, CA, USA, April 8–12, 2001. Springer, Heidelberg, Germany. (Cited on page 14, 70, 82, 94.)

[BBM00] Mihir Bellare, Alexandra Boldyreva, and Silvio Micali. Public-key encryption in a multi-user setting: Security proofs and improvements. In Bart Preneel, editor, *Advances in Cryptology – EUROCRYPT 2000*, volume 1807 of *Lecture Notes in Computer Science*, pages 259–274, Bruges, Belgium, May 14–18, 2000. Springer, Heidelberg, Germany. (Cited on page 17.)

[BDL97] Dan Boneh, Richard A. DeMillo, and Richard J. Lipton. On the importance of checking cryptographic protocols for faults (extended abstract). In Walter Fumy, editor, *Advances in Cryptology – EUROCRYPT'97*, volume 1233 of *Lecture Notes in Computer Science*, pages 37–51, Konstanz, Germany, May 11–15, 1997. Springer, Heidelberg, Germany. (Cited on page 15.)

[BDWY11] Mihir Bellare, Rafael Dowsley, Brent Waters, and Scott Yilek. Standard security does not imply security against selective-opening. Cryptology ePrint Archive, Report 2011/581, 2011. `http://eprint.iacr.org/2011/581`. (Cited on page 19, 20, 103, 133.)

[BDWY12] Mihir Bellare, Rafael Dowsley, Brent Waters, and Scott Yilek. Standard security does not imply security against selective-opening. In David Pointcheval and Thomas Johansson, editors, *Advances in Cryptology – EUROCRYPT 2012*, volume 7237 of *Lecture Notes in Computer Science*, pages 645–662, Cambridge, UK, April 15–19, 2012. Springer, Heidelberg, Germany. (Cited on page 19, 27.)

[BF06] Alexandra Boldyreva and Marc Fischlin. On the security of OAEP. In Xuejia Lai and Kefei Chen, editors, *Advances in Cryptology – ASIACRYPT 2006*, volume 4284 of *Lecture Notes in Computer Science*, pages 210–225, Shanghai, China, December 3–7, 2006. Springer, Heidelberg, Germany. (Cited on page 84.)

[BHK12] Florian Böhl, Dennis Hofheinz, and Daniel Kraschewski. On definitions of selective opening security. In Marc Fischlin, Johannes Buchmann, and Mark Manulis, editors, *PKC 2012: 15th International Conference on Theory and Practice of Public Key Cryptography*, volume 7293 of *Lecture Notes in Computer Science*, pages 522–539, Darmstadt, Germany, May 21–23, 2012. Springer, Heidelberg, Germany. (Cited on page 18, 19, 20, 31, 67, 69.)

[BHY09] Mihir Bellare, Dennis Hofheinz, and Scott Yilek. Possibility and impossibility results for encryption and commitment secure under selective opening. In Antoine Joux, editor, *Advances in Cryptology – EUROCRYPT 2009*, volume 5479 of *Lecture Notes in Computer Science*, pages 1–35, Cologne, Germany, April 26–30, 2009. Springer, Heidelberg, Germany. (Cited on page 18, 20, 21, 31, 85.)

[Bla06] John Black. The ideal-cipher model, revisited: An uninstantiable blockcipher-based hash function. In Matthew J. B. Robshaw, editor, *Fast Software Encryption – FSE 2006*, volume 4047 of *Lecture Notes in Computer Science*, pages 328–340, Graz, Austria, March 15–17, 2006. Springer, Heidelberg, Germany. (Cited on page 15.)

[Ble98] Daniel Bleichenbacher. Chosen ciphertext attacks against protocols based on the RSA encryption standard PKCS #1. In *Advances in Cryptology - CRYPTO '98, 18th Annual International Cryptology Conference, Santa Barbara, California, USA, August 23-27, 1998, Proceedings*, pages 1–12, 1998. (Cited on page 14.)

[BLN16] Daniel J. Bernstein, Tanja Lange, and Ruben Niederhagen. Dual EC: A standardized back door. In *The New Codebreakers - Essays Dedicated to David Kahn on the Occasion of His 85th Birthday*, pages 256–281, 2016. (Cited on page 16.)

[BN00] Mihir Bellare and Chanathip Namprempre. Authenticated encryption: Relations among notions and analysis of the generic composition paradigm. In Tatsuaki Okamoto, editor, *Advances in Cryptology – ASIACRYPT 2000*, volume 1976 of *Lecture Notes in Computer Science*, pages 531–545, Kyoto, Japan, December 3–7, 2000. Springer, Heidelberg, Germany. (Cited on page 104, 105, 112.)

[BR93] Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *CCS '93, Proceedings of the 1st ACM Conference on Computer and Communications Security, Fairfax, Virginia, USA, November 3-5, 1993.*, pages 62–73, 1993. (Cited on page 14.)

[BR95] Mihir Bellare and Phillip Rogaway. Optimal asymmetric encryption. In Alfredo De Santis, editor, *Advances in Cryptology – EUROCRYPT'94*, volume 950 of *Lecture Notes in Computer Science*, pages 92–111, Perugia, Italy, May 9–12, 1995. Springer, Heidelberg, Germany. (Cited on page 14, 23, 84, 94.)

[BR97] Mihir Bellare and Phillip Rogaway. Minimizing the use of random oracles in authenticated encryption schemes. In *Information and Communication Security, First International Conference, ICICS'97, Beijing, China, November 11-14, 1997, Proceedings*, pages 1–16, 1997. (Cited on page 14, 23, 70, 94.)

[BR06] Mihir Bellare and Phillip Rogaway. The security of triple encryption and a framework for code-based game-playing proofs. In Serge Vaudenay, editor, *Advances in Cryptology – EUROCRYPT 2006*, volume 4004 of *Lecture Notes in Computer Science*, pages 409–426, St. Petersburg, Russia, May 28 – June 1, 2006. Springer, Heidelberg, Germany. (Cited on page 29.)

[Bro06] Daniel R. L. Brown. What hashes make RSA-OAEP secure? Cryptology ePrint Archive, Report 2006/223, 2006. `http://eprint.iacr.org/`. (Cited on page 85.)

[BWY11] Mihir Bellare, Brent Waters, and Scott Yilek. Identity-based encryption secure against selective opening attack. In Yuval Ishai, editor, *TCC 2011: 8th Theory of Cryptography Conference*, volume 6597 of *Lecture Notes in Computer Science*, pages 235–252, Providence, RI, USA, March 28–30, 2011. Springer, Heidelberg, Germany. (Cited on page 21.)

[BY09] Mihir Bellare and Scott Yilek. Encryption schemes secure under selective opening attack. Cryptology ePrint Archive, Report 2009/101, 2009. `http://eprint.iacr.org/2009/101`. (Cited on page 20, 22, 56.)

[CA06] T. Clancy and W. Arbaugh. Extensible Authentication Protocol (EAP) Password Authenticated Exchange. RFC 4746 (Informational), November 2006. (Cited on page 84.)

[CDNO97] Ran Canetti, Cynthia Dwork, Moni Naor, and Rafail Ostrovsky. Deniable encryption. In Burton S. Kaliski Jr., editor, *Advances in Cryptology – CRYPTO'97*, volume 1294 of *Lecture Notes in Computer Science*, pages 90–104, Santa Barbara, CA, USA, August 17–21, 1997. Springer, Heidelberg, Germany. (Cited on page 21.)

[CFGN96] Ran Canetti, Uriel Feige, Oded Goldreich, and Moni Naor. Adaptively secure multi-party computation. In *28th Annual ACM Symposium on Theory of Computing*, pages 639–648, Philadephia, PA, USA, May 22–24, 1996. ACM Press. (Cited on page 21.)

[CGH98] Ran Canetti, Oded Goldreich, and Shai Halevi. The random oracle methodology, revisited (preliminary version). In *30th Annual ACM Symposium on Theory of Computing*, pages 209–218, Dallas, TX, USA, May 23–26, 1998. ACM Press. (Cited on page 14.)

[CPS08] Jean-Sébastien Coron, Jacques Patarin, and Yannick Seurin. The random oracle model and the ideal cipher model are equivalent. In David Wagner, editor, *Advances in Cryptology – CRYPTO 2008*, volume 5157 of *Lecture Notes in Computer Science*, pages 1–20, Santa Barbara, CA, USA, August 17–21, 2008. Springer, Heidelberg, Germany. (Cited on page 15, 103, 106.)

[CS98] Ronald Cramer and Victor Shoup. A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack. In *Advances in Cryptology - CRYPTO '98, 18th Annual International Cryptology Conference, Santa Barbara, California, USA, August 23-27, 1998, Proceedings*, pages 13–25, 1998. (Cited on page 14.)

[CS02] Ronald Cramer and Victor Shoup. Universal hash proofs and a paradigm for adaptive chosen ciphertext secure public-key encryption. In Lars R. Knudsen, editor, *Advances in Cryptology – EUROCRYPT 2002*, volume 2332 of *Lecture Notes in Computer Science*, pages 45–64, Amsterdam, The Netherlands, April 28 – May 2, 2002. Springer, Heidelberg, Germany. (Cited on page 14.)

[CS03] 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(1):167–226, 2003. (Cited on page 71, 80, 106.)

[Dac14] Dana Dachman-Soled. On minimal assumptions for sender-deniable public key encryption. In Hugo Krawczyk, editor, *PKC 2014: 17th International Conference on Theory and Practice of Public Key Cryptography*, volume 8383 of *Lecture Notes in Computer Science*, pages 574–591, Buenos Aires, Argentina, March 26–28, 2014. Springer, Heidelberg, Germany. (Cited on page 21.)

[DDN91] Danny Dolev, Cynthia Dwork, and Moni Naor. Non-malleable cryptography (extended abstract). In *Proceedings of the 23rd Annual ACM Symposium on Theory of Computing, May 5-8, 1991, New Orleans, Louisiana, USA*, pages 542–552, 1991. (Cited on page 14.)

[DH76] Whitfield Diffie and Martin E. Hellman. New directions in cryptography. *IEEE Trans. Information Theory*, 22(6):644–654, 1976. (Cited on page 13.)

[DKT16] Dana Dachman-Soled, Jonathan Katz, and Aishwarya Thiruvengadam. 10-round feistel is indifferentiable from an ideal cipher. In Marc Fischlin and Jean-Sébastien Coron, editors, *Advances in Cryptology – EUROCRYPT 2016, Part II*, volume 9666 of *Lecture Notes in Computer Science*, pages 649–678, Vienna, Austria, May 8–12, 2016. Springer, Heidelberg, Germany. (Cited on page 15.)

[DNRS99] Cynthia Dwork, Moni Naor, Omer Reingold, and Larry J. Stockmeyer. Magic functions. In *40th Annual Symposium on Foundations of Computer Science*, pages 523–534, New York, NY, USA, October 17–19, 1999. IEEE Computer Society Press. (Cited on page 18, 20, 21, 22, 56.)

[DR08] T. Dierks and E. Rescorla. The Transport Layer Security (TLS) Protocol Version 1.2. RFC 5246 (Proposed Standard), August 2008. Updated by RFCs 5746, 5878, 6176. (Cited on page 84.)

[DS16] Yuanxi Dai and John P. Steinberger. Indifferentiability of 8-round feistel networks. In Matthew Robshaw and Jonathan Katz, editors, *Advances in Cryptology – CRYPTO 2016, Part I*, volume 9814 of *Lecture Notes in Computer Science*, pages 95–120, Santa Barbara, CA, USA, August 14–18, 2016. Springer, Heidelberg, Germany. (Cited on page 15.)

[Dwo01] Morris J. Dworkin. SP 800-38A: Recommendation for block cipher modes of operation: Methods and techniques. Technical report, National Institute of Standards & Technology, Gaithersburg, MD, United States, 2001. `http://csrc.nist.gov/publications/nistpubs/800-38a/sp800-38a.pdf`. (Cited on page 113.)

[Dwo07a] Morris J. Dworkin. SP 800-38C: Recommendation for block cipher modes of operation: The CCM mode for authentication and confidentiality. Technical report, National Institute of Standards & Technology, Gaithersburg, MD, United States, 2007. http://csrc.nist.gov/publications/nistpubs/800-38C/SP800-38C_updated-July20_2007.pdf. (Cited on page 113, 119.)

[Dwo07b] Morris J. Dworkin. SP 800-38D: Recommendation for block cipher modes of operation: Galois/Counter Mode (GCM) and GMAC. Technical report, National Institute of Standards & Technology, Gaithersburg, MD, United States, 2007. http://csrc.nist.gov/publications/nistpubs/800-38D/SP-800-38D.pdf. (Cited on page 113.)

[Dwo10] Morris J. Dworkin. Addendum to SP 800-38A: Recommendation for block cipher modes of operation: Three variants of ciphertext stealing for CBC mode. Technical report, National Institute of Standards & Technology, Gaithersburg, MD, United States, 2010. http://csrc.nist.gov/publications/nistpubs/800-38a/addendum-to-nist_sp800-38A.pdf. (Cited on page 113, 116.)

[Dwo15] Morris J. Dworkin. FIPS 202: Sha-3 standard: Permutation-based hash and extendable-output functions. Technical report, National Institute of Standards & Technology, Gaithersburg, MD, United States, 2015. http://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.202.pdf. (Cited on page 103.)

[ElG84] Taher ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. In G. R. Blakley and David Chaum, editors, *Advances in Cryptology – CRYPTO'84*, volume 196 of *Lecture Notes in Computer Science*, pages 10–18, Santa Barbara, CA, USA, August 19–23, 1984. Springer, Heidelberg, Germany. (Cited on page 14.)

[EM93] Shimon Even and Yishay Mansour. A construction of a cipher from a single pseudorandom permutation. In Hideki Imai, Ronald L. Rivest, and Tsutomu Matsumoto, editors, *Advances in Cryptology – ASIACRYPT'91*, volume 739 of *Lecture Notes in Computer Science*, pages 210–224, Fujiyoshida, Japan, November 11–14, 1993. Springer, Heidelberg, Germany. (Cited on page 106.)

[FHKP16] Georg Fuchsbauer, Felix Heuer, Eike Kiltz, and Krzysztof Pietrzak. Standard security does imply security against selective opening for markov distributions. In Eyal Kushilevitz and Tal Malkin, editors, *TCC 2016-A: 13th Theory of Cryptography Conference, Part I*, volume 9562 of *Lecture Notes in Computer Science*, pages 282–305, Tel Aviv, Israel, January 10–13, 2016. Springer, Heidelberg, Germany. (Cited on page 23.)

[FHKW10] Serge Fehr, Dennis Hofheinz, Eike Kiltz, and Hoeteck Wee. Encryption schemes secure against chosen-ciphertext selective opening attacks. In Henri Gilbert, editor, *Advances in Cryptology – EUROCRYPT 2010*, volume 6110 of *Lecture Notes in Computer Science*, pages 381–402, French Riviera, May 30 – June 3, 2010. Springer, Heidelberg, Germany. (Cited on page 19, 21, 24, 67.)

[FO99] Eiichiro Fujisaki and Tatsuaki Okamoto. Secure integration of asymmetric and symmetric encryption schemes. In Michael J. Wiener, editor, *Advances in Cryptology – CRYPTO'99*, volume 1666 of *Lecture Notes in Computer Science*, pages 537–554, Santa Barbara, CA, USA, August 15–19, 1999. Springer, Heidelberg, Germany. (Cited on page 14, 94, 95, 96.)

[FO13] Eiichiro Fujisaki and Tatsuaki Okamoto. Secure integration of asymmetric and symmetric encryption schemes. *Journal of Cryptology*, 26(1):80–101, January 2013. (Cited on page 95, 96.)

[FOPS01] Eiichiro Fujisaki, Tatsuaki Okamoto, David Pointcheval, and Jacques Stern. RSA-OAEP is secure under the RSA assumption. In Joe Kilian, editor, *Advances in Cryptology – CRYPTO 2001*, volume 2139 of *Lecture Notes in Computer Science*, pages 260–274, Santa Barbara, CA, USA, August 19–23, 2001. Springer, Heidelberg, Germany. (Cited on page 84, 93.)

[FS87] Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In Andrew M. Odlyzko, editor, *Advances in Cryptology – CRYPTO'86*, volume 263 of *Lecture Notes in Computer Science*, pages 186–194, Santa Barbara, CA, USA, August 1987. Springer, Heidelberg, Germany. (Cited on page 14.)

[Gam84] Taher El Gamal. A public key cryptosystem and a signature scheme based on discrete logarithms. In *Advances in Cryptology, Proceedings of CRYPTO '84, Santa Barbara, California, USA, August 19-22, 1984, Proceedings*, pages 10–18, 1984. (Cited on page 13, 70.)

[GL89] Oded Goldreich and Leonid A. Levin. A hard-core predicate for all one-way functions. In *Proceedings of the 21st Annual ACM Symposium on Theory* *of Computing, May 14-17, 1989, Seattle, Washigton, USA*, pages 25–32, 1989. (Cited on page 13.)

[GM82] Shafi Goldwasser and Silvio Micali. Probabilistic encryption and how to play mental poker keeping secret all partial information. In *Proceedings of the 14th Annual ACM Symposium on Theory of Computing, May 5-7, 1982, San Francisco, California, USA*, pages 365–377, 1982. (Cited on page 13, 21.)

[GM84] Shafi Goldwasser and Silvio Micali. Probabilistic encryption. *J. Comput. Syst. Sci.*, 28(2):270–299, 1984. (Cited on page 13.)

[GP99] Louis Goubin and Jacques Patarin. DES and differential power analysis (the "duplication" method). In *Cryptographic Hardware and Embedded Systems, First International Workshop, CHES'99, Worcester, MA, USA, August 12-13, 1999, Proceedings*, pages 158–172, 1999. (Cited on page 15.)

[GST14] Daniel Genkin, Adi Shamir, and Eran Tromer. RSA key extraction via low-bandwidth acoustic cryptanalysis. In Juan A. Garay and Rosario Gennaro, editors, *Advances in Cryptology – CRYPTO 2014, Part I*, volume 8616 of *Lecture Notes in Computer Science*, pages 444–461, Santa Barbara, CA, USA, August 17–21, 2014. Springer, Heidelberg, Germany. (Cited on page 15.)

[Har06] B. Harris. RSA Key Exchange for the Secure Shell (SSH) Transport Layer Protocol. RFC 4432 (Proposed Standard), March 2006. (Cited on page 84.)

[HHK10] Javier Herranz, Dennis Hofheinz, and Eike Kiltz. Some (in)sufficient conditions for secure hybrid encryption. *Inf. Comput.*, 208(11):1243–1257, 2010. (Cited on page 73.)

[HJKS15] Felix Heuer, Tibor Jager, Eike Kiltz, and Sven Schäge. On the selective opening security of practical public-key encryption schemes. In Jonathan Katz, editor, *PKC 2015: 18th International Conference on Theory and Practice of Public Key Cryptography*, volume 9020 of *Lecture Notes in Computer Science*, pages 27–51, Gaithersburg, MD, USA, March 30 – April 1, 2015. Springer, Heidelberg, Germany. (Cited on page 23, 69.)

[HJR16] Dennis Hofheinz, Tibor Jager, and Andy Rupp. Public-key encryption with simulation-based selective-opening security and compact ciphertexts. In Martin Hirt and Adam D. Smith, editors, *TCC 2016-B: 14th Theory* *of Cryptography Conference, Part II*, volume 9986 of *Lecture Notes in Computer Science*, pages 146–168, Beijing, China, October 31 – November 3, 2016. Springer, Heidelberg, Germany. (Cited on page 21.)

[HJSK16] Felix Heuer, Tibor Jager, Sven Schäge, and Eike Kiltz. Selective opening security of practical public-key encryption schemes. *IET Information Security*, 10(6):304–318, 2016. (Cited on page 23.)

[HK07] Dennis Hofheinz and Eike Kiltz. Secure hybrid encryption from weakened key encapsulation. In Alfred Menezes, editor, *Advances in Cryptology – CRYPTO 2007*, volume 4622 of *Lecture Notes in Computer Science*, pages 553–571, Santa Barbara, CA, USA, August 19–23, 2007. Springer, Heidelberg, Germany. (Cited on page 112.)

[HLMC15] Zhengan Huang, Shengli Liu, Xianping Mao, and Kefei Chen. Non-malleability under selective opening attacks: Implication and separation. In Tal Malkin, Vladimir Kolesnikov, Allison Bishop Lewko, and Michalis Polychronakis, editors, *ACNS 15: 13th International Conference on Applied Cryptography and Network Security*, volume 9092 of *Lecture Notes in Computer Science*, pages 87–104, New York, NY, USA, June 2–5, 2015. Springer, Heidelberg, Germany. (Cited on page 19.)

[HLOV11] Brett Hemenway, Benoît Libert, Rafail Ostrovsky, and Damien Vergnaud. Lossy encryption: Constructions from general assumptions and efficient selective opening chosen ciphertext security. In Dong Hoon Lee and Xiaoyun Wang, editors, *Advances in Cryptology – ASIACRYPT 2011*, volume 7073 of *Lecture Notes in Computer Science*, pages 70–88, Seoul, South Korea, December 4–8, 2011. Springer, Heidelberg, Germany. (Cited on page 21.)

[Hof12] Dennis Hofheinz. All-but-many lossy trapdoor functions. In David Pointcheval and Thomas Johansson, editors, *Advances in Cryptology – EUROCRYPT 2012*, volume 7237 of *Lecture Notes in Computer Science*, pages 209–227, Cambridge, UK, April 15–19, 2012. Springer, Heidelberg, Germany. (Cited on page 19, 21.)

[HOR15] Brett Hemenway, Rafail Ostrovsky, and Alon Rosen. Non-committing encryption from ϕ-hiding. In Yevgeniy Dodis and Jesper Buus Nielsen, editors, *TCC 2015: 12th Theory of Cryptography Conference, Part I*, volume 9014 of *Lecture Notes in Computer Science*, pages 591–608, Warsaw, Poland, March 23–25, 2015. Springer, Heidelberg, Germany. (Cited on page 21.)

[Hou03] R. Housley. Use of the RSAES-OAEP Key Transport Algorithm in Cryptographic Message Syntax (CMS). RFC 3560 (Proposed Standard), July 2003. (Cited on page 84.)

[HP16] Felix Heuer and Bertram Poettering. Selective opening security from simulatable data encapsulation. In Jung Hee Cheon and Tsuyoshi Takagi, editors, *Advances in Cryptology – ASIACRYPT 2016, Part II*, volume 10032 of *Lecture Notes in Computer Science*, pages 248–277, Hanoi, Vietnam, December 4–8, 2016. Springer, Heidelberg, Germany. (Cited on page 24.)

[HPW15] Carmit Hazay, Arpita Patra, and Bogdan Warinschi. Selective opening security for receivers. In Tetsu Iwata and Jung Hee Cheon, editors, *Advances in Cryptology – ASIACRYPT 2015, Part I*, volume 9452 of *Lecture Notes in Computer Science*, pages 443–469, Auckland, New Zealand, November 30 – December 3, 2015. Springer, Heidelberg, Germany. (Cited on page 20, 21.)

[HR14] Dennis Hofheinz and Andy Rupp. Standard versus selective opening security: Separation and equivalence results. In Yehuda Lindell, editor, *TCC 2014: 11th Theory of Cryptography Conference*, volume 8349 of *Lecture Notes in Computer Science*, pages 591–615, San Diego, CA, USA, February 24–26, 2014. Springer, Heidelberg, Germany. (Cited on page 19, 20, 21, 22.)

[HRW16] Dennis Hofheinz, Vanishree Rao, and Daniel Wichs. Standard security does not imply indistinguishability under selective opening. In Martin Hirt and Adam D. Smith, editors, *TCC 2016-B: 14th Theory of Cryptography Conference, Part II*, volume 9986 of *Lecture Notes in Computer Science*, pages 121–145, Beijing, China, October 31 – November 3, 2016. Springer, Heidelberg, Germany. (Cited on page 20, 21, 27, 133.)

[HSH$^+$08] J. Alex Halderman, Seth D. Schoen, Nadia Heninger, William Clarkson, William Paul, Joseph A. Calandrino, Ariel J. Feldman, Jacob Appelbaum, and Edward W. Felten. Lest we remember: Cold boot attacks on encryption keys. In *Proceedings of the 17th USENIX Security Symposium, July 28-August 1, 2008, San Jose, CA, USA*, pages 45–60, 2008. (Cited on page 15.)

[HV17] Felix Heuer and Jorge Villar. Private communication with the author, March 2017. (Cited on page 53.)

[JK03] J. Jonsson and B. Kaliski. Public-Key Cryptography Standards (PKCS) #1: RSA Cryptography Specifications Version 2.1. RFC 3447 (Informational), February 2003. (Cited on page 93.)

[JQY01] Marc Joye, Jean-Jacques Quisquater, and Moti Yung. On the power of misbehaving adversaries and security analysis of the original EPOC. In David Naccache, editor, *Topics in Cryptology – CT-RSA 2001*, volume 2020 of *Lecture Notes in Computer Science*, pages 208–222, San Francisco, CA, USA, April 8–12, 2001. Springer, Heidelberg, Germany. (Cited on page 95.)

[KJJ99] Paul C. Kocher, Joshua Jaffe, and Benjamin Jun. Differential power analysis. In Michael J. Wiener, editor, *Advances in Cryptology – CRYPTO'99*, volume 1666 of *Lecture Notes in Computer Science*, pages 388–397, Santa Barbara, CA, USA, August 15–19, 1999. Springer, Heidelberg, Germany. (Cited on page 15.)

[KL07] Jonathan Katz and Yehuda Lindell. *Introduction to Modern Cryptography (Chapman & Hall/Crc Cryptography and Network Security Series)*. Chapman & Hall/CRC, 2007. (Cited on page 30.)

[Koc96a] Paul C. Kocher. Timing attacks on implementations of diffie-hellman, rsa, dss, and other systems. In *Advances in Cryptology - CRYPTO '96, 16th Annual International Cryptology Conference, Santa Barbara, California, USA, August 18-22, 1996, Proceedings*, pages 104–113, 1996. (Cited on page 15.)

[Koc96b] Paul C. Kocher. Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems. In Neal Koblitz, editor, *Advances in Cryptology – CRYPTO'96*, volume 1109 of *Lecture Notes in Computer Science*, pages 104–113, Santa Barbara, CA, USA, August 18–22, 1996. Springer, Heidelberg, Germany. (Cited on page 15.)

[KOS10] Eike Kiltz, Adam O'Neill, and Adam Smith. Instantiability of RSA-OAEP under chosen-plaintext attack. In Tal Rabin, editor, *Advances in Cryptology – CRYPTO 2010*, volume 6223 of *Lecture Notes in Computer Science*, pages 295–313, Santa Barbara, CA, USA, August 15–19, 2010. Springer, Heidelberg, Germany. (Cited on page 85.)

[KP09] Eike Kiltz and Krzysztof Pietrzak. On the security of padding-based encryption schemes - or - why we cannot prove OAEP secure in the standard model. In Antoine Joux, editor, *Advances in Cryptology – EUROCRYPT 2009*, volume 5479 of *Lecture Notes in Computer Science*, pages 389–406, Cologne, Germany, April 26–30, 2009. Springer, Heidelberg, Germany. (Cited on page 14, 85.)

[KR01] Joe Kilian and Phillip Rogaway. How to protect DES against exhaustive key search (an analysis of DESX). *Journal of Cryptology*, 14(1):17–35, 2001. (Cited on page 106.)

[LDL+14] Junzuo Lai, Robert H. Deng, Shengli Liu, Jian Weng, and Yunlei Zhao. Identity-based encryption secure against selective opening chosen-ciphertext attack. In Phong Q. Nguyen and Elisabeth Oswald, editors, *Advances in Cryptology – EUROCRYPT 2014*, volume 8441 of *Lecture Notes in Computer Science*, pages 77–92, Copenhagen, Denmark, May 11–15, 2014. Springer, Heidelberg, Germany. (Cited on page 21.)

[LP15] Shengli Liu and Kenneth G. Paterson. Simulation-based selective opening CCA security for PKE from key encapsulation mechanisms. In Jonathan Katz, editor, *PKC 2015: 18th International Conference on Theory and Practice of Public Key Cryptography*, volume 9020 of *Lecture Notes in Computer Science*, pages 3–26, Gaithersburg, MD, USA, March 30 – April 1, 2015. Springer, Heidelberg, Germany. (Cited on page 21, 24, 69, 103.)

[Mer78] Ralph C. Merkle. Secure communications over insecure channels. *Commun. ACM*, 21(4):294–299, 1978. (Cited on page 13.)

[MPL+11] Amir Moradi, Axel Poschmann, San Ling, Christof Paar, and Huaxiong Wang. Pushing the limits: A very compact and a threshold implementation of AES. In Kenneth G. Paterson, editor, *Advances in Cryptology – EUROCRYPT 2011*, volume 6632 of *Lecture Notes in Computer Science*, pages 69–88, Tallinn, Estonia, May 15–19, 2011. Springer, Heidelberg, Germany. (Cited on page 15.)

[MSZ16] Eric Miles, Amit Sahai, and Mark Zhandry. Annihilation attacks for multilinear maps: Cryptanalysis of indistinguishability obfuscation over GGH13. In Matthew Robshaw and Jonathan Katz, editors, *Advances in Cryptology – CRYPTO 2016, Part II*, volume 9815 of *Lecture Notes in Computer Science*, pages 629–658, Santa Barbara, CA, USA, August 14–18, 2016. Springer, Heidelberg, Germany. (Cited on page 20.)

[Mun] Randall Munroe. xkcd, A webcomic of romance, sarcasm, math, and language. Remixed under creative commons licence CC BY-NC 2.5. (Cited on page 17.)

[NP00] Pierpaolo Natalini and Biagio Palumbo. Inequalities for the incomplete gamma function. *Mathematical Inequalities and Applications*, 3:69–77, 2000. (Cited on page 38.)

[NRR06] Svetla Nikova, Christian Rechberger, and Vincent Rijmen. Threshold implementations against side-channel attacks and glitches. In *Information and Communications Security, 8th International Conference, ICICS 2006, Raleigh, NC, USA, December 4-7, 2006, Proceedings*, pages 529–545, 2006. (Cited on page 15.)

[NS09] Moni Naor and Gil Segev. Public-key cryptosystems resilient to key leakage. In *Advances in Cryptology - CRYPTO 2009, 29th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2009. Proceedings*, pages 18–35, 2009. (Cited on page 15.)

[NSF05] T. Nadeau, C. Srinivasan, and A. Farrel. Multiprotocol Label Switching (MPLS) Management Overview. RFC 4221 (Informational), November 2005. (Cited on page 84.)

[NY90] Moni Naor and Moti Yung. Public-key cryptosystems provably secure against chosen ciphertext attacks. In *Proceedings of the 22nd Annual ACM Symposium on Theory of Computing, May 13-17, 1990, Baltimore, Maryland, USA*, pages 427–437, 1990. (Cited on page 14.)

[OP01] Tatsuaki Okamoto and David Pointcheval. REACT: Rapid Enhanced-security Asymmetric Cryptosystem Transform. In David Naccache, editor, *Topics in Cryptology – CT-RSA 2001*, volume 2020 of *Lecture Notes in Computer Science*, pages 159–175, San Francisco, CA, USA, April 8–12, 2001. Springer, Heidelberg, Germany. (Cited on page 71, 94.)

[Pei14] Chris Peikert. Lattice cryptography for the internet. In Michele Mosca, editor, *Post-Quantum Cryptography - 6th International Workshop, PQCrypto 2014, Waterloo, ON, Canada, October 1-3, 2014. Proceedings*, volume 8772 of *Lecture Notes in Computer Science*, pages 197–219. Springer, 2014. (Cited on page 94.)

[PW08] Chris Peikert and Brent Waters. Lossy trapdoor functions and their applications. In Richard E. Ladner and Cynthia Dwork, editors, *40th Annual ACM Symposium on Theory of Computing*, pages 187–196, Victoria, British Columbia, Canada, May 17–20, 2008. ACM Press. (Cited on page 20, 21, 85.)

[Rae05] K. Raeburn. Encryption and Checksum Specifications for Kerberos 5. RFC 3961 (Proposed Standard), February 2005. (Cited on page 84.)

[Res02] E. Rescorla. Preventing the Million Message Attack on Cryptographic Message Syntax. RFC 3218 (Informational), January 2002. (Cited on page 84.)

[Rog02] Phillip Rogaway. Authenticated-encryption with associated-data. In Vijayalakshmi Atluri, editor, *ACM CCS 02: 9th Conference on Computer and Communications Security*, pages 98–107, Washington D.C., USA, November 18–22, 2002. ACM Press. (Cited on page 113.)

[RS92] Charles Rackoff and Daniel R. Simon. Non-interactive zero-knowledge proof of knowledge and chosen ciphertext attack. In Joan Feigenbaum, editor, *Advances in Cryptology – CRYPTO'91*, volume 576 of *Lecture Notes in Computer Science*, pages 433–444, Santa Barbara, CA, USA, August 11–15, 1992. Springer, Heidelberg, Germany. (Cited on page 14.)

[RSA78] Ronald L. Rivest, Adi Shamir, and Leonard M. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Commun. ACM*, 21(2):120–126, 1978. (Cited on page 13.)

[RT10] B. Ramsdell and S. Turner. Secure/Multipurpose Internet Mail Extensions (S/MIME) Version 3.2 Message Specification. RFC 5751 (Proposed Standard), January 2010. (Cited on page 84.)

[SBZ02] Ron Steinfeld, Joonsang Baek, and Yuliang Zheng. On the necessity of strong assumptions for the security of a class of asymmetric encryption schemes. In Lynn Margaret Batten and Jennifer Seberry, editors, *ACISP 02: 7th Australasian Conference on Information Security and Privacy*, volume 2384 of *Lecture Notes in Computer Science*, pages 241–256, Melbourne, Victoria, Australia, July 3–5, 2002. Springer, Heidelberg, Germany. (Cited on page 14, 70, 72.)

[Sha49] Claude E. Shannon. Communication theory of secrecy systems. *Bell Systems Technical Journal*, 28(4):656–715, 1949. (Cited on page 15.)

[Sho97] Victor Shoup. Lower bounds for discrete logarithms and related problems. In Walter Fumy, editor, *Advances in Cryptology – EUROCRYPT'97*, volume 1233 of *Lecture Notes in Computer Science*, pages 256–266, Konstanz, Germany, May 11–15, 1997. Springer, Heidelberg, Germany. (Cited on page 20.)

[Sho02] Victor Shoup. OAEP reconsidered. *Journal of Cryptology*, 15(4):223–249, 2002. (Cited on page 14, 84.)

[Sho04a] Victor Shoup. ISO 18033-2: An emerging standard for public-key encryption. http://shoup.net/iso/std6.pdf, December 2004. Final Committee Draft. (Cited on page 83.)

[Sho04b] Victor Shoup. Sequences of games: a tool for taming complexity in security proofs, 2004. shoup@cs.nyu.edu 13166 received 30 Nov 2004, last revised 18 Jan 2006. (Cited on page 29.)

[Sho04c] Victor Shoup. Sequences of games: a tool for taming complexity in security proofs. *IACR Cryptology ePrint Archive*, 2004:332, 2004. (Cited on page 77.)

[ST02] Kouichi Sakurai and Tsuyoshi Takagi. A reject timing attackon an IND-CCA2 public-key cryptosystem. In Pil Joong Lee and Chae Hoon Lim, editors, *Information Security and Cryptology - ICISC 2002, 5th International Conference Seoul, Korea, November 28-29, 2002, Revised Papers*, volume 2587 of *Lecture Notes in Computer Science*, pages 359–373. Springer, 2002. (Cited on page 95.)

[Sta13] SPIEGEL Staff. Documents reveal top nsa hacking unit. http://www.spiegel.de/international/world/the-nsa-uses-powerful-toolbox-in-effort-to-spy-on-global-networks-a-940969.html, December 2013. last retrieved 17.04.2017. (Cited on page 16.)

[Vil15] Jorge Villar. Private communication with the author, July 2015. (Cited on page 53.)

[WC81] Mark N. Wegman and Larry Carter. New hash functions and their use in authentication and set equality. *J. Comput. Syst. Sci.*, 22(3):265–279, 1981. (Cited on page 82, 121.)

[Yil10] Scott Yilek. *Public-Key Encryption Secure in the Presence of Randomness Failures*. PhD thesis, University of California, San Diego, 2010. (Cited on page 18.)

Curriculum Vitae

PERSONAL DATA

Full name:	Felix Lars Heuer
Email:	felix.heuer@rub.de

EDUCATION

1998 – 2007	Abitur (EQF Level 4) Geschwister-Scholl-Gymnasium, Wetter
Winter 08/09 – Summer 2011	BSc. Mathematics Ruhr University Bochum, Germany
Summer 2011 – 01/2014	MSc. Mathematics Ruhr University Bochum, Germany
01/2014 – 06/2017	PhD Mathematics Chair for Cryptography Ruhr University Bochum, Germany
since 07/2017	PostDoc Chair for Cryptology and IT security Ruhr University Bochum, Germany

Publications

[FHKP16] Georg Fuchsbauer, Felix Heuer, Eike Kiltz, and Krzysztof Pietrzak. Standard security does imply security against selective opening for markov distributions. In Eyal Kushilevitz and Tal Malkin, editors, *TCC 2016-A: 13th Theory of Cryptography Conference, Part I*, volume 9562 of *Lecture Notes in Computer Science*, pages 282–305, Tel Aviv, Israel, January 10–13, 2016. Springer, Heidelberg, Germany.

[HJKS15] Felix Heuer, Tibor Jager, Eike Kiltz, and Sven Schäge. On the selective opening security of practical public-key encryption schemes. In Jonathan Katz, editor, *PKC 2015: 18th International Conference on Theory and Practice of Public Key Cryptography*, volume 9020 of *Lecture Notes in Computer Science*, pages 27–51, Gaithersburg, MD, USA, March 30 – April 1, 2015. Springer, Heidelberg, Germany.

[HJSK16] Felix Heuer, Tibor Jager, Sven Schäge, and Eike Kiltz. Selective opening security of practical public-key encryption schemes. *IET Information Security*, 10(6):304–318, 2016.

[HP16] Felix Heuer and Bertram Poettering. Selective opening security from simulatable data encapsulation. In Jung Hee Cheon and Tsuyoshi Takagi, editors, *Advances in Cryptology – ASIACRYPT 2016, Part II*, volume 10032 of *Lecture Notes in Computer Science*, pages 248–277, Hanoi, Vietnam, December 4–8, 2016. Springer, Heidelberg, Germany.