Try again. Fail again. Fail better : new notions of security, broken assumptions, and increased efficiency in cryptography

Aisling Connolly

PSL
UNIVERSITÉ PARIS

THÈSE DE DOCTORAT
DE L'UNIVERSITÉ PSL

Préparée à École normale supérieure

Try Again. Fail Again. Fail Better.
New Notions of Security, Broken Assumptions, and Increased Efficiency in Cryptography.

Soutenue par
Aisling Connolly
Le 13 septembre 2019

École doctorale n°386
Sciences Mathématiques de Paris Centre

Spécialité
Informatique

Composition du jury :

Whitfield Diffie Consulting Professor Emeritus, Stanford	*Président du Jury*
Bart Preneel Professor, KU Leuven	*Rapporteur*
Moti Yung Professor, Columbia & Google	*Rapporteur*
Michel Abdalla Professor, École normale supérieure	*Examinateur*
Manuel Barbosa Assistant Professor, University of Porto	*Examinateur*
Delaram Kahrobaei Professor, University of York	*Examinatrice*
Christian Rechberger Professor, TU Graz	*Examinateur*
Guénolé Silvestre Associate Professor, UCD Dublin	*Invité*
David Naccache Professor, École normale supérieure	*Directeur de thèse*

1794
ENS | PSL

TRY AGAIN. FAIL AGAIN. FAIL BETTER.

AISLING CONNOLLY

New Notions of Security, Broken Assumptions, and Increased Efficiency in Cryptography.

September 2019 – PhD Thesis (Defence Version)

ABSTRACT

This thesis presents new results addressing three fundamental areas of cryptography: *security* notions, *assumptions*, and *efficiency*.

The first part encompasses the security of symmetric primitives. We give a new security notion that provides the strongest security for symmetric primitives proven in the random oracle model (ROM). Key-correlated attacks (KCA) model the scenario where all inputs (keys, messages, and possibly nonces and headers) are correlated with the secret key. Under mild assumptions, we prove KCA security of blockciphers, and show that 3-rounds of Even-Mansour are necessary to achieve this. Then, we define a KCA-security notion for nonce-based authenticated encryption (AE), and provide a black-box transformation that turns a multiuser-secure AE into an AE scheme that is provably KCA secure in the ROM. We show relations and separations with older notions (related-key and key-dependent message security) to show that security under KCA is strictly stronger, and implies the others.

The next part turns to public-key cryptography, and analyses the assumptions underlying the new public-key cryptosystem of AJPS. Cryptanalysis of their assumption, based on arithmetic modulo a Mersenne prime, allowed us to reconstruct the secret key, given only the public key. Based on a modified version of the assumption, we propose a variant public-key cryptosystem secure against known quantum attacks.

The last part turns to efficiency, and studies the Schnorr Signature scheme. Exploiting the group structure we leverage the nonce-material to generate a batch of signatures. This, together with some preprocessing tricks, allow us to increase the efficiency of Schnorr signature generation. Security is maintained under a new assumption of intractability, provable in the generic group model.

RESUMÉ

Cette thèse présente des résultats nouveaux portant sur trois domaines fondamentaux de la cryptographie : les propriétés de sécurité, les hypothèses cryptographiques, et l'efficacité algorithmique.

La première partie s'intéresse à la sécurité des primitives symétriques. Nous introduisons une nouvelle propriété de sécurité correspondant à la plus forte sécurité pour les primitives symétriques prouvées sûres dans le modèle de l'oracle aléatoire. Les attaques par clé corrélées capturent les scénarios dans lesquels toutes les entrées (clés, messages, et éventuellement nonces et en-têtes) sont corrélées avec avec la clé secrète. Sous des hypothèses relativement faibles nous prouvons la sécurité contre les attaques par clé corrélées pour les algorithmes de chiffrement par bloc, et montrons que trois tours d'Even-Mansour sont nécessaires pour cela. Nous étendons ensuite les attaques par clés corrélées au chiffrement authentifié basé sur les nonces, et fournissons une transformation en boîte noire qui, partant d'un chiffrement authentifié à utilisateurs multiples, donne un chiffrement authentifié démontré résistant aux attaques par clés corrélés dans le modèle de l'oracle aléatoire. Nous établissons les relations et séparations avec les notions déjà existantes (sécurité contre les attaques par clés apparentées et par message dépendant de la clé) pour montrer que la sécurité contre les attaques par clé corrélés est strictement plus forte, et implique les autres.

La partie suivante porte sur la cryptographie à clé publique, et analyse les hypothèses sous-jacentes au nouveau cryptosystème introduit dans AJPS17. La cryptanalyse de cette hypothèse, reposant sur l'arithmétique modulo un premier de Mersenne, nous permet de reconstruire la clé secrète à partir de la clé publique uniquement. Nous proposons alors une variante de ce sytème à clé publique, fondée sur une modification de l'hypothèse précédente, résistant aux attaques connues (classiques et quantiques).

La dernière partie aborde l'efficacité algorithmique du schéma de signature de Schnorr. En mettant à profit la structure de groupe nous pouvons tirer parti du nonce pour produire un lot de signatures. Combinant ceci avec des méthodes de précalcul nous permet de rendre plus efficace l'algorithme de signature de Schnorr . La sécurité est préservée sous une hypothèse nouvelle, dont on montre qu'elle est vraie dans le modèle du groupe générique.

PREFACE

This thesis is not perfect, indeed far from it, yet I like it very much. I have little doubt that inconsistencies reside in every page, and I reserve the right to leave them there. I have, however, tried in ways to make this document somehow *nice*, a little bit different, in the hope that it becomes more pleasurable to read, and navigate, than a standard 200-page block of text.

I have been told I thought about this too much, but I disagree. Science, without communication, is a fruitless endeavour, so I feel it is as much our duty to think about, and experiment with communication as it is to uncover, or create (however you like to see it), the truths of our world.

On Motivation. If you know me, you'll know that I'm not a very 'motivated' person. I just want to learn some things, and simple things at that. *Things*, however, have become quite complicated over the course of time, and to pare them back to what could be considered simple, takes quite some effort. It would be somewhat pointless, or selfish, to learn things, and then not use them, so while going to the effort of paring things back to their simplicities, I also try to share them with other people. This thesis was concieved as a result of my *paring* and *sharing* carried out as a student of the École normale supérieure in Paris, and as a member of the Advanced Reserach team at Ingenico Group. Both places, but particularly Ingenico, have been unexpectedly and incredibly kind to me. I never dreamed I would land amongst such good company.

On Style. While trying to learn simple things, I wanted to produce a simple-looking document to describe them. Tufte has thought a lot about how to arrange information nicely on a page; his style is often used when building textbooks. As I am not writing a textbook (although I would like to at some point), I have little use for much of the functionality that comes with the style. I have instead used the Classic Thesis package which is similar in style to Tufte, yet more simple, and is based on Robert Bringhurst's Elements of Typographic Style. I have modified the package a little to make it even simpler again. There are a number of reasons why I like these styles, I have listed some of them below

wide margins. Well, you say 'wide margins', I say *short lines*. I thought often about the one-column vs. two-column style that is used when publishing papers. Different venues have their preferenes, and both syles have their advantages. One column looks very *clean* in that there is just one place for your eyes to be dragged to. Two columns are nice because the lines are

short to read which somehow helps to keep my attention longer. However, with two columns, a lot is packed into the page, which makes it look quite ugly. I thought it would be nice to have something *in-between*. One column so that the page stays tidy looking, but short enough lines so that you don't feel like you've read War and Peace before you get to move to the next line. This is what you'll find here.

MARGIN PARAGRAPHS. In using 'short lines' there is a lot of space left in the margins. The tufte style has even wider margins than you will find here, and often people use this space to put little diagrams, or notes that summarise the adjacent paragraph, or references to more explanatory text. I like the idea of using the margin space, but putting too much text or image in the margin runs the risk of falling into a two-column cluttered-looking page problem. Here, the margins have been kept wide enough so as to include *some* text, but not so wide that it could be considered a second column.

More often than not, they probably resemble more snide *remarks.*

The content you will see in the margin paragraphs is therefore limited in scope to less than you will find in tufte-style books; no diagrams, no rambling explainations. I intended to write summaries or a sort of tl;dr of the paragraph, and some side remarks. Hopefully you will find them useful, and where the content gets particularly involved, e.g., in the proofs, I try to highlight what the key point is, or where the 'trick' is, or at least to indicate "this is the only part you really need to read." Now I have gotten used to doing this, it hurts that there's nowhere to put these pointers in regular scientific papers.

TABLE OF CONTENTS. I have always wondered why, in a table of contents, the section titles are flushed left, and the page numbers flushed right. Is it not difficult to trace across from left to right to find the corresponding page? It is not so bad on printed paper, but on a screen, I just find it unnecessarily complicated. People sometimes put a row of dots between the section title and the page numbers, to assist in the left-right tracing, but with a table of these rows the page just looks like it's about to start explaining lattice-based crypto to me. Here you will find the page numbers beside the section titles. No right-flushing, no dots, no tracing.

ON CAPITALISATION. I have a terrible habit of capitalising words I shouldn't, and not capitalising words I should. I don't know if this is perhaps due to a very good German teacher, Berit, I had at one point, who drilled into us correct capitalisation rules in German, so much that it ruined my English, or if it is related to the fact that I always mix up left and right. In English, something with a capital letter is often of greater importance to something without, so it can

cause offence to (mis-)place capital letters. In any case, please don't take it personally if I diminuate you by not capitalising your name, or if I canonise someone by referring to 'His', 'Her', or 'Their' work.

On Grammar. When I am not making capitalisation mistakes, I have, tried at least, to use nice grammar. I would like to say that I have included elements from the Chicago Manual of Style. In reality, this book is too big for me, but I have at least read the small Strunk and White book on The Elements of Style. I am also, as you will come to learn, quite fond of using Oxford commas.

On English. I come from a country that happened to be colonised by the British Empire for a bit too long. We picked up some of their habits, and have not managed to shake them, so the language used here is (mostly) British English. Many academics seem perturbed by the use of British (as opposed to American) English in scientific writing, I imagine because it gives rise to inconsistencies in spelling when different authors collaborate. Details are important when they may give rise to ambiguity, particularly in the sciences, but I feel that writing *color* or *colour* will not lead to too much confusion, so I don't care to correct it. I do understand that some things in British English can have more meanings than most people would like, and in these instances I assure you I will not write *flat*, or *lift*, and instead write 'appartment' and 'elevator'. In any case, you should count yourself lucky that I didn't write in Irish English!

Although, if you're wide to this gill's g that'd be pure tome.

On Beckett. You may have noticed I have included a number of references to Samuel Beckett and his work. People have asked what is the 'Beckett obsession', and I would like to clarify. First, it is not an obsession, but rather just a few references to something that nicely ties together some memories and principles I care about from this place and time. Let me elaborate, and then you can read the thesis, I promise.

the title "*Ever tried. Ever failed. No matter. Try again. Fail again. Fail better.*" Perhaps his most famous line, I had to include it somewhere. I like it so much, I put it as the title of this thesis. I like it so much still that I name the three main technical parts of this thesis by those last three short sentences.

the quotes Many people include quotes in their PhD theses. Perhaps they wish to show some personality, or to inspire. I tried to think of quotes that I would like to include, but I felt this is not to place to share the sentences I would really like to share. Those ones, I would rather tell you over a coffee. So instead, all quotes stem only from Beckett, first, because he's a great writer and you should read more literature, second, because in truth, I do find many of them inspirational, and third, simply because it matched the theme.

I feel safe in saying that you should read more literature, as everyone should.

THE ROOM I organised my PhD defence to take place in Salle Beckett. It is one of the smallest rooms at ENS and so is quite inappropriate for a defence. Additionally, it is one of the main teaching rooms (of which there are very few), so it's constantly booked during term time. Furthermore, it doesn't *belong* to the DI, we have our own room for defences, so why care so much about Salle Beckett?

As Ireland was once colonised by the British Empire, it has now become colonised by the Tech Empire. With very low (and sometimes zero) corporate tax rates, many international tech companies set up their European headquarters in Dublin. People are damned because the salaries in these places are so high that if you don't work in one of them, you can't afford to live in the city (I begin to call it the San Francisco effect). The universities then scramble to educate people in the topics needed to become employed in these companies, and those alone, which, in the short-run, maybe raises employment, but in the not-even-long-run, sees a country devoid of thought, algorithmically producing machine-like staff for Tech Corp. This is an exageration, of course, I just mean to say that there are few people who stop to really think what we're losing for all this percieved growth. This is not my Ireland.

Many schools within the humanities have been closed, and even within the math and cs departments, the scope to study something 'not immediately useful' is almost non-existant.

Ireland, the 'land of saints and scholars' produced Beckett, and many great writers, scientists, actors, and politicians. Beckett is one of the most noteworthy, and one of the only ones to make it to the ENS. That is the Ireland I like to think of, and the Irish that I try to be.

Joyce is great too, and he also made it to the ENS, albeit somewhat unoficially.

I am not special, and definitely not Beckett-level special, but, as an Irish(wo)man at ENS, if it's the last thing I do, I'll stand in his room and promise to be *something*.

Je suis comme ça.
Ou j'oublie tout de suite ou je n'oublie jamais.

A CHÁIRDE

On a cool evening in Istanbul in the spring of 2015, I found myself sat at a table of cryptographers. They were a curious bunch and upon asking me what I was doing, I told them I hoped to pursue a PhD in computational geometry. They laughed, more than they should, and told me that the only right thing to do, is a PhD in cryptography. The topic was not too far from my mind, as also in that spring, *Citizen Four* had just won an academy award, *Signal* had just been launched, and my house in Ireland had just been raised by the cops because someone was using *TOR*. After a few beers and some chat, I began to think... maybe I *should* do a PhD in cryptography...

Actually, I believe I told them I was interested in "playing with shapes in space."

The following months saw me heeding the advice of the cryptographers and by the end of the year, I was admitted to one of the finest schools on the continent, to work with the "free man" of cryptography. I was very excited, but not at all prepared for what was about to come.

Now, I sit here, four years later, and there is negligible chance that I would come to write these words for you without the help, encouragement, patience, and magic of a good number of people, and one in particular. Let me take a moment to recount some of the thanks I have wished to express over these past few years.

Every time I mentioned David Naccache, people would rejoice "you know he made a computer from... plants!?"

To David. I could write a second thesis on all the things I would like to say to you. Many people would remark on your cleverness, your quirkiness, your helpfulness, but I wont, for you have given me something far greater than all of those combined. You left me alone when I wanted, you trusted me to think, to experiment, to play with my thoughts and talks and papers. Never once have you said 'no', not once have you discouraged me from dreaming up my crazy plans, nor did you scorn me when I couldn't deliver. All of this boils down to a rare and fundamental gift; you've given me *freedom*, and for this, I will be forever grateful.

Perhaps they still pale in comparison to the craziness of yours :)

To Bart. My thesis reviewer, and one of the busiest men I know. The first time I saw you, you gave a talk at Eurocrypt about Snowden and mass surveillance. I remember excitedly turning to the person beside me and asking 'who's this guy?!' to which they responded "He's the president of the IACR.." You were one of the first to make me excited to enter this field, so it's an honour to have you on my jury as I make my sort of *début*. Most people either talk to me about crypto, or the 'fluffy' social stuff I like, but you are one of the few who talks to me

about both, which I appreciate greatly. Thanks for remaining one of the 'good guys'.

To Moti. My second thesis reviewer, thanks for not living up to the reputation of the dreaded 'reviewer number 2!' You make me laugh and cheered me up on days where you didn't know I needed it. My curiosity in you arose from several events, all of which you are unaware. I reviewed some of your work, which introduced me to an area I would later become very excited about and between this and various events I've seen you speak, the intrigue silently grew. Thanks for being so accommodating throughout these past few months, your ease of style is much appreciated.

Forgive me for breaking protocol, but I dont say which papers.

To Whit. For being the *human* cryptographer. I enjoy so much to see you on the RSA panel, and every year to relinquish your opportunity to push your opinion of the latest cryptographic trends in favour of remembering the people who helped cryptography get to where it is today. Thanks for all the stories, for the few laughs, for helping me see the importance and amusement in the small things, and for being the one who kept me somewhat excited about living in Paris.

To Christian. For dragging me up and down mountains. Through rocks, and rivers, and mud, safely. For teaching me about S-boxes in Colombia, Grostl in Austria, ropes in Turkey, and knots in the snow. You're the most positive person I know, and the definition of a good friend. I am grateful for every experience we've shared, except one. I will forever curse you for introducing me to those smoked almonds. I cannot get enough of them.

To Manuel. For having patience, where others could not. It means more to me than you'll ever know. I'll say no more.

To Delaram. For being inclusive. For the encouragement, advice, the coffees, and for sprinkling a little glamour into cryptography, and into my days. It was such a pleasure to meet you in Paris and I look forward to working together in the future.

To Guénolé. For being one of the rational people at UCD. You know how hard a time I had there and I thank you greatly for being the one to make me believe there's *something better* that I can do. Thanks for doing things right, and for all the conversations and hours spent mulling over everything. Thanks for inviting me to give lectures, I didn't realise at the time how significant this was. You're the only one from Ireland who has been involved in this PhD process, and I am overjoyed that I could fulfill my promise to bring you to the jury if I ever made it to the end. Now that I'm here, I somehow feel like *we've* made it to the end.

To Michel Abdalla. For being a friend, my conference companion, my confidant, and now on my jury. Thanks for being there always.

To Michel Léger. They say you can tell a lot about a person by the company they keep. You know me now, and you know that I am not the most '*company*' type of person. Yet, you (probably singlehandedly) have made me grow to love the industry. Literally every single time we speak you are full of excitement and encouragement even on the most stressful days. It's quite amazing, and you're a force to be reckoned with. You have helped me achieve more than I ever thought possible, for which I'm super appreciative. Here's to the future, and to realising all those projects we've dreamed about :)

To Marc. I don't know what to say to Marc that I haven't already said to his face turning him a puse shade of red with embarrassment. Marc is my academic brother, with whom I sat for three years discussing everything from 'what is a primitive,' to 'how to look like Axl Rose,' to 'philosophy' at the bistro d'en face. Thanks for always listening to me, and for putting up with me when I talked about things that made you feel really awkward. Thanks for being one of the few that never made me feel stupid, and also for not giving me unsolicited advice.*

If I had to estimate, I'd say I've talk more to Marc, than everyone else combined, during my time in Paris.

**A rare skill amongst young, educated men.*

To Hiba. For being one of the most surprising and coolest people I've ever met. For taking care of a lot of the annoying stuff so we didn't have to. Your efforts did not go unnoticed and are much appreciated, really.

To Rémi. It is clear to me now why David gave you the secret to living 400 years, while he only told me how to manage 200. Thanks for being the one to do many things, quickly. Thanks also for taking care of me at the start, and for easing me into my many new lives.

To Octavio. For being so bright, and so warm, and so kind. I'm so happy to have met you. I hope you go on to do fantastic things, and act as a glue in the centre of the community. I really wish the best for you, people like you are too rare.

To Mihir. For having the special ability to teach more through one carefully constructed sentence, than most people could in a lifetime.

To James. For being my one cool non-crypto friend in Paris. I will never forget the night we met, on set at France 24, commenting live on Zuckerberg in Congress, Saudi Princes in Paris, and the UN Security Council's opinions on the use of chemical weapons. You, your "oh! are you Irish?", and your topic jumping left my mind spinning that night, as have the nights we spent together since. Thanks for adding a little variety to my life. Yer a tonic, as they say, and I hope I get to hear you sing, over and over again, and that we get to go for many Long Walks in the years to come.

Georg & Alex. For all the trips, and conversations, crypto and otherwise. You were our closest friends in Paris, and I'm so grateful for all the memories we have to look back on.

To Ireland. And the Irish. Naturally to Shell and John. To Emma and Aileen and Motja, for patiently listening to me when I talk too much during hard times, and for waiting for me, when I talk too little during the good times. To the NUIGalway gang; Kevin, Niall, Rachel, Aisling, Jerome, you shaped me in the nicest ways in my 'early' years and I'm so appreciative for all you took the time to teach me. And to Oliver, I'll be *eternally* grateful for the days we spent in UCD, my only sadness is that I didn't have a friend like you in Paris.

To ENS. For acting as my academic home over the past years. You've been everything I wanted from a school, and I'm so glad to know that places like you still exist. To the Naccache gang. Petra for the cocktails and the *sex and the city* chats. Mirko for the eh.. unspeakable. Marius for the tricks. Fatima for being the normal one—thanks for grounding us! Bruno, for being calm and putting up with us. Georges-Axel the office is all yours now. And Natacha for having a larger than life personality! And to the Crypto gang. David for being so calm and kind to everyone. Hoeteck for being fun, for preparing the best talks, and giving the toughest advice. Damien for being the proof God. Anca for showing us how to do style and substance simultaneously. Geoffroy for being like, literally everything, if I could be half what you are, I'd be doing well. Chloé for being one of the few that seriously sees the technical interest in social issues. Melissa for being **amazing**. Michele for being too cool, I hope we get to chill together in the future. Azam, for making it to Paris, I am so proud to know you. And Julia for always remembering, and taking care of us.

Finally to Stéphane and Sophie, for being the two that showed too much patience with me trying to navigate French administration.

To Ingenico. For acting as my industrial home over the past few years. I've grown to like you more than I ever thought I would. To Laurence, for taking such good care of us, and for keeping me entertained at lunchtime. To all at the labs, thanks for patiently listening to me as I got excited about too many privacy topics over the years. And to Erlin, the greatest intern that ever lived. I miss you.

Misc. There are some moments that make a profound impact on your work but you may never be sure whether to thank or curse them. Here I will share a few acknowledgments to those random things. One is for the mosquito that feasted on the ball of my palm the week before I submitted this thesis. You made these last words all the more unbearable to type. Another is for the creators of the game *Two Dots*. I'm not sure if this game has been a blessing or a curse, but it has kept me from looking at Twitter and getting very sad at the state of the world we live in. The third is to the creators of *Star Trek*. The original one, from the '60s. It is so calming that it has helped me to quieten all these thoughts and get to sleep every night. Actually, I have learned that they were quite pioneering in science.

Don't get me wrong, I believe it's super important to know what's going on in the world, but while one is writing a thesis, it can be quite distracting.

And then, After all is said and done. After all the fun is over. After I have made all of my excuses. After dinner. After too-long breakfast. After I have cried, and swore, and laughed (twice). After all the people, and the problems, have given up. After the stress and misunderstandings. After Dublin. After drawing one-way functions on the window in the Library bar. After moving. After traveling. After too much coffee. After toothaches. After all my insights, and all my ignorance. After undeserved successes. After failure, after failure, after failure. After all the untaken advice. After sushi, and after pizza. After the mistakes in booking, mistakes in proofs, mistakes in coffee machines. After missed flights, missed meetings, missed deadlines. After walks in the woods, walks in the forest, walks in the hills, walks in the *proper* mountains, walks in the rivers, walks in the mud, walks in the rocks, walks in the snow, walks in the volcanoes, after walking for weeks, after walking home. After working too much. Too much in the evenings, too much at the weekends, and sometimes not enough at all. Too often not enough at all. After Joyce, after mohsen, after beckett. After all this imbalance and inefficiency.

After all my whimsy, and my daydreams. After all my wondering what exactly *is* the difference between symmetric and asymmetric cryptography, instead of just studying katz-lindell. After Marie, and bagels and soup, and her cancerous dog. After smushed pears in the usb port of the newest macbook. After Christmas trees. And after deriving the general formula to predict the spruciness-longevity of Christmas trees. After bowls of randomness. After jungles, no, sorry, after *rainforests*. After whales. After pomegranates, and hierarchies, and sahndeviches. After broken cups, broken bowls, broken glasses, broken plates, broken bottles, and broken promises. After clicking 'submit' while we're still working. After so many typos. After so many started, so many left, so many unfinished projects. After so many words and so little action. After kale soup.

After I have changed, sometimes for the better, sometimes for worse. After reluctance to change, after stubbornness, after the thoughtlessness. After too many no's. After picking fingers and scrunchy faces. After 'you people'. After blankets, and cushions, and hot-water bottles, and foxes, and hippos, and bunnies, and goats, and marx, and gah. After political theorists. After critical theorists. After Amy and Juan, and Nermeen Shaikh. After Chomsky and Said and Rushdie and Sontag. After all the stress, after all the things I've forgotten now, and after all those things you do that I don't even notice.

After all the things the others don't see. Even after all of this, after all the afters... you remain by my side.

And now, after a moment of reflection, after thinking for a sec, after trying to instill the lessons learned, I see that all these afters sum up to one giant *before*.

So, before it's too late, and before I forget, and before we embark on our journey towards the next set of proofs, the next set of mountains, the next set of books, the next set of ..afters, let me give you these words, too many as they may be, as a marker in time, to note that however good or bad the days may seem, however many afters I may write, there are equally as many befores to come.

I can't give you thanks. I can't give you an acknowledgement. I can't give you so many things. The best I can do, in this chaotic world and point in time, is to hold your hand, and give you hope.

Alors, pa'lante, Pooya jan.

CONTENTS

LIST OF TABLES

Part I

PROLOGUE

1 INTRODUCTION

Let us begin by taking a brisk stroll through history.

1.1 THE BIRTH OF MODERN CRYPTOGRAPHY

The year is 1943. You need a key. Deciding to keep it simple, you press 'A', a rotor turns, you take some paper and write 'K'. Press 'B', write 'Q'. Press 'C', write 'G'. Again, press 'A', then 'B', then 'C'. Write 'R, N, J'. Next, you can begin communication, press 'W', write 'D' and continue; press E, T, T, E, R, B, E, R, I, C, H, T, write OAJKXTQHETTI. You have your message. Move to your radio and transmit KQGRNJDOAJKXTQHETTI. . . and you've sent your first encrypted text. Does the thought ever arise in your mind as to whether or not it is dishonest to scramble your message? You do this for the sake of national security, for strategy in time of war, for your nation. You need ask no questions; this is your duty.

We jump to 1970. The height of the post-war, Golden Age of Capitalism. Electronic fund transfers (EFTs) are rampant and the number of issued credit cards surpasses 1 million in the United States. The world's economy is booming. Life is sweet.

It's 1977. Recovering from the 73–75 recession you are more skeptical about EFTs. Data protection laws surrounding the collection of payment information are passed. You need more secure systems and welcome the development of DES [PUB77]. But to use it is not straightforward. What was once an instrument solely used for military advantage, encryption is now commercially required due to post-recession insecurities and the growth of electronic and computing industries, and is permissioned only via the licensing of your IP.

Let's move to 1991. You possess your own Personal Computer. Imagine that! For the first time you see the ability to encrypt moving into the hands of the citizen. This yields excitement, but also, it is immediately obvious that this will cause some consternation. On one hand, the First Amendment of the U.S. Constitution [U.S. Const. am. I] strongly protects freedom of speech and expression, which—in a roundabout way—means that cryptography within the U.S. can not be controlled. On the other hand, cryptography remains on the U.S. Munitions List, meaning that its export is still heavily regulated.

The next decade sees some of the bloodiest years of the Crypto Wars. With global connection to the Internet, pressure mounts on the U.S.

Government to loosen the laws surrounding the export of software. The battles were fought in court and in 1996 [Cli96], encryption software was removed from the munitions list. By the turn of the century, rules surrounding the export of commercial and open-source software containing cryptography were greatly simplified, restrictions on keys were lifted, backdoors were prohibited, and you, the citizen, felt that progress was underway.

FAST FORWARD TO JUNE 2013. You sit happily tip tapping on your smartphone, sharing doge memes and giggling over screaming goats, before moving on to check the news. You learn that the U.S. Government has forced Verizon to hand over the phone records of millions of Americans [Gre13]. It's not such a nice story. Over the coming days you see more articles of a similar vein. You discover the NSA's direct access to data held by all your beloved internet giants. You learn of secret programs and backdoors, and within months, you come to terms with the fact that you live in a quasi-surveillance state. This is a grave situation and once again, as you did two decades ago, you find yourself debating the same disparity between individual privacy and state security.

BETWEEN THEN AND NOW. The debate raged on and the disparity still exists. Several nations have the desire to forbid encryption, to keep it as a military tool, to stifle progression, to retain control, and to undermine democracy. But there has also been much progress. In 2018, the EU put a new regulation into place: the General Data Protection Regulation [Reg16]. It expands and solidifies the points set out in the Charter of Fundamental Rights of the European Union [Uni12] and the Treaty on the Functioning of the European Union [Uni07] which state that EU citizens have the right to privacy both online and offline. It insists that in order to maintain security of the individual while ensuring compliance with the regulation, appropriate measures (such as encryption) must be used. The regulation covers the collection, storage, processing, and deletion of personal information and communications. The regulation applies to the handling of personal information within Europe irrespective of the location of the organisation handling the data.

The United Nations Human Rights Council and the General Assembly have also specified the necessity for encryption to ensure the right to privacy in the digital age, building their argument by paying particular focus to the dangers faced by journalists. The UN [HRC16]

> *Emphasizes* that, in the digital age, encryption and anonymity tools have become vital for many journalists to exercise freely their work and their enjoyment of human rights, in particular their rights to freedom of expression and to privacy, including to secure their communications and to protect the confidentiality of their sources, and calls upon

> States not to interfere with the use of such technologies, with any restrictions thereon complying with States' obligations under international human rights law.

These are simply two examples of many that build upon the arguments of yore. However strong were the efforts made by the privacy advocates in the 90's, they were very few voices. Now, with technology in every inch of our lives, with the increased media attention due to the Snowden revelations, high profile court cases, and freer flowing information, citizens are much more aware of the consequences of not using encryption. This time around, there are many voices.

And so here we are. You have come a long way since your button pressing days on the Enigma. You've seen four world recessions, men walking on the moon, the fall of the Berlin wall, and some moves towards equality. You've danced to records, to cassette tapes, to mp3s, and now to Spotify. You must be tired. But you cannot sit yet! The world has grown around you, and there are problems old and new left to be solved. If I ask you to send me an encrypted email, right here and now, can you do it? If I ask you to remove any records of me, to grant my request to be forgotten, is it easy? If I want to travel, to meet people in the world, will you stop me at the border? Will you question me and demand my passwords? If I want to talk, to exercise a curiosity, to learn and teach and spread information, but without prying eyes, will you let me? Ultimately, all I'm asking is to exercise a right, is it possible?

Until the answer to all of these questions is a definitive **Yes**, then I'm afraid we've still got some work to do.

1.2 CONTRIBUTIONS OF THIS THESIS

We saw the development of several worlds all becoming dependent on cryptography. While a traditional thesis aims to dig deep and focus on one particular topic, the goal of this work was to arrive at a level of scientific understanding and creativity that would allow to tackle general problems that arise in this increasingly technological society. As such, this thesis studies a variety of cryptographic topics. It encompasses new theoretical results, new security notions, constructions and generic transforms for both public-key and symmetric cryptosystems, and efficiency gains. The results mainly appear in three papers, briefly outlined below.

1.2.1 *Try Again*

Part II introduces a new attack model, namely key-correlated attacks (KCA), that subsumes two previous models (related-key attacks, and key-dependent message attacks), and further shows that even if these

two security notions are both guaranteed, together they are still not enough to yield KCA-security as was previously thought. As such, it is a lesson that we should expect *try again* before getting the most complete security notions.

KEY-CORRELATED ATTACKS. The part encompasses the security of symmetric primitives. In Chapter 3 we give the background and motivation for why it is interesting to study the security of key-dependent messages encrypted under related keys. Chapter 4 describes a new security notion that provides the strongest security for symmetric primitives proven in the random oracle model (ROM). Key-correlated attacks (KCA) model the scenario where all inputs (keys, messages, and possibly nonces and headers) are correlated with the secret key. We show relations and separations with older notions (related-key and key-dependent message security) to show that security under KCA is strictly stronger, and implies the others. In Chapter 5 we describe the assumptions necessary to prove KCA security of block-ciphers, and show that 3-rounds of Even-Mansour are necessary to achieve this. Finally, in Chapter 6 we define a KCA-security notion for nonce-based authenticated encryption (AE), and provide a black-box transformation that turns a multiuser-secure AE into an AE scheme that is provably KCA secure in the ROM.

1.2.2 *Fail Again*

Part III turns to public-key cryptography, and describes an attack against a new assumption. Experimentally we show that we can obtain the secret key without too much computing power. We go on and attempt to modify the assumption and construct a variant scheme, but alas, it also succumbs to the same attacks, and hence in this part we are destined to *fail again*.

UNSTUDIED ASSUMPTIONS. Chapter 7 analyses the assumptions underlying the new public-key cryptosystem of [AJPS17c]. Cryptanalysis of their assumption, based on arithmetic modulo a Mersenne prime, allowed us to reconstruct the secret key, given only the public key. Based on a modified version of the assumption, in Chapter 8 we propose a variant post-quantum secure public-key cryptosystem. This variant is also vulnerable to the attacks described against [AJPS17c], but the scheme still could be of use as it is a simpler construction, is based on a different assumption, and thus potentially easier to cryptanalyse.

1.2.3 *Fail Better*

Part IV turns to efficiency, and studies the efficiency of signature schemes. We describe a way to efficiently generate a batch of Schnorr signatures using the same amount of 'random' as was traditionally

required to generate one signature. We fail to maintain security under traditional assumptions and are required to reduce to a new assumption. If we are going to fail to use traditional assumptions, at least we gain efficiency and *fail better*.

EFFICIENCY. Chapter 9 sees the exploitation of the group structure used in Schnorr signatures to reuse the nonce, with which we can then generate a batch of signatures. This, together with some preprocessing tricks, allow us to increase the efficiency of Schnorr signature generation. We prove security of the signature scheme in the random oracle model, but require a new assumption which we show to be intractable in the Generic Group Model.

2

PRELIMINARIES

When writing cryptography, there are a number of design choices that need to be made. There are definitions, notations, paradigms, frameworks and each can be presented in different styles. This chapter aims to clarify conceptual and notational choices, by introducing the language and tools used throughout this thesis.

2.1 PROVABLE SECURITY

It is often the case that we cannot prove security in an absolute sense. This can be limiting because, in practice, we are not tasked with defending against adversaries that have *unlimited* resources to perform attacks against the system we are trying to protect. Yet, at the same time we should hope to protect against more than just ad-hoc security approaches. It used to be, and is sometimes still, the case that schemes are built according to an iterated 'bug, fix, bug, ...' approach, where there is no formal proof of security, and no particular attack. If flaws are found, the scheme is updated to capture and protect against them. This is not very safe at all, and we see in practice that bugs *are* found, but rather than being disclosed, they're saved, stored, and used to exploit a system at a later date. We can't always get systems with *full* and *unconditional* security, yet, it's risky to rely on schemes that have *no* proof of security, so what to do?

These are known as zerodays and are actually very scary.

In fact, there's only one time *we can get it.*

A successful paradigm that emerged to cover a middle ground is known as *provable security*. It allows us to:

1. Abstract and *define* general notions for cryptographic primitives which enable us to *reason* about their security.
2. *Relate* these notions to each other to form a clearer picture about the relative strength of the security guarantees of a particular primitive.

Overall, it gives us a *sense* of the security and gives us the confidence to say that a particular primitive has no inherent design flaws.

For example, AES has no security proof, but with provable security we can reason well about the security of a generic blockcipher. Then we can ask ourselves if the specific instantiation (AES) satisfies that reasoning.

DEFINITIONS. Generally when we talk about cryptographic objects, we refer to them as primitives. We define a general class of primitives, rather than relying on reasoning about the security of a particular instance or instantiation of a primitive. These primitive abstractions are built in the form of *definitions*. In writing definitions, there are a number of things to consider:

SYNTAX is the language used to describe this primitive. It lays down the algorithms, mandates whether they are randomised or deter-

ministic, describes the inputs expected by the algorithms, notes from which domain (or *space*) the inputs can be chosen, and the outputs the algorithms return, along with the range or space in which they can lie.

CORRECTNESS describes the interaction between algorithms that forces the primitive to *work correctly*. For example, in any given encryption scheme, we can say that it is *correct* if it is always the case that the message obtained from the decryption of a ciphertext is the same message that was encrypted to produce that ciphertext.

For example, we may wish to achieve the security goal of confidentiality, or authenticity.

SECURITY is where the fun starts. Security specifies the *goal* that the primitive should achieve, and the *power* of an attacker trying to inveigle the system. It is often expressed as a *game* played by an adversary with a given set of computational resources. We say that the security goal is achieved if the adversary's *advantage*, or their probability of winning the game, is below some threshold.

REDUCTIONS. Given a definition of a primitive, we know how it should look (through its syntax), how it should behave (through its correctness), and what security goals it should achieve when faced by an adversary. In proving that the primitive achieves these goals, we need to *bound* the advantage of the adversary playing the security game. To generate these bounds, we often need to rely on some *assumptions*. The assumptions used are often well studied problems that are universally considered to be *hard enough* at a given point in time. New assumptions undergo careful cryptanalysis by many in the community, but we'll see more on that later, and particularly in Part III. Equipped with carefully studied assumptions, the aim is to show a relationship between the hardness of breaking primitive and the hardness of the assumption. Describing this relationship is typically done via a *reduction*.

Without assumptions, most proofs would boil down to solving the $\mathsf{P} \stackrel{?}{=} \mathsf{NP}$ *problem along the way, and this is still a bit beyond us.*

A reduction is simply a process that shows how to transform an efficient adversary $\mathcal{A}$ that succeeds in breaking the target primitive, into a solver $\mathcal{B}$ that solves the problem that was assumed to be hard. Reductions often proceed in the contrapositive, and we say that because it its assumed that no solver $\mathcal{B}$ exists to solve the hard problem, it cannot be the case that we have an adversary $\mathcal{A}$ that breaks the primitive.

Throughout this thesis, particularly in Parts II and IV we will see a number of proofs by reduction.

2.2 HARD PROBLEMS

Now that we're equipped with our primitive definitions, and our methods to prove their security, we need some assumptions to reduce to. There are a number of hard problems, and we are forever searching

for more, but often the very first thing people think of when they hear the word cryptography are the elusive *prime numbers*. Their infamousity is largely due to the fact that the most widely used public-key cryptosystem, RSA, is based on the hardness of integer factoring. It was not, however, the first hard problem used to construct public-key encryption. In 1976, Diffie and Hellman [DH76] proposed methods for key exchange and digital signatures based on the Discrete Log Problem. Below, you'll find a brief description of these, and other hard problems used in cryptography.

2.2.1 *One-way Functions*

Modern cryptography exists largely due to the fact that we have some functions which we assume to be *one-way*. The problem is that we do not have functions that are *known* to be one-way, and so we rely on this as an assumption. However, this assumption would not form the basis for modern cryptography if we did not have some candidates that we strongly believe to be one-way.

ONE-WAY FUNCTION. A function is said to be 'one-way' if it satisfies two properties:

1. It's *easy* to compute.
2. It's *hard* to invert.

It's as simple as that. Here, *easy* means that if we're given a uniform value x from the domain of some function f, $y := f(x)$ should be computable in polynomial time. On the other hand, *hard* means that if we're given such a y that was produced by applying the function f to the uniform value x, it should be infeasible for any probabilistic polynomial time algorithm to invert f. The following paragraphs describe some candidates strongly assumed to be one-way.

2.2.2 *Integer Factorisation*

INTEGER FACTORISATION. Suppose we are given $n = pq$ with p, q prime. The problem is simply to find p. Provided that p and q are *large enough* this problem is assumed to be hard.

RSA PROBLEM. Given integers n, x, e the problem posed is to compute y such that $y^e = x \bmod n$. One way to solve this problem is by factoring n as above. In fact, it could be the only practical way, see [Bro16], which addresses amongst others some initial doubts raised by [BV98]. Assuming access to special oracles, efficient attacks are known, see e.g. [JLNT09].

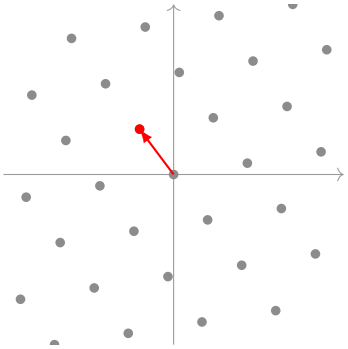


Figure 2.1 – Illustration of the shortest-vector problem in a two-dimensional lattice.

2.2.3 *Discrete Logarithms*

Discrete Logarithm Problem, DLP. Let $\mathbb{G}$ be a group, and g be a generator of $\mathbb{G}$. Given $y \in \mathbb{G}$, find x such that $g^x = y$.

The DLP was first introduced by Diffie & Hellman [DH76] to enable key exchange by describing the function as a trapdoor the exponentiation in a finite field. Fixing a prime p and a generator g of the finite field of order p, computing $g^x \bmod p$ is easy (i. e. polynomial time) by fast exponentiation. The converse, which is finding x given $g^x \bmod p$ is hard. The discrete logarithm problem lies at the heart of a vast proportion of modern public-key cryptography and has many variations. Below we list some, and introduce a new one in Part IV.

For some primes, den Boer [den90], Maurer [Mau94], and others have shown equivalence between CDH *and* DLP *but the problem is still open.*

Computational Diffie-Hellman, CDH. Let $\mathbb{G}$ be a group, and g be a generator of $\mathbb{G}$. Given g^x and g^y, compute g^{xy}. The best approach to solving this problem is by using a solution to the DLP.

Decisional Diffie-Hellman, DDH. Let $\mathbb{G}$ be a group of order q, and g be a generator of $\mathbb{G}$. Given g^x and g^y, it should be hard to decide, or *distinguish* the value g^{xy} from g^z, where z a random element in $\mathbb{G}$.

2.2.4 *Lattice Based Problems*

Lattice-based cryptosystems rely on the hardness of solving certain problems appearing in discrete subgroups of $\mathbb{Z}^n$, typically of the form $\mathcal{L} = \sum_i c_i \mathbf{b}_i$ where $b_i \in \mathbb{Z}^n$ are called the *basis vectors*. The most common hard problem is the shortest vector problem (SVP), which intuitively seeks to find the shortest vector from any point to a point in the lattice. More formally:

In higher dimensions, the problem of finding the shortest vector of a lattice is believed to be hard, this then can be leveraged to construct public-key cryptographic schemes.

SVP_γ. Given a basis of $\mathcal{L}$, find a non-zero vector $\mathbf{v} \in \mathcal{L}$ of length $\|\mathbf{v}\| \leq \gamma \lambda_1(\mathcal{L})$, where $\lambda_1(\mathcal{L})$ denotes the length of the shortest vector in $\mathcal{L}$. Figure 2.1 illustrates the SVP problem on a simple case.

The results in Part III pertain to lattice-based public-key cryptography.

2.3 IDEALISED MODELS OF COMPUTATION

Now we have our definitions, our reductions, our assumptions, and some examples, are we ready to start yet? Not exactly..

Hard problems, assumptions, and security in general, have been studied for a number of decades now. Traditionally in proving security, we had a well defined security target that we hoped to achieve. Yet, as cryptographic primitives become increasingly complex, the notions of security get deeper, and these schemes are used in ways that were not necessarily envisioned when they were designed. As such, it's not always the case that we know *exactly* what property we want to achieve, so there is a move to prove security in a more *general* sense. Rather than reducing to a specific hard problem as outlined above, we can also leverage *black-box* reductions where we are given access to some *ideal* object that encompasses many desired security properties. Proving security within these ideal models allows us to keep proofs self-contained, and to bound adversaries against a general set of attacks.

RANDOM ORACLES. The random oracle model ([BR93; CGH98; FS87]) models the operation of a random function as queries to an entity called a random oracle. When queried with a given input, the random oracle replies with a truly random number from some sample space. Input queries and corresponding outputs are stored in a table; such that upon receiving the same query twice, the oracle returns the same corresponding output value. With a 'full' table containing all input/output values from a given space, a random oracle models an ideal hash function.

There are some uses of Random Oracles that we will call upon throughout this thesis.

1. Lazy Sampling. Working in these idealised models, H can be defined over an infinite domain. This can be technically cumbersome to deal with, but also non-intuitive to reason about. For this reason, it is often convenient to *lazily sample* H such that as queries arrive, the output values are uniformly sampled, and the input/output pair is stored in the table. This way, the table is built up over time, rather than being pre-defined before use.
2. Programmability. Before we said that a *reduction* is process describing how to convert an adversary that breaks the scheme in question, into an adversary that solves some hard problem. A property gained by using random oracles within a reduction is that adversarial queries to H can be *seen* by the reduction and as such, the reduction can return or *program* H to return appropriate values.

3. Forgetfulness. Upon receiving a query, H returns a uniformly random value, without checking whether or not this query has been made before.

I believe this to be one of the more over-used and unexciting comments regarding random oracles.

However, no efficient algorithm can implement a true random oracle; this can lead to the situation where a cryptosystem is provably secure in the ROM, but provably *insecure* when instantiated with any concrete hash function [CGH98]. However, such constructions are very unnatural from a cryptographic point of view [KM15], but illustrate that there exists a gap between the ideal model and real-world implementations.

Ideal Cipher Model. The ideal cipher model (ICM) is very close in spirit to the random oracle model. Where the ROM models a random function (or an ideal hash function), the ICM models a random blockcipher (or an ideal cipher). Many schemes have been proven secure in the ICM [BRS02; Des00; EM93; KR01]. As with the ROM, there are (artificial) schemes that are provably secure in the ICM, but provably insecure for any concrete block cipher [Bla06]. In fact the ROM and ICM are equivalent [CPS08].

The reduction given in [CPS08] is not tight; thus the ROM and the ICM are qualitatively *equivalent, but the ICM is* quantitatively *a stronger assumption.*

Generic Group Model. The generic group model (GGM) was introduced by Shoup [Sho97] and gives an adversary access to a randomly chosen encoding of a group, rather than a specific efficient encoding.

The generic group model is useful, in that it gives more theoretical power to prove statements, but it is unsatisfactory from a cryptographic standpoint, because *no group is truly generic*. Groups used in practice have additional structure, which leads to more efficient algorithms. Like the ROM and the ICM, cryptosystems which are provably secure in the GGM may be *insecure* when the generic group is replaced by *any* concrete group [Den02; CGH98].

2.4 NOTATION

Before getting into descriptions of primitives used throughout the thesis, you will find listed below some preliminary notation to assist in reading.

Strings. We let $\{0,1\}^n$ denote the set of bit strings of length n and $\{0,1\}^*$ the set of all finite-length bit strings. For two bit strings X and Y, $X|Y$ denotes their concatenation and (X,Y) denotes a uniquely (and efficiently) decodable encoding of X and Y. The length of X is denoted by $|X|$.

Sampling. By $x \xleftarrow{} S$ we mean sampling x uniformly from set S, whereas by $y \xleftarrow{} \mathcal{A}(x)$ we mean the action of assigning the output of the randomized algorithm $\mathcal{A}$ on input x to y. Assignment from

deterministic algorithms and calculations are denoted by the symbol $\leftarrow$.

LISTS. We denote appending element X (resp., a list L') to a list L by $\mathsf{L} : X$ (resp., $\mathsf{L} : \mathsf{L}'$).

GAMES. We adopt the code-based game-playing language of Bellare and Rogaway [BR06], for which all lists are initialized to empty and all bad flags to false. A game Game sometimes consists of an initializing procedure Init, one or more procedures to respond to oracle queries, and a finalizing procedure Fin. We denote changes from one game to the next by highlighting $\boxed{\text{boxed}}$ sections.

PARAMETERS. Following the concrete security approach, we will primarily content ourselves with defining advantages and providing concrete reductions, without dwelling too much on the question when a scheme is actually deemed secure or not (e.g., for a sufficiently large class of adversaries, the advantages are sufficiently small). One can of course easily recast our work in an asymptotic framework (where for all probabilistic polynomial-time adversaries advantages should be negligible in the security parameter λ, i.e., in $\lambda^{-\omega(1)}$). From time to time we do use an asymptotic approach, for measuring efficiency, and in such cases we denote the security parameter by $\lambda \in \mathbb{N}$ which is given to all algorithms in the unary form 1^λ.

ALGEBRA. If n is an integer, we write $\mathbb{Z}_n$ for the ring $\mathbb{Z}/n\mathbb{Z}$. We let $\mathbb{Z}_n^*$ denote the invertible elements of $\mathbb{Z}_n$. The set of numbers $1, 2, \ldots, k$ is denoted $[k]$.

ALGORITHMS. $f \in \mathrm{Negl}(\lambda)$ denotes a function that decreases faster than the inverse of any polynomial in λ; such functions are called *negligible*. We often refer to an *efficient* algorithm or a probabilistic polynomial time (PPT) algorithm, which is a probabilistic algorithm running in time polynomial in its input size, on all inputs and all random coins.

2.5 SYMMETRIC CRYPTOGRAPHY

Symmetric, or private-key *cryptography was the first developed notion of cryptography and remains today the most efficient and widely used primitive.*

In Part II we will take a look at contributions in symmetric cryptography. This section builds up the preliminaries and notation necessary to read fluidly there.

Symmetric primitives are largely based on blockciphers built from pseudorandom permutations and pseudorandom functions. These in turn are used to build symmetric encryption primitives such as Authenticated Encryption and its variants. Outlined below the reader will find the descriptions from basic to advanced primitives, and the security notions associated with them. Before we can build anything we need some randomness.

A PRF *is the generalisation of a* PRG.

PSEUDORANDOM FUNCTIONS (PRF). A family of functions $F : \mathcal{K} \times$

$\mathcal{D} \to \mathcal{R}$ is a two argument function that takes a key K from the *key space* $\mathcal{K}$, and input x in the *domain* $\mathcal{D}$ and returns an output $F(K, x)$ in the *range* $\mathcal{R}$. We define the PRF-advantage of an adversary $\mathcal{A}$ with respect to the function family F as:

$$\mathbf{Adv}_F^{\mathsf{prf}}(\mathcal{A}) = |\Pr[\mathcal{A}^{F_K(\cdot)} = 1 : K \twoheadleftarrow \mathcal{K}] - \Pr[\mathcal{A}^{f(\cdot)} = 1 : f \twoheadleftarrow \mathrm{Func}(\mathcal{D}, \mathcal{R})]|$$

F is said to be a pseudorandom function if the above advantage is "small" for every "reasonable" adversary $\mathcal{A}$.

A PRP *is a restricted class of* PRF*s mapping inputs to outputs in the same domain. They are really the fundamental building block of symmetric cryptography.*

Pseudorandom Permutations (PRP). A family of permutations $\Pi : \mathcal{K} \times \mathcal{D} \to \mathcal{D}$ is a two argument function that takes a key K from the key space $\mathcal{K}$, and input x in the domain $\mathcal{D}$ and returns an output $\Pi(K, x)$ in $\mathcal{D}$. We define the PRP-advantage of an adversary $\mathcal{A}$ with respect to the permutation family Π as:

$$\mathbf{Adv}_\Pi^{\mathsf{prp}}(\mathcal{A}) = |\Pr[\mathcal{A}^{\Pi_K(\cdot)} = 1 : K \twoheadleftarrow \mathcal{K}] - \Pr[\mathcal{A}^{\pi(\cdot)} = 1 : \pi \twoheadleftarrow \mathrm{Perm}(\mathcal{D})]|$$

Π is said to be a pseudorandom permutation if the above advantage is "small" for every "reasonable" adversary $\mathcal{A}$.

2.5.1 *Blockciphers*

The most commonly used blockcipher models are the Feistel network, and Even–Mansour.

Blockciphers are the fundamental building block for encrypting messages under a shared key. They can be constructed in a number of ways by connecting PRPs together under various *models*. Now, let's put all the earlier preliminaries into action and think about blockciphers more formally.

Blockciphers. Given a non-empty finite set $\mathcal{K}$ and a non-empty set $\mathcal{M}$, called the key space and the message space respectively, we let $\mathrm{Block}(\mathcal{K}, \mathcal{M})$ denote the set of all functions $\mathsf{E} : \mathcal{K} \times \mathcal{M} \to \mathcal{M}$ such that for each $K \in \mathcal{K}$ the map $\mathsf{E}(K, \cdot)$ is

1. a permutation on $\mathcal{M}$, and
2. length-preserving, in the sense that for all $M \in \mathcal{M}$ we have that $|\mathsf{E}(K, M)| = |M|$. Such an E uniquely defines its inverse $\mathsf{D} : \mathcal{K} \times \mathcal{M} \to \mathcal{M}$.

A *blockcipher* for key space $\mathcal{K}$ and message space $\mathcal{M}$ is a tuple of efficient algorithms $\mathsf{BC} := (\mathsf{E}, \mathsf{D})$ such that $\mathsf{E} \in \mathrm{Block}(\mathcal{K}, \mathcal{M})$ and D is its inverse. We assume that the keys of a blockcipher are chosen uniformly from the key space $\mathcal{K}$, which is typically $\{0,1\}^k$ for some $k \in \mathbb{N}$ called the key length. Algorithm E is the deterministic enciphering algorithm $\mathsf{E} : \mathcal{K} \times \mathcal{M} \to \mathcal{M}$. Typically $\mathcal{M} = \{0,1\}^n$ for some $n \in \mathbb{N}$ called the block length. Algorithm D is the deterministic deciphering algorithm $\mathsf{D} : \mathcal{K} \times \mathcal{M} \to \mathcal{M}$.

A blockcipher is *correct* if for all $K \in \mathcal{K}$ and $M \in \mathcal{M}$ we have $\mathsf{D}(K, \mathsf{E}(K, M)) = M$. A (public) permutation on $\mathcal{M}$ is a blockcipher with a singleton key space $\mathcal{K} = \{\varepsilon\}$. We denote a permutation with P and its inverse with P^{-}.

IDEAL CIPHERS. The ideal cipher for key space $\mathcal{K}$ and message space $\mathcal{M}$ is the uniform distribution over $\mathrm{Block}(\mathcal{K}, \mathcal{M})$. The ideal-cipher model (ICM) for given key and message spaces $\mathcal{K}$, $\mathcal{M}$ is a model of computation where all parties, honest or otherwise, have oracle access to a uniformly random element in $\mathrm{Block}(\mathcal{K}, \mathcal{M})$ and its inverse. The ideal-cipher model when restricted to $\mathcal{K} = \{\varepsilon\}$ gives rise to the random-permutation model (RPM). We abbreviate $\mathrm{Block}(\{0,1\}^k, \{0,1\}^n)$ by $\mathrm{Block}(k, n)$ and $\mathrm{Block}(\{\varepsilon\}, \{0,1\}^n)$ by $\mathrm{Perm}(n)$.

The ideal-cipher model and the random-permutation model will be used extensively in Chapter 5 to assist in proving security for blockciphers.

2.5.2 *Symmetric Encryption*

Blockciphers are very useful, and will call upon them often, but they are subject to some constraints that can be debilitating in the real world. Because they are permutations, and deterministic, it means that the encrypted message returned will always be the same given the same input, and all input output pairs will have the same length. Hence upon intercepting an encrypted message, the adversary can already infer information about the plaintext. We extend the ideas found in blockciphers to overcome these obstacles by defining possibly randomised encrypting algorithms which are not restricted to produce a ciphertext of the same length as the message. As such, we define a more *general* sense of encryption, which, when used with a shared secret key, we call symmetric encryption (SE). Let's use the framework from item 2.1 to define a symmetric encryption scheme.

Spoiler: we get around these constraints by using modes of operation to turn blockciphers into encryption schemes.

SE SYNTAX. A *symmetric encryption* scheme SE consists of three algorithms $\mathrm{SE} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ that work as follows:

1. Gen is the randomized key generation algorithm that returns a bit string K from a finite key-space $\mathcal{K}$ (which is typically $\{0,1\}^k$ for some $k \in \mathbb{N}$).
2. $\mathsf{Enc} : \mathcal{K} \times \mathcal{M} \twoheadrightarrow \mathcal{C}$ is the randomized encryption algorithm which takes as input a key K, a message M from message space $\mathcal{M}$, and returns a ciphertext C from some ciphertext-space $\mathcal{C}$. We write this as $C \leftarrow \mathsf{Enc}(K, M)$.
3. The deterministic decryption algorithm $\mathsf{Dec} : \mathcal{K} \times \mathcal{C} \to \mathcal{M} \cup \{\perp\}$ takes a key K, a ciphertext C, and returns either a plaintext M or the failure symbol $\perp \notin \mathcal{M}$.

SE CORRECTNESS. Correctness requires that $\mathsf{Dec}(K, \mathsf{Enc}(K, M)) = M$ for all values of the inputs above.

2.6 SECURITY NOTIONS FOR ENCRYPTION

Taking a look at any security book, you will not be able to escape the three pillars of information security, namely, confidentiality, integrity, and authenticity.

CONFIDENTIALITY provides the guarantee that the message cannot be understood by anyone other than those intended to read them. It is most often guaranteed by symmetric and public-key encryption.

INTEGRITY convinces us that a ciphertext has not been altered or tampered with on its journey from sender to receiver. Integrity is often guaranteed by hash functions.

AUTHENTICITY gives the assurance that the message received by the receiver is indeed the one sent by the sender. This is usually guaranteed by sending a digital signature, or a message authentication code.

It is good to model the strongest possible adversaries to give the best security guarantees.

Any good encryption scheme aims to achieve confidentiality at the very least. Depending on how much *power* is given to the adversary, this can be easy or hard. There are a number of notions to capture the power of an adversary; some are old, some are newer, and we consider here only the stronger notions of *chosen-plaintext attacks* (CPA), and *chosen-ciphertext attacks* (CCA) as outlined below and captured in Figure 2.2.

CPA-SECURITY In conducting a chosen-plaintext attack, and adversary $\mathcal{A}$ is given access to an encryption oracle ENC under which it can encrypt plaintext messages M of its choice. When queried, ENC encrypts messages under a key K which remains unknown to $\mathcal{A}$. The encryption oracle then returns a ciphertext $C \leftarrow \text{ENC}_K(M)$ as the reply to the query.

CCA-SECURITY In a chosen-ciphertext attack, and adversary $\mathcal{A}$ has access to ENC, and an additional oracle for *decryption*, DEC, under which it can decrypt ciphertexts C of its choice. When queried, ENC encrypts messages under a key K which remains unknown to $\mathcal{A}$. The encryption oracle then returns a ciphertext $C \leftarrow \text{ENC}_K(M)$. When queried, DEC decrypts ciphertexts under a key K which remains unknown to $\mathcal{A}$. The decryption oracle then returns a plaintext message $M \leftarrow \text{DEC}_K(C)$ as the reply to the query. To avoid trivial wins, we disallow the adversary to query DEC (resp. ENC) with ciphertexts (resp. messages) that were previously generated by ENC (resp. DEC).

It is clear to see that CCA is the stronger notion of the security as the adversary is given access to both encryption and decryption oracles. This is the common standard security goal to aim for, and the one used in chapters 4 and 5.

Game $\text{CPA}^{\mathcal{A}}_{\text{SE}}$:	Game $\text{CCA}^{\mathcal{A}}_{\text{SE}}$:
$b \twoheadleftarrow \{0,1\}; K \twoheadleftarrow \mathcal{K}$	$b \twoheadleftarrow \{0,1\}; K \twoheadleftarrow \mathcal{K}$
$b' \twoheadleftarrow \mathcal{A}^{\text{Enc}}$	$b' \twoheadleftarrow \mathcal{A}^{\text{Enc},\text{Dec}}$
return $(b' = b)$	return $(b' = b)$

Figure 2.2 – CPA and CCA experiments for a symmetric encryption scheme.

We say that we aim to achieve confidentiality through ciphertext indistinguishability, and that we give and adversary access to oracles that return ciphertext and/or plaintexts, but what exactly does it try to distinguish? There are a number of *flavours* of indistinguishability game, but there are two more common than others, which are described as follows:

LEFT-OR-RIGHT INDISTINGUISHABILITY (LR): A challenge bit b and key K are chosen. An adversary $\mathcal{A}$ queries two messages (M_0, M_1) to the encryption oracle. One of the messages, corresponding to the challenge bit is encrypted and returned as C_b. The adversary must then distinguish whether M_0 or M_1 was encrypted. It is often mandated that $\mathcal{A}$ can make at most q queries, after which their guess is returned as b'. $\mathcal{A}$ wins if $b' = b$. The advantage of $\mathcal{A}$ against a symmetric encryption scheme SE in the LR game is defined by

$$\mathbf{Adv}^{\text{lr-cca}}_{\text{SE}}(\mathcal{A}) = 2 \cdot \Pr[b' = b] - 1.$$

REAL-OR-RANDOM INDISTINGUISHABILITY (RR): As in LR, a challenge bit b and key K are chosen. In this game, an adversary $\mathcal{A}$ queries a single message (M) to the encryption oracle. Depending on the bit b, the oracle will encrypt either the real message, or will choose a string uniformly at random from the prescribed ciphertext space and returns the ciphertext as C_b. The adversary $\mathcal{A}$ must distinguish whether it is an encryption of the real message or of a random string. Often with $b = 1$ being 'real' and $b = 0$ being 'random', $\mathcal{A}$ returns a bit b' and wins if $b' = b$. The CCA advantage of $\mathcal{A}$ against SE in the RR game is defined by

$$\mathbf{Adv}^{\text{rr-cca}}_{\text{SE}}(\mathcal{A}) = 2 \cdot \Pr[b' = b] - 1.$$

These notions appeared first in [BDJR97] and are displayed pictorially in Figure 2.3 as a starting point to see how games will be presented throughout the thesis.

2.6.1 *Authenticated Encryption*

So far we have understood some provable security, something about blockciphers, using them to make symmetric encryption, are we done? Not yet. There's one more thing to consider...

Symmetric encryption gives strong guarantees for the confidentiality of a ciphertext. We know that in the CCA security model, even if we give an adversary $\mathcal{A}$ access to encryption and decryption oracles

Game LR-CCA$^{\mathcal{A}}_{\mathsf{SE}}$:
$b \twoheadleftarrow \{0,1\}$; $K \twoheadleftarrow \mathcal{K}$
$b' \twoheadleftarrow \mathcal{A}^{\text{LRENC,LRDEC}}$
return $(b' = b)$

Proc. LRENC(M_0, M_1):
$C_0 \leftarrow \mathsf{Enc}(K, M_0)$
$C_1 \leftarrow \mathsf{Enc}(K, M_1)$
$\mathsf{CL} \leftarrow \mathsf{CL} : (K, C_b)$
return C_b

Proc. LRDEC(C_0, C_1):
if $(K, C) \in \mathsf{CL}$: return $\perp$
$M_0 \leftarrow \mathsf{Dec}(K, C_0)$
$M_1 \leftarrow \mathsf{Dec}(K, C_1)$
return M_b

Game RR-CCA$^{\mathcal{A}}_{\mathsf{SE}}$:
$b \twoheadleftarrow \{0,1\}$; $K^* \twoheadleftarrow \mathcal{K}$
$b' \twoheadleftarrow \mathcal{A}^{\text{RRENC,DEC}}$
return $(b' = b)$

Proc. RRENC(M):
$C_0 \leftarrow \mathsf{Enc}(M_0)$
$C_1 \twoheadleftarrow \{0,1\}^{|C_0|}$
$\mathsf{CL} \leftarrow \mathsf{CL} : (K, C_b)$
return C_b

Proc. DEC(C):
if $(K, C) \in \mathsf{CL}$: return $\perp$
$M \leftarrow \mathsf{Dec}(K, C)$
return M

Figure 2.3 – Left-or-right, and real-or-random indistinguishability games for a symmetric encryption scheme SE in the CCA-security model.

so they can obtain ciphertexts and messages, we can still maintain security (ciphertext indistinguishability). However, it is often the case that this is insufficient. Attacks have shown, that unless the encryption is authenticated in some way, the integrity of the system can be compromised. Authenticated Encryption aims to solve this by producing a *message authentication code* based on the plaintext, and encrypting this together with the plaintext to create the ciphertext. Although they differ in instantiation and implementation, there is only one syntactical difference between a symmetric encryption scheme SE and an authenticated encryption scheme AE. The encryption scheme Enc is randomised in SE, but is mandated to be deterministic in AE.

AE SYNTAX. An AE scheme is a 3-tuple of algorithms $\mathsf{SE} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$, where

1. Gen is the randomized key generation algorithm that returns a bit string K from a finite key-space $\mathcal{K}$ (which is typically $\{0,1\}^k$ for some $k \in \mathbb{N}$).
2. $\mathsf{Enc} : \mathcal{K} \times \mathcal{M} \to \mathcal{C}$ is the deterministic encryption algorithm which takes as input a key K, a message M from message space $\mathcal{M}$ and returns a ciphertext C from some ciphertext-space $\mathcal{C}$. We write this as $C \leftarrow \mathsf{Enc}(K, M)$.
3. The deterministic decryption algorithm $\mathsf{Dec} : \mathcal{K} \times \mathcal{C} \to \mathcal{M} \cup \{\perp\}$ takes a key K and a ciphertext C, and returns either a plaintext M or the failure symbol $\perp \notin \mathcal{M}$.

Correctness requires that $\mathsf{Dec}(K, \mathsf{Enc}(K, M)) = M$ for all values of the inputs above.

AE SECURITY. For symmetric encryption we were content with achieving CCA security. For an authenticated encryption scheme, however, there is another guarantee that we need to define, *ciphertext integrity* (CI). If we can show a scheme to be *both* CCA secure, and CI secure, then we infer that the scheme is AE secure.

CIPHERTEXT INTEGRITY. In a CI game a challenge key K is chosen at random. An adversary $\mathcal{A}$ queries a message M to the challenger and receives encryptions of the message $C \twoheadleftarrow \textsc{Enc}(K, M)$. $\mathcal{A}$ can make up to q such queries, after which it outputs a candidate ciphertext C' that is different from all Cs returned by the challenger. We say that $\mathcal{A}$ wins the game if C' is a valid ciphertext and $\textsc{Dec}(K, C') \neq \perp$.

Given CCA and CI we say the AE advantage of an adversary $\mathcal{A}$ against an authenticated encryption scheme SE is defined by

$$\mathbf{Adv}^{\text{cca+ci}}_{\text{SE}}(\mathcal{A}) := 2 \cdot \Pr\left[\text{AE}^{\mathcal{A}}_{\text{SE}}\right] - 1 \,.$$

2.6.2 *Nonce-Based Authenticated Encryption*

With blockciphers we ran into trouble sometimes because the enciphering algorithm was mandated to be deterministic. We introduced symmetric encryption to overcome this (amongst other issues) by defining a randomised encryption algorithm. Now, in the authenticated setting, because of how authentication is formed, we are forced back to using deterministic encryption algorithms. A way to solve this issue is to introduce a user-specified nonce, that acts as an input to Enc and has the responsibility of flipping the random coins during the encryption process. The syntax is similar to AE, but introduces a nonce N from a nonce space $\mathcal{N}$. For completeness, we describe the syntax below.

NONCE BASED AE SYNTAX. An nonce-based AE scheme is a 3-tuple of algorithms $\text{SE} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$, where

1. Gen is the randomized key generation algorithm that returns a bit string K from a finite key-space $\mathcal{K}$ (which is typically $\{0,1\}^k$ for some $k \in \mathbb{N}$).
2. $\mathsf{Enc} : \mathcal{K} \times \mathcal{M} \times \mathcal{N} \to \mathcal{C}$ is the deterministic encryption algorithm which takes as input a key K, a message M, and a nonce N and returns a ciphertext C. We write this as $C \leftarrow \mathsf{Enc}(K, M, N)$.
3. The deterministic decryption algorithm $\mathsf{Dec} : \mathcal{K} \times \mathcal{C} \times N \to \mathcal{M} \cup \{\perp\}$ takes a key K, a ciphertext C, and a nonce N, and returns either a plaintext M or the failure symbol $\perp \notin \mathcal{M}$.

Correctness requires that $\mathsf{Dec}(K, \mathsf{Enc}(K, M, N), N) = M$ for all values of the inputs above.

2.6.3 *Authenticated Encryption with Associated Data*

With SE we got confidentiality, and with AE we introduced integrity. Authenticated Encryption with Associated Data (AEAD) addresses the third pillar, *authenticity*. It involves the addition of associated data (known as header data) that is bound to the ciphertext and to the environment.

AEAD SYNTAX. An AEAD scheme is a 3-tuple of algorithms $\mathrm{SE} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$, where $\mathcal{K}$, $\mathcal{M}$, and $\mathcal{C}$ are defined as above, and

1. Gen is the randomized key generation algorithm that returns a bit string K.
2. $\mathsf{Enc} : \mathcal{K} \times \mathcal{M} \times \mathcal{N} \times \mathcal{H} \to \mathcal{C}$ is the deterministic encryption algorithm which takes as input a key K, a message M, a nonce N from nonce space $\mathcal{N}$, and possibly some associated header data H from header-space $\mathcal{H}$ and returns a ciphertext C. We write this as $C \leftarrow \mathsf{Enc}(K, M, N, H)$.
3. The deterministic decryption algorithm $\mathsf{Dec} : \mathcal{K} \times \mathcal{C} \times \mathcal{N} \times \mathcal{H} \to \mathcal{M} \cup \{\perp\}$ takes a key K, a ciphertext C, a nonce N, and possibly some associated header data H and returns either a plaintext M or the failure symbol $\perp \notin \mathcal{M}$.

Correctness requires that $\mathsf{Dec}(K, \mathsf{Enc}(K, M, N, H), N, H) = M$ for all values of the inputs above. Note that K, N, and H, must be the same in Enc and Dec for decryption to work.

AEAD SECURITY. We define the AEAD-security of an authenticated encryption scheme $\mathrm{SE} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ by considering the game described in the middle in Fig. 2.4. The AEAD advantage of an adversary $\mathcal{A}$ against SE is defined by

$$\mathbf{Adv}^{\mathrm{aead}}_{\mathrm{SE}}(\mathcal{A}) := 2 \cdot \Pr\left[\mathrm{AEAD}^{\mathcal{A}}_{\mathrm{SE}}\right] - 1 \ .$$

It is required that $\mathcal{A}$ is nonce-respecting in that it does not repeat nonces in its *encryption* queries (but it may repeat them in decryption or across encryption and decryption queries).

MISUSE-RESISTANT AUTHENTICATED ENCRYPTION (MRAE). We define the MRAE-security of an authenticated encryption scheme $\mathrm{SE} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ by considering the rightmost game described in Fig. 2.4. The MRAE advantage of an adversary $\mathcal{A}$ against SE is defined by

$$\mathbf{Adv}^{\mathrm{mrae}}_{\mathrm{SE}}(\mathcal{A}) := 2 \cdot \Pr\left[\mathrm{MRAE}^{\mathcal{A}}_{\mathrm{SE}}\right] - 1 \ .$$

However, in this case, unlike AEAD, $\mathcal{A}$ is not required to be nonce-respecting and can repeat nonces in all queries.

It is clear to see that the syntax and the correctness of AE and its variants are *very* similar. Sometimes it is difficult to see the subtle differences between the notions. This is where the beauty of game based definitions comes in. Figure 2.4 outlines the RR indistinguishability games for the three variations.

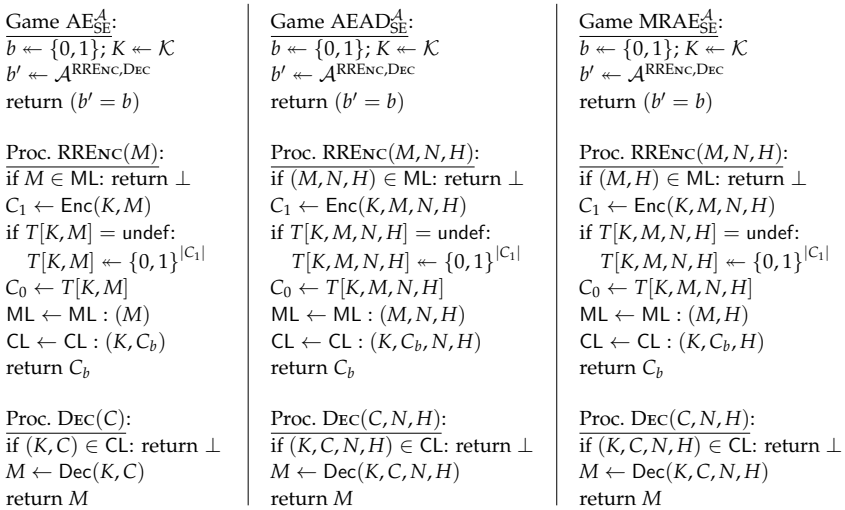


Figure 2.4 – Real-or-random indistinguishability games for an adversary playing against AE (left), nonce-based AEAD (middle), and MRAE (right).

2.7 PUBLIC-KEY CRYPTOGRAPHY

It is interesting to note that unlike English where the term 'Public-Key' pertains to both the concept *of public-key encryption, and the public key itself. In (correct) french, a distinction is made between the concept 'la cryptographie à clé révélée' and the the keys themselves 'la clé publique' and 'la clé secrète'.*

Up until now, we have discussed symmetric encryption where communicating parties share a common key. Symmetric encryption is very efficient, and very secure, but its major limitation is that it does not allow to distribute keys over insecure channels that allow eavesdropping. In an attempt to solve this, public-key cryptography was invented in the 1970's and redefined how we thought about keys. Schemes were designed to perform key exchange over an insecure channel, allowing parties to jointly compute a symmetric key. Further developments led to schemes producing *two* keys; one public, one private, such that the public key could be widely distributed and used by anyone to encrypt a message, but only the private key, kept secret, could be used to decrypt.

2.7.1 *Key Exchange*

When starting to learn cryptography, this is where the subject really feels like magic

The first key exchange protocol was described in [DH76] and was based on the DLP (Section 2.2) To exchange a key, two parties agree on public parameters p and g. The first party generates a random x and computes $w = g^x \bmod p$ which they send to the second party. By the DLP it is assumed that the second party, nor an eavesdropper, can infer the value x. The second party generates a random y, computes $z = g^y \bmod p$ and sends to the first party. After these two exchanges the first party knows x and $z = g^y \bmod p$. They can then compute the exponentiation of z by x to get $z^x = (g^y)^x \bmod p = g^{yx} \bmod p$. The second party knows y, and $w = g^x \bmod p$, and can similarly compute $w^y = g^{xy} = g^{yx} = z^x \bmod p$. They have agreed on a common value. The eavesdropper only knows $g^x \bmod p$ and $g^y \bmod p$. Without

knowing x and y, there is no known efficient way for an eavesdropper to compute the shared secret, which can therefore be used to derive a symmetric key.

2.7.2 *Public Key Encryption*

In addition to key exchange, Diffie and Hellman's *New Directions in Cryptography* also introduced interactive *public-key encryption* (PKE). The first (published) non-interactive public-key encryption scheme, known as RSA was introduced by Rivest, Shamir, and Adelman [RSA78] and remains the most widely used PKE today, based on the hardness of the RSA problem. Although they are very different in spirit, syntactically, SE and PKE are not all that different. The main difference, naturally, is the addition of a public key.

PKE SYNTAX. A public-key encryption scheme is a 3-tuple of algorithms $\mathsf{PKE} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ where

1. Gen is the randomised key generation algorithm that outputs a pair $(\mathsf{pk}, \mathsf{sk}) \twoheadleftarrow \mathsf{Gen}$ where pk is the public key, and sk is the secret key.
2. Enc is the randomised encryption algorithm that takes as input a public key pk, a message M from some message space $\mathcal{M}$, and outputs a ciphertext $C \twoheadleftarrow \mathsf{Enc}(\mathsf{pk}, M)$ where C belongs to a ciphertext space $\mathcal{C}$.
3. Dec is the deterministic decryption algorithm which takes as input a secret key sk, a ciphertext C, and returns a message $M \leftarrow \mathsf{Dec}(\mathsf{sk}, C)$.

PKE CORRECTNESS. It is required that for all possible outputs $(\mathsf{pk}, \mathsf{sk})$ from Gen, and messages M, $\mathsf{Dec}(\mathsf{sk}, \mathsf{Enc}(\mathsf{pk}, M)) = M$.

PKE SECURITY. In the public-key setting, CPA security is somewhat meaningless, as such, as in the symmetric setting, we focus on the CCA security notion. It is arguably more important in the public-key setting as the receiver can receive encrypted messages from many, and possibly unknown sources.

CCA SECURITY IN THE PUBLIC-KEY SETTING. Playing in the real-or-random CCA indistinguishability game an adversary $\mathcal{A}$ interacts with the PKE environment to try to distinguish whether a real or random message was encrypted. Gen is run to produce a pair of keys $(\mathsf{pk}, \mathsf{sk})$. The adversary $\mathcal{A}$ is given access to the public key pk and to the decryption oracle DEC and outputs a message M. A uniform bit b is chosen, and a ciphertext $C \leftarrow \textsc{Enc}(\mathsf{pk}, M)$ is generated. Depending on whether $b = 0$ or $b = 1$, ENC either encrypts the message sent by $\mathcal{A}$, or uniformly selects a message of the same length as M and encrypts it. The ciphertext C is sent to $\mathcal{A}$. The adversary can then continue to interact with the decryption oracle in this manner, with the restriction

that decryptions of C may not be asked. Eventually, $\mathcal{A}$ outputs a bit b' and wins if $b' = b$. The CCA advantage of $\mathcal{A}$ against PKE in the RR game is defined by

$$\mathbf{Adv}^{\text{rr-cca}}_{\text{PKE}}(\mathcal{A}) = 2 \cdot \Pr[b' = b] - 1.$$

2.7.3 *Key Encapsulation Mechanisms*

The advent of public-key cryptography solved the majority of key distribution problems. The first, through key exchange protocols, another, through public-key encryption, but there's a third, and it's sweet. We talked a lot about symmetric cryptography and how it gave nice security guarantees, and is very efficient. A way to overcome the private-key distribution problem is to use a key-encapsulation mechanism (KEM) to encrypt a symmetric key, in a public-key setting. This way, only the intended recipient, i. e. the one with the private key corresponding to the public key, can decrypt to reveal the symmetric secret key. This has come to be known as *hybrid* encryption.

KEY-ENCAPSULATION MECHANISM. A key-encapsulation mechanism (KEM) is a 3-tuple of algorithms $(\mathsf{Gen}, \mathsf{Encaps}, \mathsf{Decaps})$ such that:

1. Gen is the randomised key generation algorithm that outputs a pair $(\mathsf{pk}, \mathsf{sk}) \twoheadleftarrow \mathsf{Gen}$ where pk is the public key, and sk is the secret key. We assume pk and sk each have length at least k, and that k can be determined from pk.
2. The encapsulation algorithm Encaps takes as input a public key pk and outputs a ciphertext C and a key $K \in \{0,1\}^{\ell(k)}$ where ℓ is the key length. We write this as $(C, K) \twoheadleftarrow \mathsf{Encaps}(\mathsf{pk})$.
3. Decaps is the deterministic decapsulation algorithm, which takes as input a private key sk and a ciphertext C, and outputs a key K or a special error symbol $\perp$. We write this as $K \leftarrow \mathsf{Decaps}(\mathsf{sk}, C)$.

KEM SECURITY. As with SE and PKE, the standard notion of security for KEM is that of CCA security. Again, playing in the CCA indistinguishability game, with an adversary $\mathcal{A}$ interacting with the KEM environment. Gen is run to produce a pair of keys $(\mathsf{pk}, \mathsf{sk})$. Then $\mathsf{Encaps}(\mathsf{pk})$ is run to obtain a pair (C, K). A bit b is chosen, and depending on whether $b = 0$ or $b = 1$ we set $K' = K$ or choose a uniform $K' \in \{0,1\}^n$. The adversary $\mathcal{A}$ is given (pk, C, K') and access to the decapsulation oracle $\mathsf{Decaps}(\mathsf{sk})$. The adversary can then continue to interact with Decaps in this manner, with the restriction that decapsulations of C may not be asked. Eventually, $\mathcal{A}$ outputs a bit b' and wins if $b' = b$. The CCA advantage of $\mathcal{A}$ against the KEM game is defined by

$$\mathbf{Adv}^{\text{cca}}_{\mathsf{KEM}}(\mathcal{A}) = 2 \cdot \Pr[b' = b] - 1.$$

Public-key encryption and key-encapsulation mechanisms appear mostly in Chapters 7 and 8

2.8 DIGITAL SIGNATURES

We saw in the case of symmetric encryption that there was the need for more than message confidentiality. When introducing authenticated encryption we saw the need to achieve message integrity. A similar situation arises in public-key cryptography, and is solved by the creation of digital signatures. Yes, you guessed it, they were also introduced in the 1976 paper by Diffie and Hellman [DH76]. A digital signature scheme is simply a method to verify the authenticity of a message. The come in various flavours, and are based on similar hard problems as public-key encryption schemes; perhaps most notably, the DLP and integer factorisation problems.

When they wrote "We stand today on the brink of a revolution in cryptography", they were not exaggerating!

DIGITAL SIGNATURE SYNTAX. A digital signature scheme is a 3-tuple of algorithms $\Sigma = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Verify})$.

1. Gen is the randomised key generation algorithm that generates a key-pair $(\mathsf{sk}, \mathsf{pk}) \twoheadleftarrow \mathsf{Gen}$. The user then keeps the secret (or signing) key sk, and reveals the public (or verifying) key pk to everyone.
2. Sign is the signing algorithm that takes as input a message M, and a secret key sk, and outputs a signature $\sigma \twoheadleftarrow \mathsf{Sign}(\mathsf{pk}, M)$.
3. Verify is the deterministic verification algorithm takes as input a signature σ, a message M, and a public key pk, and outputs `True` if verification works, and `False` otherwise. $\mathsf{Verify}(\mathsf{pk}, M, \sigma) =$ `True` $\vee$ `False`.

Correctness requires that $\mathsf{Verify}(\mathsf{pk}, M, \mathsf{Sign}(\mathsf{sk}, M)) =$ `True` for all possible pairs $(\mathsf{pk}, \mathsf{sk})$ generated by Gen and all valid messages M.

SIGNATURE SECURITY. In the symmetric setting, we introduced the notion of ciphertext integrity to authenticate our encryption scheme. With digital signatures, we have a similar concern. We can also think of the analogy with a handwritten signature where the greatest 'security' issue is that they can potentially be forged. In the digital setting a successful forgery is a signature that is accepted by the verifier, while being generated by an adversary, without using the secret key. Since signatures are by nature public, adversaries can generally access a vast quantity of message-signature pairs; in some scenarios they can even ask their targets to sign well-crafted documents. To guarantee security, we require that even despite having access to all this information, an adversary should not be able to produce forgeries.

There is also the possibility that the secret key could be somehow extracted and then used, which constitutes a total break.

UNFORGABILITY. We hope to achieve a notion that provides assurance that there exists no forgery under a chosen message attack. The notion has become known as 'existential unforgability', and we denote it by EUF-CMA. Given a digital signature scheme $\Sigma = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Verify})$, an an adversary $\mathcal{A}$ playing the EUF-CMA game aims to come up with a pair (M, σ) such that $\mathsf{Verify}(\mathsf{pk}, M, \sigma) = 1$. In the EUF-CMA game,

Game $\text{EUF-CMA}_{\Sigma}^{\mathcal{A}}$:
$(\mathsf{sk}, \mathsf{pk}) \twoheadleftarrow \mathsf{Gen}$
$(M^*, \sigma^*) \leftarrow \mathcal{A}^{\mathsf{Sign},\mathsf{Verify},\mathrm{H}}$
if $(M^*, \sigma^*) \notin \mathsf{L}$
    return $\mathsf{Verify}(\mathsf{pk}, M^*)$
return $\bot$

$\mathsf{Sign}(M)$:
$\sigma \twoheadleftarrow \mathsf{Sign}(\mathsf{sk}, M)$
$\mathsf{L} \leftarrow \mathsf{L} \cup \{M, \sigma\}$
return σ

$\mathsf{Verify}(M, \sigma)$:
return $\mathsf{Verify}(\mathsf{pk}, M, \sigma)$

Figure 2.5 – The EUF-CMA experiment for a digital signature scheme.

Gen is run to obtain public-private key pair $(\mathsf{pk}, \mathsf{sk})$. $\mathcal{A}$ is given pk, and access to a signing oracle $\mathsf{Sign}(\mathsf{sk})$. After some time, $\mathcal{A}$ outputs a candidate message and signature pair (M, σ). Let ML denote the list of all $\mathcal{A}$s queries to the signing oracle. We say that $\mathcal{A}$ succeeds if two conditions are met:

1. $\mathsf{Verify}(\mathsf{pk}, M, \sigma) = \mathtt{True}$, and
2. $M \notin \mathsf{ML}$

A signature scheme Σ is *secure against existential forgeries in a chosen-message attack* (strongly EUF-CMA-secure) if the advantage of any PPT adversary $\mathcal{A}$ against the EUF-CMA game defined in Figure 9.1 is negligible:

$$\mathbf{Adv}_{\Sigma,\mathcal{A}}^{\text{euf-cma}} = \Pr\left[\mathsf{EF}_{\Sigma}^{\mathcal{A}}(\lambda) = 1\right] \in \mathrm{Negl}$$

2.8.1 *Concrete Signature Schemes*

DLP-Based Signatures. ElGamal introduced in 1984 the first DLP-based signature scheme [ElG84]. It inspired the more efficient Schnorr signature scheme [Sch90], for which a security proof is known in the random oracle model assuming the hardness of DLP [PS00]. Schnorr's scheme was initially protected by patents, which had the effect of limiting its diffusion. All these schemes can be implemented on elliptic curves (EC), to decrease parameter size and improve performance. The patent surrounding Schnorr signatures expired in 2008, and since then, they have been gaining popularity.

Schnorr Signatures. Schnorr signatures are provably secure in the Random Oracle Model under the assumed hardness of solving generic DLP instances and work in the following way. A cyclic group $\mathbb{G} = \mathbb{Z}_p$ of prime order p is chosen, in which it is assumed that the DLP is hard, along with a generator $g \in \mathbb{G}$. A hash function $\mathrm{H} : \mathbb{G} \times \{0,1\}^k \to \mathbb{Z}_p$ is chosen. A Schnorr signature scheme is a 3-tuple of algorithms that work as follows:

— $\mathsf{Gen}(g)$ takes as input the generator g of $\mathbb{G}$ and chooses a (secret) signing key $\mathsf{sk} \twoheadleftarrow \mathbb{Z}_p$ and a (public) verification key $\mathsf{pk} \leftarrow g^{\mathsf{sk}}$.

— $\mathsf{Sign}(\mathsf{sk}, M)$ takes as input the signing key sk and a message $M \in \{0,1\}^k$. It chooses a random integer $r \xleftarrow{} \mathbb{Z}_p$, sets $R \leftarrow g^r$, and $C \leftarrow \mathrm{H}(R, M)$, and computes $y \leftarrow \mathsf{sk} \cdot C + r \bmod p$. Finally, it outputs a signature $\sigma = (R, y)$.
— $\mathsf{Verify}(\mathsf{pk}, M, (R, y))$: Computes $e \leftarrow \mathrm{H}(M, R)$ and returns `True` if $g^y = \mathsf{pk}^e \cdot R$, and `False` otherwise.

Schnorr signatures form the basis for the work contained in Part IV.

Part II

TRY AGAIN

Abstract

We study the security of symmetric primitives against *key-correlated attacks* (KCA), whereby an adversary can arbitrarily correlate keys, messages, ciphertexts, and possibly nonces and associated data. Security against KCA is required whenever a primitive should securely encrypt key-dependent data, even when used under related keys. KCA is a strengthening of the previously considered notions of related-key attack (RKA) and key-dependent message (KDM) security. This strengthening is strict, as we show that 2-round Even–Mansour fails to be KCA secure even though it is *both* RKA and KDM secure. We provide feasibility results in the ideal-cipher model for KCAs and show that 3-round Even–Mansour is KCA secure under key offsets in the random-permutation model. We also give a natural transform that converts any authenticated encryption scheme to a KCA-secure one in the random-oracle model. Conceptually, these results allow for a unified treatment of RKA and KDM security in idealised models of computation.

This is joint work with Pooya Farshim and Georg Fuchsbauer.

3

KEY CORRELATED SECURITY

Cryptographic algorithms are subject to a multitude of threats. Many of these threats are accounted for in the theoretical security analysis carried out by cryptographers, but not all. For example, early on, the seminal paper of Goldwasser and Micali [GM84] pointed out that guarantees of semantic security may break down if the adversary sees encryptions of the secret key. Formal analyses of protocols can also become moot [Bih94b; Bih94a] when the assumption that cryptosystems are run on independently generated keys no longer holds. A number of works have analyzed the security of cryptosystems in the presence of key-dependent messages or when different keys are generated in dependent ways (see the related work section below). We continue this line of work and ask to what extent basic cryptosystems (such as blockciphers and symmetric encryption) can resist attacks that exploit correlated inputs.

3.1 MOTIVATION

Our motivation for studying correlated-input security is twofold. We are interested in settings where a cryptosystem may be run on related keys—either by design or due to attacks—to securely encrypt messages that depend on the encryption key. Suppose a user stores a secret key K on its hard drive. An adversary may be able to tamper with this key, for example flip some of its bits and change it to $K \oplus \Delta$ for some bit string Δ. It may then obtain a full-disk encryption under this key. It is not clear what security assertions can be made, as this setting falls outside both the related-key attack (RKA) and the key-dependent message (KDM) models. Indeed, the RKA model only allows the adversary to obtain encryptions of the form $\mathsf{Enc}(\phi(K), M)$, for functions ϕ mapping keys to keys, but for key-independent messages, while KDM accounts for key-dependent encryptions of type $\mathsf{Enc}(K, \psi(K))$, for functions ψ mapping keys to messages, but under untampered keys. In the described attack, the adversary obtains $\mathsf{Enc}(K \oplus \Delta, K \oplus \Delta)$. This is not covered by either of these models since both the key and the message are correlated with the original key. Other applications of KCAs include efficient garbling of XORs [App16], where KCA security (called RK-KDM there) with respect to linear functions or the form $\alpha \cdot K \oplus \Delta$ for a bit α are used.

These settings require a stronger security notion, which is what we introduce here. The Key-Correlated Attacks (KCAs) model lets the adversary obtain encryptions of key-dependent messages under related

keys. Generally, wherever there is a possibility of both RKAs and KDM attacks, i.e., key-correlated encryptions of the form $\mathsf{Enc}(\phi(K), \psi(K))$, there is good chance that the actual security needed is KCA security. A typical use case is when the round functions of a block cipher are keyed via related keys, and the construction is used to encrypt key-dependent data.

In our model, for generality, simplicity and strength, we symmetrically allow for key-dependent *ciphertexts*, that is, the adversary can see $\mathsf{Dec}(\phi(K), \psi(K))$. Such settings arise when the decryption algorithm of a blockcipher is run during encryption, which is for example the case in the triple DES construction [BR06], the ElmD construction [BDMN16], or in amplification theorems for blockciphers [MP04; CPS14].

Our second motivation is conceptual in that KCA provides a unified approach to RKA and KDM security analyses of symmetric primitives. More concretely, our goal is to prove KCA feasibility theorems and then derive RKA and KDM security as simple corollaries. This allows for reuse of security proofs and identifies classes of permitted attacks more generally, while leading to stronger security results.

3.2 RELATED WORK

RKA SECURITY. Knudsen and Biham [Knu93; Bih94b; Bih94a] initiated the study of RKAs and Bellare and Kohno [BK03] gave a theoretical treatment. High-profile RKAs on AES were discovered by Biryukov et al. [BKN09; BK09]. The RKA model was extended by Albrecht et al. [AFPW11] to account for attacks that depend on an ideal primitive [**Har09**; **Ber10**]. The RKA security of Feistel networks [BF15] and Even–Mansour ciphers [FP15; CS15] have been studied. Bellare, Cash, and Miller [BCM11] present a comprehensive treatment of RKA security for various cryptographic primitives.

KDM SECURITY. Goldwasser and Micali [GM84] already hinted at the need for KDM security. The first use of KDM security appears in the work of Camenisch and Lyskanskaya [CL01] for anonymous credentials. Black, Rogaway, and Shrimpton [BRS03] formulated KDM security for symmetric encryption and proved its feasibility in the random-oracle model. Halevi and Krawczyk [HK07] later gave feasibility results in the standard model. Bellare and Keelveedhi [BK11] studied KDM in the context of authenticated encryption. Bellare, Cash, and Keelveedhi [BCK11] give a generic construction of a tweakable blockcipher from a blockcipher which is KDM secure. More recently, Farshim, Khati, and Vergnaud [FKV17] studied KDM security for the ideal cipher and the iterated Even–Mansour constructions. In the asymmetric setting the first feasibility result in the standard model for rich classes of functions was by Boneh et al. [BHHO08]. Camenisch,

Chandran and Shoup [CCS09] gave a KDM-CCA secure public-key encryption scheme.

CORRELATED INPUTS. Study of security under correlated inputs goes back to the work of Ishai et al. [IKNP03] as correlation-robustness in the design of oblivious transfer protocols. Correlated-input security was made explicit for hash functions by Goyal, O'Neill, and Rao [GOR11], who show relations with related-key attacks. The work of Böhl, Davies and Hofheinz [BDH14] considers related-key attacks in the presence of key-dependent messages. Their RKA-KDM security could be considered a natural analogue of our model for public-key encryption. They construct schemes that achieve their notion based on number-theoretic assumptions such as DDH, LWE, QR, or DCR. Applebaum [App16] gives an RKA-KDM symmetric encryption scheme based on the LPN assumption.

3.3 CONTRIBUTIONS

Building on the above line of works, we formulate a new security model incorporating and strengthening both the RKA and KDM models. We speak of key-correlated attack (KCA) in this context, a name that is loosely inspired by the introduction of correlated-input attacks against hash functions [GOR11] (note that the notion of *key-dependent input attacks* has already been used by Halevi and Krawczyk [HK07]). We give appropriate definitions of security under key-correlated attacks that relate well to the standard RKA and KDM security notions. Our definition extends that in [App16] for randomized symmetric encryption under chosen-plaintext attacks to the setting of authenticated encryption with associated data (AEAD).

We start with comparing our notion to existing ones. After proving that KCA implies RKA and KDM security, we show that KCA security is strictly stronger than even simultaneously having RKA and KDM security. We give a natural separation by demonstrating a KCA attack on the 2-round Even–Mansour cipher, which was shown to satisfy both RKA and KDM security in two previous works [FP15; FKV17].

After defining KCA and showing a separation result, we study feasibility of KCAs. Our starting point is the ideal-cipher model, in which all parties have oracle access to a keyed random permutation in both directions. We cannot allow arbitrary dependencies of keys and messages as otherwise "trivial" attacks, which work against any scheme, arise. To exclude these and thus obtain a meaningful notion, we restrict the classes of allowed dependencies. We show that if they satisfy appropriate notions of *key-unpredictability* and *claw-freeness* then the ideal cipher satisfies KCA security. Roughly speaking, key-unpredictability requires that the adversary does not obtain encryptions or decryption under predictable keys. Claw-freeness, on the other hand, requires that the inputs are distinct, and so repetition

pattern of the outputs cannot be exploited. Analogues of these notions were previously considered in the RKA and KDM settings.

In our setting we require a third condition, which we call *cross-key-claw detectability*, that allows us to deal with claws across encryption and decryption queries. This notion is sufficiently weak so that in the RKA and KDM setting it automatically follows from claw-freeness. In the KCA setting, however, it does not necessarily, as it restricts claws on keys.[1] Two results on the RKA and KDM security of the ideal cipher by Bellare and Kohno [BK03] and Farshim, Khati, Vergnaud [FKV17] respectively, fall out as natural corollaries of our theorem.

Turning to concrete constructions, we analyze the KCA security of the iterated Even–Mansour cipher with *three* rounds in the random-permutation model. We show that with reuse of keys (which is known to be necessary for RKA security [FP15]) and using different permutations (which is necessary for KDM security whenever keys are reused [BW99; FKV17]) we can provably achieve security against key-correlated attacks that concurrently encrypt messages M or offsets of key of the form $K \oplus \Delta_2$ under other offsets of the key of the form $K \oplus \Delta_1$. This strengthens two feasibility results due to Farshim, Khati, Vergnaud [FKV17] and Farshim and Procter [FP15].

From a technical point of view, the novelty of our KCA proof for 3-round Even–Mansour is that we keep the outer permutations partially consistent with the replaced forgetful oracles as well as the permutation oracles. For legal queries we show this can be done with overwhelming probability, while a detection algorithm will allow us to identify illegal queries and reject them. This proof thus deviates from previous works in which oracles are fully decoupled. As a result we also obtain a different (albeit somewhat more complex) way to prove the RKA security of 3-round Even–Mansour against key offsets by replacing the outer (rather the inner) permutation [FP15].

We end this part by showing how to generically transform any AE-secure AEAD scheme to one which is a KCA-secure in the random-oracle model by hashing the key with nonces. For this result we only require the set of allowed functions to be unpredictable, as nonces automatically prevent repetitions due to claws in the functions. In contrast to previous work by Bellare and Keelveedhi [BK11] on similar transforms for achieving KDM-security, our scheme is secure with *key-dependent nonces and headers*. Although key-dependent headers are briefly discussed in [BK11], security with respect to key-dependent nonces is not considered at all. Arguably, however, there is a stronger case for the key-dependency of nonces than of headers: when nonces are randomly chosen they might become correlated with the key due to, e.g., bad generation of random numbers. For key-dependent

1. Claw-freeness can be modified to key-claw-freeness across encryption and decryption so that cross-key-claw-freeness is automatic. But in this case the reach of our feasibility results do not extend to the KDM setting since under KDM keys always collide.

headers, Bellare and Keelveedhi give a negative result, by having the adversary exploit the pattern of decryption errors (either $\perp$ for an illegal query or 0 for failure in authenticity) to recover the key. In our setting, however, the decryption oracle only returns a single error symbol, which enables security under key-dependent inputs. If our model were modified to also have distinct error symbols, an attack similar to that in [BK11] would arise. We note that in these settings one might be able to obtain non-trivial feasibility results by requiring a form claw-freeness. [2]

WHERE TO GO FROM HERE. In Chapter 4, we define KCA security for blockciphers and study its relation to RKA and KDM security before showing two separation results in Section 7.3. In Chapter 5 we study KCA in the ideal-cipher model and we prove 3-round Even–Mansour KC-CCA secure for offsets. Chapter 6 contains our generic construction of a KCA-secure authenticated encryption scheme from any AE-secure one.

2. Consistently, the attack in [BK11] exploits claws in the key-dependent headers.

4

THE MODEL

4.1 CONCEPT AND DEFINITIONS

CORRELATION-DERIVATION FUNCTION (CDF). A correlation-derivation function (CDF) is a circuit of the form

$$\xi \colon \mathcal{K} \longrightarrow \mathcal{K} \times \mathcal{M} \ .$$

A set of such functions is called a CDF set. Throughout the paper we denote CDF sets by Ξ and require that membership of a CDF set can be efficiently decided. We will define key-correlated security primitives relative to two CDF sets Ξ^e and Ξ^d that describe allowed encryption and decryption queries, respectively.

KCA-SECURE BLOCKCIPHERS. Let BC be a blockcipher with key space $\mathcal{K} = \{0,1\}^k$ and message (and ciphertext) space $\mathcal{M} = \{0,1\}^n$. Let Ξ^e and Ξ^d be CDF sets for the input space $\mathcal{K} \times \mathcal{M}$. The KC-CCA advantage of an adversary $\mathcal{A}$ against BC is defined by

$$\mathbf{Adv}^{\text{kc-cca}}_{\text{BC}}(\Xi^e, \Xi^d, \mathcal{A}) := 2 \cdot \Pr\left[\text{KC-CCA}^{\Xi^e,\Xi^d,\mathcal{A}}_{\text{BC}}\right] - 1 \ , \qquad (4.1)$$

where game $\text{KC-CCA}^{\Xi^e,\Xi^d,\mathcal{A}}_{\text{BC}}$ is given in Figure 4.1. In this game, the adversary's goal is to decide whether (case $b = 1$) its oracles are using the blockcipher $\text{BC} = (\mathsf{E}, \mathsf{D})$ or (case $b = 0$, the *ideal* case) a random permutation iE and its inverse iD. Lists ML and CL prevent the adversary from trivially winning the game. Otherwise the adversary could, for instance, recover the challenge key K^* by querying an encryption (or decryption) of K^* and subsequently ask for decryption (or encryption) of the oracle's reply.

We note that for two sets $\Xi^e_1 \subseteq \Xi^e_2$ and $\Xi^d_1 \subseteq \Xi^d_2$ security against (Ξ^e_2, Ξ^d_2)-KC-CCA implies security against (Ξ^e_1, Ξ^d_1)-KC-CCA. Whenever $\Xi^d = \emptyset$ we obtain a chosen-plaintext attack model.

NOTE. Analogues of KCA security can be formulated for hash functions and pseudorandom generators, which become equivalent to correlated-input security for hash functions [GOR11] and RKA security for PRGs [BCM11].

4.2 EXAMPLES

As examples of KC functions, suppose that related keys, described by functions from some set Φ, are used within the specification of an encryption scheme (an example are the 3GPP protocols [IK04]). Suppose further that the scheme is used to encrypt messages that depend

Game KC-CCA$^{\Xi^e,\Xi^d,\mathcal{A}}_{\mathsf{BC}}$:	Proc. KCEnc(ξ^e):	Proc. KCDec(ξ^d):
$b \twoheadleftarrow \{0,1\}$	$(K,M) \leftarrow \xi^e(K^*)$	$(K,C) \leftarrow \xi^d(K^*)$
$(\mathsf{iE},\mathsf{iD}) \twoheadleftarrow \mathsf{Block}(\mathcal{K},\mathcal{M})$	if $(K,M) \in \mathsf{ML}$: return $\perp$	if $(K,C) \in \mathsf{CL}$: return $\perp$
$K^* \twoheadleftarrow \mathcal{K}$	$C \leftarrow \mathsf{E}(K,M)$	$M \leftarrow \mathsf{D}(K,M)$
$b' \twoheadleftarrow \mathcal{A}^{\text{KCEnc},\text{KCDec}}$	if $b = 0$: $C \leftarrow \mathsf{iE}(K,M)$	if $b = 0$: $M \leftarrow \mathsf{iD}(K,M)$
return $(b' = b)$	$\mathsf{CL} \leftarrow \mathsf{CL} : (K,C)$	$\mathsf{ML} \leftarrow \mathsf{ML} : (K,M)$
	return C	return M

Figure 4.1 – Game defining the (Ξ^e,Ξ^d)-KC-CCA security of a blockcipher $\mathsf{BC} = (\mathsf{E},\mathsf{D})$ with key space $\mathcal{K}$ and message space $\mathcal{M}$. We require that $\xi^e \in \Xi^e$ and $\xi^d \in \Xi^d$ for all queries.

on the key in ways represented by functions from a set Ψ. The overall effect is that an adversary has access to ciphertexts corresponding to *key-dependent messages under related keys*. In other words, the adversary can see key-correlated ciphertexts for functions:

$$\xi\colon K \mapsto (\phi(K),\psi(K)) \text{ with } \phi \in \Phi \text{ and } \psi \in \Psi\ , \quad \text{and}$$
$$\xi\colon K \mapsto (\phi(K),M) \text{ with } \phi \in \Phi \text{ and } M \in \mathcal{M}\ .$$

Alternatively, suppose an adversary has access to encryptions under related keys with respect to Φ through, say, injection of faults that change bits of a hardware-stored encryption key [BDL97; BS97]. [1] If the scheme is used to encrypt key-dependent messages with respect to Ψ, the adversary would be able to launch a KCA for the functions:

$$\xi\colon K \mapsto \big(\phi(K),\psi(\phi(K))\big) \text{ with } \phi \in \Phi \text{ and } \psi \in \Psi\ , \quad \text{and}$$
$$\xi\colon K \mapsto (\phi(K),M) \text{ with } \phi \in \Phi \text{ and } M \in \mathcal{M}\ .$$

The KCA model thus captures, among other things, a variety of *joint* RKA and KDM attacks on a blockcipher.

4.3 RELATION WITH RKA AND KDM

Let id denote the identity function on a key space $\mathcal{K}$ and let Γ denote the set of all constant functions $K \mapsto M$ for $M \in \mathcal{M}$. We identify a pair of functions (ϕ,ψ), with ranges $\mathcal{K}$ and $\mathcal{M}$ respectively, with the correlation-derivation function $K \mapsto (\phi(K),\psi(K))$.

CPA/CCA-only sets. We call a pair (Ξ^e,Ξ^d) CCA-only if $\Xi^e = \Xi^d = \{id\} \times \Gamma$. (The adversary can only make regular encryption and decryption queries.) We call (Ξ^e,Ξ^d) CPA-only if $\Xi^e = \{id\} \times \Gamma$ and $\Xi^d = \emptyset$.

RKA/KDM-only sets. We call a pair (Ξ^e,Ξ^d) RKA-only if $\Xi^e = \Xi^d = \Phi \times \Gamma$ for an RKA set Φ of functions mapping keys to keys. We call (Ξ^e,Ξ^d) KDM-only if $\Xi^e = \{id\} \times \Psi$ and $\Xi^d = \{id\} \times \Gamma$ (i.e., no key-dependent ciphertexts allowed [FKV17]) for a KDM set Ψ of functions mapping keys to messages. (The CPA versions are defined

1. KCA security does not imply security against fault attacks in general.

analogously by demanding that $\Xi^d = \emptyset$.) We assume that any KDM set Ψ contains Γ, as it is natural to always allow for chosen plaintexts.

We show that our definition of KC security extends the standard RKA and KDM definitions for blockciphers, which we recall in Appendix Section 4.5. In particular, for KDM-only sets we have that KCA and KDM security are equivalent and similarly for RKA-only sets KCA and RKA security are equivalent. For two of the four implications, we assume *claw-freeness* of the correlation-derivation functions. This requires that it is hard to find two different functions that have the same output given a random input and is defined formally in Section 5.1 (p. 54). Claw-freeness is required because the KCA game returns $\perp$ whenever claws are detected whereas the RKA game does not. The proof of the following proposition can be found in Section 4.5.

Proposition 4.1. *Let* BC *be a blockcipher. Let Ξ^e and Ξ^d be two CDF sets.*

RKA: *Suppose (Ξ^e, Ξ^d) are RKA-only with $\Xi^e = \Xi^d = \Phi \times \Gamma$. If* BC *is (Ξ^e, Ξ^d)-*KC-CCA*-secure then it is Φ-*RK-CCA *secure. If* BC *is Φ-*RK-CCA*-secure and $\Xi^e = \Xi^d$ is claw-free then* BC *is (Ξ^e, Ξ^d)-*KC-CCA*-secure.*

In particular, let n be the block length, $\mathcal{A}$ be an adversary and q the maximum number of its queries. Then there exists an adversary $\mathcal{B}$ such that

$$\mathbf{Adv}^{\text{rk-cca}}_{\text{BC}}(\Phi, \mathcal{A}) \leq \mathbf{Adv}^{\text{kc-cca}}_{\text{BC}}(\Xi^e, \Xi^d, \mathcal{B}) + q^2/2^n \ .$$

Moreover, there exist adversaries $\mathcal{B}$ and $\mathcal{B}^{\text{cf}}$ such that

$$\mathbf{Adv}^{\text{kc-cca}}_{\text{BC}}(\Xi^e, \Xi^d, \mathcal{A}) \leq q^2 \cdot \left(\mathbf{Adv}^{\text{cf}}_{\Phi}(\mathcal{B}^{\text{cf}}) + 1/2^n\right) + \mathbf{Adv}^{\text{rk-cca}}_{\text{BC}}(\Phi, \mathcal{B}) \ .$$

KDM: *Suppose (Ξ^e, Ξ^d) are KDM-only with $\Xi^e = \{id\} \times \Psi$ and $\Xi^d = \{id\} \times \Gamma$. If* BC *is Ψ-*KDM-CCA *secure then it is (Ξ^e, Ξ^d)-*KC-CCA*-secure. If* BC *is (Ξ^e, Ξ^d)-*KC-CCA*-secure and Ξ^e is claw-free then* BC *is Ψ-*KDM-CCA*-secure.*

In particular, for every adversary $\mathcal{A}$ there exists an adversary $\mathcal{B}$ such that

$$\mathbf{Adv}^{\text{kc-cca}}_{\text{BC}}(\Xi^e, \Xi^d, \mathcal{A}) = \mathbf{Adv}^{\text{kdm-cca}}_{\text{BC}}(\Psi, \mathcal{B}) \ ,$$

and there exist adversaries $\mathcal{B}$ and $\mathcal{B}^{\text{cf}}$ such that

$$\mathbf{Adv}^{\text{kdm-cca}}_{\text{BC}}(\Psi, \mathcal{A}) \leq q^2 \cdot \mathbf{Adv}^{\text{cf}}_{\Psi}(\mathcal{B}^{\text{cf}}) + \mathbf{Adv}^{\text{kc-cca}}_{\text{BC}}(\Xi^e, \Xi^d, \mathcal{B}) \ ,$$

where q is the maximum number of queries adversary $\mathcal{A}$ makes.

Analogous implications hold in the CPA setting.

$\mathsf{E}'(K,M)$:	$\mathsf{D}'(K,C)$:
$\overline{C^* \leftarrow \mathsf{E}(\min(K,\overline{K}),0^n)}$	$\overline{C^* \leftarrow \mathsf{E}(\min(K,\overline{K}),0^n)}$
if $M = K$: return C^*	if $C = C^*$: return K
if $M = \mathsf{D}(K,C^*)$: return $\mathsf{E}(K,K)$	if $C = \mathsf{E}(K,K)$: return $\mathsf{D}(K,C^*)$
return $\mathsf{E}(K,M)$	return $\mathsf{D}(K,C)$

Figure 4.2 – Blockcipher $(\mathsf{E}',\mathsf{D}')$ that is both KDM-secure and RKA-secure secure but not KCA-secure.

4.4 KEY-CORRELATED ATTACKS

In this section we show that even if a blockcipher simultaneously achieves security against RK and KDM attacks, it may still fail to achieve security against concurrent RK/KDM attacks, and hence also fail to achieve KCA security. We provide two separations, one artificial and one natural, to demonstrate this.

4.4.1 *A generic separation result*

Let $\mathcal{K} = M = \{0,1\}^n$ and define the following sets of functions:

$$\begin{aligned}\Phi^{\oplus} &:= \{K \mapsto K \oplus M : M \in \mathcal{M}\}\\ \Psi^{\oplus} &:= \{K \mapsto \alpha \cdot K \oplus M : M \in \mathcal{M}, \alpha \in \{0,1\}\}\\ \Xi^{\oplus} &:= \{K \mapsto (K \oplus M_1, \alpha \cdot K \oplus M_2) : M_1, M_2 \in \mathcal{M}, \alpha \in \{0,1\}\}\ .\end{aligned} \tag{4.2}$$

Given a $\Phi^{\oplus}$-RKA secure and $\Psi^{\oplus}$-KDM secure blockcipher (E,D), consider the modified cipher $(\mathsf{E}',\mathsf{D}')$ shown in Fig. 4.2, where min is the lexicographic minimum and $\overline{K} := K \oplus 1^k$. Note that $(\mathsf{E}',\mathsf{D}')$ is again a blockcipher, i.e., a permutation for each value of the key (we simply swapped $\mathsf{E}(K,K)$ and C^* in the image space).

Note that $\min(K,\overline{K}) = \min(\overline{K},\overline{\overline{K}})$*, thus* $\mathsf{E}'(K,K) = \mathsf{E}'(\overline{K},\overline{K})$.

To see that $(\mathsf{E}',\mathsf{D}')$ is not $(\Xi^{\oplus},\emptyset)$-KC-CCA secure, consider an adversary that queries $K \mapsto (K,K)$ and $K \mapsto (\overline{K},\overline{K}) = (K \oplus 1^n, K \oplus 1^n)$ to its KCEnc oracle. For the modified cipher both queries yield the same result:

$$\mathsf{E}'(\overline{K},\overline{K}) = \mathsf{E}\big(\min(\overline{K},\overline{\overline{K}}),0^n\big) = \mathsf{E}\big(\min(K,\overline{K}),0^n\big) = \mathsf{E}'(K,K)\ ,$$

while for the ideal cipher this would only happen with probability $1/2^n$. On the other hand, the modified cipher $(\mathsf{E}',\mathsf{D}')$ remains both RKA and KDM secure as it essentially behaves like (E,D) when no joint RKA and KDM attacks as above can be mounted. We kept this discussion informal as our second separating example below is more natural and practically relevant.

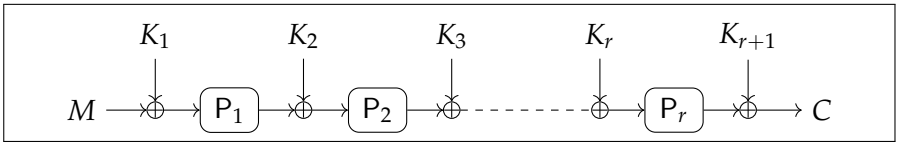


Figure 4.3 – The r-round iterated Even–Mansour cipher.

4.4.2 *Attack on 2-round Even–Mansour*

The r-round Even–Mansour cipher EM is defined by r permutations $\mathsf{P}_1, \ldots, \mathsf{P}_r$, has key space $\mathcal{K} = \{0,1\}^{(r+1)n}$, domain $\mathcal{M} = \{0,1\}^n$ and en- and deciphering algorithms (cf. Fig. 4.3)

$$\mathsf{E}\big((K_1, \ldots, K_{r+1}), M\big) := \mathsf{P}_r\big(\ldots \mathsf{P}_2(\mathsf{P}_1(M \oplus K_1) \oplus K_2) \ldots\big) \oplus K_{r+1} \,,$$
$$\mathsf{D}\big((K_1, \ldots, K_{r+1}), C\big) := \mathsf{P}_1^-\big(\ldots \mathsf{P}_{r-1}^-(\mathsf{P}_r^-(C \oplus K_{r+1}) \oplus K_r) \ldots\big) \oplus K_1 \,.$$

The EM ciphers can also be considered in configurations where (some of the) keys and/or (some of the) permutations are reused in different rounds. We denote the EM cipher where P_i and K_{i+1} are used in round i by $\mathsf{EM}^{\mathsf{P}_1,\ldots,\mathsf{P}_r}[K_1, K_2, \ldots, K_{r+1}]$.

Recall the function sets $\Phi^{\oplus}$, $\Psi^{\oplus}$, and $\Xi^{\oplus}$ from (4.2). The 2-round Even–Mansour cipher with key reuse and independent permutations $\mathsf{E}(K, M) := \mathsf{EM}^{\mathsf{P}_1,\mathsf{P}_2}[K, K, K]$ was shown to be $\Phi^{\oplus}$-RK-CPA secure in [FP15] and $\Psi^{\oplus}$-KDM-CCA secure in [FKV17]. We now show that this construction fails to be $(\Xi^{\oplus}, \emptyset)$-KC-CCA secure. Consider the KDM encryption

$$\mathsf{E}(K, K) = \mathsf{P}_2(\mathsf{P}_1(0^n) \oplus K) \oplus K \,. \tag{4.3}$$

Now let $\Delta^* := \mathsf{P}_1(0^n) \oplus \mathsf{P}_1(1^n)$ and consider the key-correlated encryption

$$\begin{aligned} C_1 := \mathsf{E}(K \oplus \Delta^*, K \oplus \Delta^* \oplus 1^n) &= \mathsf{P}_2(\mathsf{P}_1(1^n) \oplus K \oplus \Delta^*) \oplus K \oplus \Delta^* \\ &= \mathsf{P}_2(\mathsf{P}_1(0^n) \oplus K) \oplus K \oplus \Delta^* \\ &\overset{(4.3)}{=} \mathsf{E}(K, K) \oplus \Delta^* \,. \end{aligned}$$

Thus the two key-correlated ciphertexts C_1 and $\mathsf{E}(K, K)$ differ by a known value Δ^*. For the ideal cipher this event only occurs with probability $1/2^n$, as both ciphertexts would be randomly distributed among at least $2^n - 1$ values. An attacker thus merely needs to make two queries $K \mapsto (K, K)$ and $K \mapsto (K \oplus \Delta^*, K \oplus \Delta^* \oplus 1^n)$ and check whether the answers differ by Δ^*.

4.5 RELATION BETWEEN KCA, RKA, AND KDM

RKA AND KDM SECURITY. Fig. 4.4 gives the definitions of RKA security [BK03] and KDM security [FKV17] for blockciphers. The

(Top) Game defining the Φ-RK-CCA security of $\mathrm{BC} = (\mathsf{E}, \mathsf{D})$. We require that $\phi \in \Phi$ for all queries ϕ. (Bottom) Game defining the Ψ-KDM-CCA security of BC with key space $\mathcal{K}$ and message space $\mathcal{M}$. We require that $\psi \in \Psi$ for all queries ψ.

Game $\mathrm{RK\text{-}CCA}^{\Phi,\mathcal{A}}_{\mathrm{BC}}$:	Proc. $\mathrm{RKEnc}(\phi, M)$:	Proc. $\mathrm{RKDec}(\phi, C)$:
$b \twoheadleftarrow \{0,1\}$; $K^* \twoheadleftarrow \mathcal{K}$	$K \leftarrow \phi(K^*)$	$K \leftarrow \phi(K^*)$
$(\mathsf{iE}, \mathsf{iD}) \twoheadleftarrow \mathrm{Block}(\mathcal{K}, \mathcal{M})$	$C \leftarrow \mathsf{E}(K, M)$	$M \leftarrow \mathsf{D}(K, C)$
$b' \twoheadleftarrow \mathcal{A}^{\mathrm{RKEnc},\mathrm{RKDec}}$	if $b = 0$: $C \leftarrow \mathsf{iE}(K, M)$	if $b = 0$: $M \leftarrow \mathsf{iD}(K, C)$
return $(b' = b)$	return C	return M

Game $\mathrm{KDM\text{-}CCA}^{\Psi,\mathcal{A}}_{\mathrm{BC}}$:	Proc. $\mathrm{KDMEnc}(\psi)$:	Proc. $\mathrm{Dec}(C)$:
$b \twoheadleftarrow \{0,1\}$; $K^* \twoheadleftarrow \mathcal{K}$	$M \leftarrow \psi(K^*)$	if $C \in \mathsf{CL}$: return $\perp$
$(\mathsf{iE}, \mathsf{iD}) \twoheadleftarrow \mathrm{Block}(\mathcal{K}, \mathcal{M})$	$C \leftarrow \mathsf{E}(K^*, M)$	$M \leftarrow \mathsf{D}(K^*, C)$
$b' \twoheadleftarrow \mathcal{A}^{\mathrm{KDMEnc},\mathrm{Dec}}$	if $b = 0$: $C \leftarrow \mathsf{iE}(K^*, M)$	if $b = 0$: $M \leftarrow \mathsf{iD}(K^*, C)$
return $(b' = b)$	$\mathsf{CL} \leftarrow \mathsf{CL} : C$; return C	return M

Figure 4.4 – RKA-, and KDM-security games.

advantage functions are defined in the standard way (as for KCA, Equation (4.1) in Sec. 4). The CPA versions are obtained naturally by disallowing decryption queries.

We now prove that for RKA-only (resp. KDM-only) claw-free sets KCA and RKA (resp. KDM) security are equivalent. For convenience we restate Proposition 4.1 as Propositions 4.2 and 4.5 below.

4.5.1 *KC(RK-only) $\Leftrightarrow$ RK*

Proposition 4.2. *Let* BC *be a blockcipher. Let Ξ^e and Ξ^d be two RKA-only CDF sets with $\Xi^e = \Xi^d = \Phi \times \Gamma$. Then for every adversary $\mathcal{B}$ there exists an adversary $\mathcal{A}$ such that*

$$\mathbf{Adv}^{\text{rk-cca}}_{\mathrm{BC}}(\Phi, \mathcal{B}) \leq \mathbf{Adv}^{\text{kc-cca}}_{\mathrm{BC}}(\Xi^e, \Xi^d, \mathcal{A}) + q^2/2^n \ ,$$

where n is the block length and q is the maximum number of queries $\mathcal{B}$ makes. Moreover, for every adversary $\mathcal{A}$ there exist adversaries $\mathcal{B}$ and $\mathcal{B}^{cf}$ such that

$$\mathbf{Adv}^{\text{kc-cca}}_{\mathrm{BC}}(\Xi^e, \Xi^d, \mathcal{A}) \leq q^2 \cdot (\mathbf{Adv}^{\text{cf}}_{\Phi}(\mathcal{B}^{cf}) + 1/2^n) + \mathbf{Adv}^{\text{rk-cca}}_{\mathrm{BC}}(\Phi, \mathcal{B}) \ ,$$

where q is the maximum number of queries adversary $\mathcal{A}$ makes.

Recall that (Ξ^e, Ξ^d) is RKA-only if $\Xi^e = \Xi^d = \Phi \times \Gamma$ for an RKA set Φ of functions mapping keys to keys (where Γ is the set of constant functions over $\mathcal{M}$). The difference between games RK and KC is that in the former there are no lists ML and CL, thus all queries are allowed. When showing KC$\Rightarrow$RK, the reduction must therefore answer all queries, while its KC challenger might reply with $\perp$. The reduction will therefore keep local lists CL' and ML', which simulate the challenger's blacklists.

The KC challenger, when queried $\mathrm{KCEnc}(\xi^e = (\phi^e, M))$ stores pairs $(K = \phi^e(K^*), C = \mathsf{E}(K, M))$ in its list CL. The reduction does not know the values K and will store pairs (C, M) in a list CL'. In the KC game, queries $\mathrm{KCDec}(\xi^d = (\phi^d, C))$ with $(\phi^d(K^*), C) \in \mathsf{CL}$ are answered with $\perp$, whereas in the RK game, $\mathrm{RKDec}(\phi^d, C)$ always returns $\mathsf{D}(\phi^d(K^*), C)$. To answer such queries, the reduction looks for

an entry (C, M') for some M' in its list CL'. If (1) there is only one such entry, it returns M', which perfectly simulates RKDec: if $(C, M') \in \mathsf{CL}'$ and $\text{KCDec}(\xi^d = (\phi^d, C))$ returned $\perp$ then $(\phi^e(K^*), C) \in \mathsf{CL}$ and if there is only one entry with C, we must have $\phi^d(K^*) = \phi^e(K^*)$. If (2) there are several entries $(C, M_0), (C, M_1) \in \mathsf{CL}'$, the reduction aborts the simulation and returns 1 (that is, it guesses that it is interacting with the actual blockcipher). As we show in the lemma below, we can bound the probability of this type of collision in CL' in case the reduction is interacting with an ideal cipher, and therefore the probability of aborting and returning a wrong guess. The list ML and queries to the KCEnc oracle are dealt with analogously.

Lemma 4.3. *Consider game* $\text{KC-CCA}_{\text{BC}}^{\Xi^e, \Xi^d, \mathcal{A}}$ *with b fixed to 0. Then the probability that there are two entries $(K_0, C), (K_1, C) \in \mathsf{CL}$ with $\mathsf{iD}(K_0, C) \neq \mathsf{iD}(K_1, C)$ is at most $q^2/2^n$, where q is the number of queries made by $\mathcal{A}$. The same holds for the probability of two entries $(K_0, M), (K_1, M) \in \mathsf{ML}$ with $\mathsf{iE}(K_0, M) \neq \mathsf{iE}(K_1, M)$.*

Proof. Let us consider CL (the case ML is completely analogous). In order for a pair (K, C) to be added to CL, the adversary must make a query $\text{KCEnc}(\xi^e = (\phi^e, M))$ such that $K = \phi^e(K^*)$ and $C = \mathsf{iE}(K, M)$.

Since $(\mathsf{iE}, \mathsf{iD})$ is ideal, the probability that after q queries the adversary finds $K_0, K_1, M_0 \neq M_1$ such that $\mathsf{iE}(K_0, M_0) = \mathsf{iE}(K_1, M_1) =: C$, *without* having queried both $\mathsf{iD}(K_0, C)$ *and* $\mathsf{iD}(K_1, C)$ before, is at most $q^2/2^n$. On the other hand, when the adversary queries both $\text{KCDec}(\phi_0, C)$ and $\text{KCDec}(\phi_1, C)$ with $\phi_i(K^*) = K_i$, then (K_0, M_0) and (K_1, M_1), with $M_i := \mathsf{iD}(K_i, C)$, are both added to ML. But then any call to $\text{KCEnc}(\xi^e = (\phi_i', M_i))$ with $\phi'(K^*) = K_i$ will be answered with $\perp$, meaning that (K_i, C) will not be added to CL. So the only way for the event to happen is that the adversary guesses $K_0, K_1, M_0 \neq M_1$ which map to the same C. □

KC $\Rightarrow$ RK. We now prove the first statement of Proposition 4.2. Consider an adversary $\mathcal{B}$ against RK; we construct a reduction $\mathcal{A}$ against KC, which maintains local lists CL' and ML', allowing it to simulate $\mathcal{B}$'s oracles even when its own oracles return $\perp$.

Whenever adversary $\mathcal{B}$ queries RKEnc on (ϕ, M), reduction $\mathcal{A}$ queries its own oracle KCEnc on $\xi^e \colon K \mapsto (\phi(K), M)$. If its answer is $\perp$ then there must be an entry (M, C') in ML'. If there is more than one such entry, $\mathcal{A}$ stops and returns 1; otherwise it returns C'. If $\mathcal{A}$'s KCEnc oracle returned a value $C \neq \perp$, $\mathcal{B}$ returns C and adds (C, M) to its local list CL'.

Analogously, if $\mathcal{B}$ queries RKDec on (ϕ, C), reduction $\mathcal{A}$ queries its own oracle KCDec on $\xi^d \colon K \mapsto (\phi(K), C)$. If its answer is $\perp$ then there must be an entry (C, M') in CL'. If there is more than one such entry, $\mathcal{A}$ stops and returns 1; otherwise it returns M'. If $\mathcal{A}$'s KCDec oracle returned a value $M \neq \perp$, $\mathcal{B}$ returns M and adds (M, C) to its list ML'.

If $\mathcal{A}$ has not stopped the simulation, it forwards $\mathcal{B}$'s output b'.

First note that to any entry $(K, C) \in \mathsf{CL}$ of $\mathcal{A}$'s challenger corresponds an entry $(C, \mathsf{D}(K, M))$ in $\mathcal{A}$'s local list CL' (and similarly for ML and ML'). From Lemma 4.3 we thus have that in case $b = 0$, the probability that there are two entires $(C, M_0), (C, M_1)$ with $M_0 \neq M_1$ in CL' is at most $q^2/2^n =: \epsilon$. Thus, the probability that $\mathcal{A}$ stops the simulation in case $b = 0$ is bounded by ϵ. In case $b = 1$, if $\mathcal{A}$ stops the simulation, it outputs the correct bit. More formally, let E_i denote the event that $\mathcal{A}$ stops the simulation when its challenger's bit $b = i$ and let KC_i denote game $\text{KC-CCA}_{\mathrm{BC}}^{\Xi^e, \Xi^d, \mathcal{A}}$ with bit b fixed to i and similarly for RK_i. Then we have:

$$
\begin{aligned}
\mathbf{Adv}_{\mathrm{BC}}^{\text{rk-cca}}(\Phi, \mathcal{B}) + 1 &= \Pr[\mathrm{RK}_0|E_0]\Pr[E_0] + \Pr[\mathrm{RK}_0|\neg E_0]\Pr[\neg E_0] \\
&\quad + \Pr[\mathrm{RK}_1|E_1]\Pr[E_1] + \Pr[\mathrm{RK}_1|\neg E_1]\Pr[\neg E_1] \\
&= \Pr[\mathrm{RK}_0|E_0]\Pr[E_0] + \Pr[\mathrm{KC}_0|\neg E_0]\Pr[\neg E_0] \\
&\quad + \Pr[\mathrm{RK}_1|E_1]\Pr[E_1] + \Pr[\mathrm{KC}_1|\neg E_1]\Pr[\neg E_1] \\
&\leq 1 \cdot \epsilon + \Pr[\mathrm{KC}_0|\neg E_0]\Pr[\neg E_0] \\
&\quad + 1 \cdot \Pr[E_1] + \Pr[\mathrm{KC}_1|\neg E_1]\Pr[\neg E_1] \\
&= \epsilon + \Pr[\mathrm{KC}_0|E_0]\Pr[E_0] + \Pr[\mathrm{KC}_0|\neg E_0]\Pr[\neg E_0] \\
&\quad + \Pr[\mathrm{KC}_1|E_1]\Pr[E_1] + \Pr[\mathrm{KC}_1|\neg E_1]\Pr[\neg E_1] \\
&= \epsilon + \mathbf{Adv}_{\mathrm{BC}}^{\text{kc-cca}}(\Xi^e, \Xi^d, \mathcal{A}) + 1 \,,
\end{aligned}
$$

where for the second equality we used that games RK and KC have the same output conditioned on $\neg E$; the inequality uses $\Pr[E_0] \leq \epsilon$ by Lemma 4.3; and the second to last equality follows from $\mathcal{B}$'s behavior in case of E happening, thus $\Pr[\mathrm{KC}_0|E_0] = 0$ and $\Pr[\mathrm{KC}_1|E_1] = 1$. This concludes this direction of the proof. □

The above simulation strategy for does not work the converse implication RK $\Rightarrow$ KC, as best illustrated by the following example: Consider a blockcipher for which there exist $K \neq K' \in \mathcal{K}$ and $M \in \mathcal{M}$ with $\mathsf{E}(K, M) = \mathsf{E}(K', M) = C$; now consider a KC adversary $\mathcal{A}$ that queries $\textsc{KCEnc}(\phi, M)$ with $\phi(K^*) = K$; thus $(K, C) \in \mathsf{CL}$. Later $\mathcal{A}$ queries $\textsc{KCDec}(\phi', C)$ and $\textsc{KCDec}(\phi'', C)$ where $\phi'(K^*) = K'$ and $\phi''(K^*) = K$. Reduction $\mathcal{B}$ can forward these queries to its RK oracle and will get M in both cases. But $\textsc{KCDec}(\phi', C)$ should be answered with M (since $(K', C) \notin \mathsf{CL}$), whereas $\textsc{KCDec}(\phi', C)$ should be answered with $\perp$. Thus, storing (C, M) in a local list CL' doesn't help, since both queries would make $\mathcal{B}$ look at this entry, but should result in different actions.

The reduction can thus not perfectly simulate the KC experiment; although this situation is very unlikely to occur for the ideal cipher, the reduction cannot abort and return 1 (as in the above proof), since the event that the adversary found a tuple (K, K', M, C) that would be

hard to find for the ideal cipher is not detectable by the reduction (in contrast to the proof for KC $\Rightarrow$ RK).

The proof will now rely on claw-freeness of the set Φ. If it was perfectly claw-free, then the reduction could perfectly simulate entries $(K, C) \in \mathsf{CL}$ by entries $(\phi, C) \in \mathsf{CL}'$ with $\phi(K^*) = K$. But if claw-freeness is only computational then the adversary could find ϕ, ϕ' with $\phi(K^*) = \phi'(K^*)$ for the key K^* chosen by the game, but which would not be a claw for a random key and thus cannot be output as a solution in the claw-freeness game.

However, we can show that until the adversary finds a claw, the ideal game can be simulated without having chosen any key K^* at all. If the adversary thus finds a claw, it must be one for a random key with non-negligible probability as well. We start with proving this last observation in the following lemma.

Lemma 4.4. *Consider game* $\text{RK-CCA}_{\text{BC}}^{\Phi,\mathcal{B}}$ *with b fixed to 0 and define event E as follows: the adversary $\mathcal{B}$ makes two* RKEnc *queries (ϕ, M) and (ϕ', M), with $\phi \neq \phi'$, both answered by the same C; or it makes two* RKDec *queries (ϕ, C), (ϕ', C), with $\phi \neq \phi'$, both answered by the same M; or it makes two queries* $\text{RKEnc}(\phi, M)$*, answered by C and* $\text{RKDec}(\phi', C)$*, answered by M, with $\phi \neq \phi'$.*

Then there exists an adversary $\mathcal{A}^{cf}$ such that $\Pr[E] \leq q^2 \cdot (\mathbf{Adv}_{\Phi}^{\text{cf}}(\mathcal{A}^{cf}) + 1/2^n) =: \epsilon_2$*, where q is the number of oracle queries $\mathcal{B}$ makes in game* RK-CCA.

Proof. The crucial observation is that game $\text{RK-CCA}_{\text{BC}}^{\Phi,\mathcal{B}}$ when b is fixed to 0 can be simulated without knowing choosing K^* up to the event E happening, that is, the point where the adversary produces the first claw. Let $(\phi_i)_{i=1}^{q}$ denote the list of all functions queried to RKEnc or RKDec. We construct an adversary $\mathcal{A}^{\text{cf}}$ against claw-freeness as follows: it first guesses $i_0, i_1 \in [1, q]$ and then simulates the game for $\mathcal{B}$ until the i_1-th query as follows:

$\mathcal{A}^{\text{cf}}$ keeps a list L with entries of the form (ϕ, M, C). Whenever $\mathcal{B}$ queries $\text{RKEnc}(\phi, M)$, it checks whether there is some entry $(\phi, M, C') \in L$ and if so, returns C'. Else it samples $C \leftarrow \mathcal{M}$, adds (ϕ, M, C) to L and returns C. Whenever $\mathcal{B}$ queries $\text{RKDec}(\phi, C)$, it checks whether there is some entry $(\phi, M', C) \in L$ and if so, returns M'. Else it samples $M \leftarrow \mathcal{M}$, adds (ϕ, M, C) to L and returns M. When $\mathcal{B}$ makes its i_1-the query, $\mathcal{A}^{\text{cf}}$ stops and returns (ϕ_{i_0}, ϕ_{i_1}).

Assume even E occurred and $\mathcal{A}^{\text{cf}}$ guessed correctly that the i_1-th query was the first query after which E occurred and the i_0-th query was the one it collided with. Consider $\mathcal{A}^{\text{cf}}$'s challenger, which chooses $K^* \leftarrow \mathcal{K}$ and checks whether $\phi_{i_0}(K^*) = \phi_{i_1}(K^*)$. If so, then $\mathcal{A}^{\text{cf}}$ won the claw-freeness game. Else the probability that $\mathcal{A}^{\text{cf}}$ would have answered the i_1-th query so that E still occurred is $1/2^n$. □

RK $\Rightarrow$ KC. This direction relies on the claw-freeness of the set Φ. Let $\mathcal{A}$ be a KC adversary; we construct a reduction $\mathcal{B}$ against RK that

maintains two local lists ML' and CL' and aborts whenever event E, defined in Lemma 4.4, occurs.

Whenever $\mathcal{A}$ queries KCEnc for $\xi^e\colon K \mapsto (\phi(K), M)$, reduction $\mathcal{B}$ queries $\text{RKEnc}(\phi, M)$ to receive C. It first checks whether there has been a "colliding" RKEnc query: if there is an entry (ϕ', C, M) in its list CL' with $\phi' \neq \phi$ then $\mathcal{B}$ aborts and returns 1.

To answer the query $\text{KCEnc}(\phi, M)$, an actual KC challenger would check whether $(\phi(K^*), M) \in \mathsf{ML}$. Instead, $\mathcal{B}$ checks its local list ML': if $(\phi, M, C) \in \mathsf{ML}'$, it returns $\perp$; if there is an entry $(\phi' \neq \phi, M, C)$ then $\mathcal{B}$ aborts the simulation and returns 1; if there is no entry $(\cdot, M, C) \in \mathsf{ML}'$, it adds (ϕ, C, M) to its local list CL' and returns C.

KCDec queries for $\xi^d\colon K \mapsto (\phi(K), C)$ are dealt with analogously: $\mathcal{B}$ first queries $\text{RKDec}(\phi, C)$ to receive M. If there is an entry (ϕ', M, C) in ML' with $\phi' \neq \phi$ then $\mathcal{B}$ aborts and returns 1. To answer $\text{KCDec}(\phi, C)$, an actual KC challenger would now check whether $(\phi(K^*), C) \in \mathsf{CL}$. Instead, $\mathcal{B}$ checks its local list CL': if $(\phi, C, M) \in \mathsf{CL}'$ it returns $\perp$; if there is an entry $(\phi' \neq \phi, C, M)$ then $\mathcal{B}$ aborts the simulation and returns 1; if there is no entry $(\cdot, C, M) \in \mathsf{CL}'$, it adds (ϕ, M, C) to its local list ML' and returns M.

We show that the simulation is perfect if $\mathcal{B}$ does not abort; consider a query $\text{KCDec}(\phi, C)$: (1) If $(\phi, C, M) \in \mathsf{CL}'$ then there must have been a query $\text{KCEnc}(\phi, M)$ with $C = \mathsf{E}(\phi(K^*), M)$; thus a KC challenger would have added $(\phi(K^*), C)$ to CL and the above KCDec query would have been answered by $\perp$. (2) If there is no entry $(\cdot, C, M) \in \mathsf{CL}'$ then $(\phi(K^*), C)$ cannot be in CL, since whenever it is added to CL, an entry $(\cdot, C, \mathsf{D}(\phi(K^*), C)$ would have been added to CL'. Perfect simulation of KCEnc queries is argued completely analogously.

On the other hand, we have that $\mathcal{B}$ aborts precisely when E, as defined in Lemma 4.4, occurs. With ϵ as defined in the latter, we thus have

$$\begin{aligned}
\mathbf{Adv}^{\text{kc-cca}}_{\text{BC}}(\Xi^e, \Xi^d, \mathcal{A}) + 1 &= \Pr[\text{KC}_0|E_0]\Pr[E_0] + \Pr[\text{KC}_0|\neg E_0]\Pr[\neg E_0] \\
&\quad + \Pr[\text{KC}_1|E_1]\Pr[E_1] + \Pr[\text{KC}_1|\neg E_1]\Pr[\neg E_1] \\
&= \Pr[\text{KC}_0|E_0]\Pr[E_0] + \Pr[\text{RK}_0|\neg E_0]\Pr[\neg E_0] \\
&\quad + \Pr[\text{KC}_1|E_1]\Pr[E_1] + \Pr[\text{RK}_1|\neg E_1]\Pr[\neg E_1] \\
&\leq 1 \cdot \epsilon + \Pr[\text{RK}_0|\neg E_0]\Pr[\neg E_0] \\
&\quad + 1 \cdot \Pr[E_1] + \Pr[\text{RK}_1|\neg E_1]\Pr[\neg E_1] \\
&= \epsilon + \Pr[\text{RK}_0|E_0]\Pr[E_0] + \Pr[\text{RK}_0|\neg E_0]\Pr[\neg E_0] \\
&\quad + \Pr[\text{RK}_1|E_1]\Pr[E_1] + \Pr[\text{RK}_1|\neg E_1]\Pr[\neg E_1] \\
&= \epsilon + \mathbf{Adv}^{\text{rk-cca}}_{\text{BC}}(\Phi, \mathcal{B}) + 1 \,,
\end{aligned}$$

where for the second equality we used that games RK and KC have the same output conditioned on $\neg E$; the inequality uses $\Pr[E_0] \leq \epsilon$ by Lemma 4.4; and the second to last equality follows from $\mathcal{B}$'s behavior in case of E happening, thus $\Pr[\text{RK}_0|E_0] = 0$ and $\Pr[\text{RK}_1|E_1] = 1$. This concludes this direction of the proof. □

4.5.2 *KC(KDM-only) ⇔ KDM*

Proposition 4.5. *Let* BC *be a blockcipher. Let* Ξ^e *and* Ξ^d *be two KDM-only CDF sets with* $\Xi^e = \{id\} \times \Psi$ *and* $\Xi^d = \{id\} \times \Gamma$*. Then for every adversary* $\mathcal{A}$ *there exists an adversary* $\mathcal{B}$ *such that*

$$\mathbf{Adv}^{\text{kc-cca}}_{\text{BC}}(\Xi^e, \Xi^d, \mathcal{A}) = \mathbf{Adv}^{\text{kdm-cca}}_{\text{BC}}(\Psi, \mathcal{B}) \ ,$$

and for every adversary $\mathcal{B}$ *there exist adversaries* $\mathcal{A}$ *and* $\mathcal{A}^{cf}$ *such that*

$$\mathbf{Adv}^{\text{kdm-cca}}_{\text{BC}}(\Psi, \mathcal{B}) \leq q^2 \cdot \mathbf{Adv}^{\text{cf}}_{\Psi}(\mathcal{A}^{cf}) + \mathbf{Adv}^{\text{kc-cca}}_{\text{BC}}(\Xi^e, \Xi^d, \mathcal{A}) \ ,$$

where q *is the maximum number of queries adversary* $\mathcal{B}$ *makes.*

We show that our definition is equivalent to the KDM definition from [FKV17] for claw-free sets Ψ. Recall that (Ξ^e, Ξ^d) is KDM-only if $\Xi^e = \{id\} \times \Psi$ and $\Xi^d = \{id\} \times \Gamma$, where Ψ is a KDM set of functions mapping keys to messages and Γ is the set of constant functions $K \mapsto M$ for all $M \in \mathcal{M}$; we assume $\Gamma \subseteq \Psi$.

We first observe that games KC and KDM are similar in that every entry C in CL in game KDM corresponds to (K^*, C) in CL in KC. The crucial difference is that in KDM there is no list ML, thus all KDMEnc queries are allowed, whereas KC disallows queries in ML. KDM implying KC can be shown because the reduction can perfectly simulate the list ML.

The converse (KC ⇒ KDM) is trickier to show and it relies on claw-freeness of the set Ψ. To illustrate this, consider the following attack against KDM, which does not work against KC: for two random $C_0, C_1 \in \mathcal{M}$, the adversary queries $\textsc{Dec}(C_0)$ and $\textsc{Dec}(C_1)$ to receive M_0 and M_1. It then queries $\text{KDM}\textsc{Enc}(\psi_{M_0,M_1})$ with $\psi\colon K^* \mapsto M_{K_1^*}$, where K_1^* is the first bit of K^*. When receiving C_i as a reply, the adversary learnt that i is the first bit of K^* and it can use the same strategy to learn all other bits of K^*. Note that in the KC game, the two decryption queries add (K^*, M_0) and (K^*, M_1) to ML and consequently $\text{KC}\textsc{Enc}(\xi^e := id \times \psi_{M_0,M_1})$ is answered with $\perp$.

KDM ⇒ KC. To show that KDM implies KC, consider a KC adversary $\mathcal{A}$ and let us construct a reduction $\mathcal{B}$ against KDM. $\mathcal{B}$ basically simulates the KC oracles for $\mathcal{A}$ by forwarding $\mathcal{A}$'s queries to its own KDM oracles. To simulate game KC correctly, $\mathcal{B}$ maintains its own list ML'. Whenever $\mathcal{A}$ queries $\text{KC}\textsc{Dec}(\xi^d)$ for $\xi_2 \in \Xi^d$, we have $\xi_2\colon K \mapsto (K, C)$ for some $C \in \mathcal{M}$. Reduction $\mathcal{B}$ queries C to $\textsc{Dec}$ to obtain M, which it returns to $\mathcal{A}$; $\mathcal{B}$ also appends (M, C) to its list ML' (whereas an actual KC challenger would have appended (K^*, M) to its list ML).

Whenever $\mathcal{A}$ queries $\text{KC}\textsc{Enc}(\xi_1)$ with $\xi_1\colon K \mapsto (K, \psi(K))$ for some $\psi \in \Psi$, the reduction queries $\text{KDM}\textsc{Enc}(\psi)$ to obtain C. If for some M' there is an entry $(M', C) \in \mathsf{ML}'$: $\mathcal{B}$ returns $\perp$; otherwise, it returns C.

To see that this correctly simulates the list ML that $\mathcal{B}$'s challenger maintains, observe that (K^*, M) is in the latter iff $(M, \mathsf{E}(K^*, M))$ is in

$\mathcal{B}$'s local list ML' and since $\mathsf{E}(K^*, \cdot)$ is a permutation, each C uniquely determines the corresponding M, thus $\mathcal{B}$'s simulation is perfect. We thus have $\mathbf{Adv}^{\text{kc-cca}}_{\text{BC}}(\Xi^e, \Xi^d, \mathcal{A}) = \mathbf{Adv}^{\text{kdm-cca}}_{\text{BC}}(\Psi, \mathcal{B})$.

KC $\Rightarrow$ KDM. In this direction, we assume an adversary $\mathcal{B}$ against KDM and construct $\mathcal{A}$ against KC. This time, $\mathcal{A}$ will not be able to perfectly simulate the game; however, $\mathcal{A}$ can detect when it is not able to. In this case $\mathcal{A}$ aborts and returns $b' = 1$ (it guesses that it is talking to the real blockcipher). As for in the proof for KC $\Rightarrow$ RK, we show that if $b = 0$ (the ideal case), the probability of $\mathcal{A}$ aborting is bounded—in contrast to the previous proof, we bound it by the advantage of breaking clawfreeness.

We start by defining the following event E in game $\text{KDM-CCA}^{\Psi,\mathcal{B}}_{\text{BC}}$: let $(M_i)_i$ denote the messages returned by the Dec oracle and let $(\psi_j)_j$ denote the queries made to the KDMEnc oracle; let K^* denote the challenge key. We say E occurs if we either have (1) $\psi_{j_0}(K^*) = \psi_{j_1}(K^*)$ for some $j_0 \neq j_1$; or (2) $\psi_j(K^*) = M_i$ for some j, i. We show the following:

Lemma 4.6. *Consider game* $\text{KDM-CCA}^{\Psi,\mathcal{B}}_{\text{BC}}$ *with b fixed to 0 and let E be the event defined above. Then there exists an adversary $\mathcal{A}^{cf}$ such that* $\Pr[E] \leq q^2 \cdot \mathbf{Adv}^{\text{cf}}_{\Psi}(\mathcal{A}^{cf})$*, where q is the number of oracle queries $\mathcal{B}$ makes in game* KDM-CCA*.*

Proof. First note that there exists some first query j_1 that makes E occur, and some query $j_0 < j_1$, which will be the query with which the j_1-th query "collides". $\mathcal{A}^{\text{cf}}$ starts with guessing j_0 and j_1 and simulates the game until the j_1-th query as follows: any query $\text{Dec}(C_i)$ is answered with a random $M_i \leftarrow \mathcal{M}$; any query $\text{KDMEnc}(\psi_i)$ is answered with a random $C_i \leftarrow \mathcal{M}$ (all repeated queries are answered consistently).

If the j_0-th query was a query $\text{Dec}(C_i)$ answered by M_i then define $\psi_{j_0} \colon K \mapsto M_i$; if it was a query $\text{KDMEnc}(\psi)$ then let $\psi_{j_0} := \psi$; define ψ_{j_1} analogously. $\mathcal{A}^{\text{cf}}$ returns (ψ_{j_0}, ψ_{j_1}).

First note that if $\mathcal{A}$ guessed j_1 correctly then the simulation is perfect, as for the random key K^*, the same message was never queried twice (via encryption or decryption). Moreover, if $\mathcal{A}$ also guessed j_0 correctly then (ψ_{j_0}, ψ_{j_1}) is indeed a collision and $\mathcal{A}$ wins the claw-freeness game. We thus have $\mathbf{Adv}^{\text{cf}}_{\Psi}(\mathcal{A}^{\text{cf}}) \geq 1/q^2 \cdot \Pr[E]$, which proves the lemma. □

We now give a reduction $\mathcal{A}$ against KC that simulates the KDM game to adversary $\mathcal{B}$ by forwarding all queries to its own oracle. When $\mathcal{A}$ cannot answer a KDMEnc query because its own oracle KCEnc replies $\perp$ it aborts the simulation. In order to use the above lemma, $\mathcal{A}$ will also abort when it could continue the simulation. In particular, $\mathcal{A}$ aborts and returns 1 when one of the two things happen:

(1) $\mathcal{B}$ makes two queries ψ, ψ' to KDMEnc, which when forwarded to $\mathcal{A}$'s RKEnc oracle return the same C; or
(2) $\mathcal{A}$ forwards a KDMEnc query to its KDMEnc oracle which is answered with $\perp$.

Note that these two cases define precisely the event E above. We thus have that $\mathcal{A}$ aborts the simulation whenever E occurs. If $\mathcal{A}$ does not abort the simulation, it returns whatever $\mathcal{B}$ outputs.

Let KC_i denote game $\text{KC-CCA}_{\mathrm{BC}}^{\Xi^e,\Xi^d,\mathcal{A}}$ with bit b fixed to i and E_i the event that E occurs in this case. Define KDM_i analogously and let $\epsilon := q^2 \cdot \mathbf{Adv}^{\mathrm{cf}}_{\Psi}(\mathcal{A}^{\mathrm{cf}})$. Then we have:

$$\begin{aligned}
\mathbf{Adv}^{\text{kdm-cca}}_{\mathrm{BC}}(\Psi,\mathcal{B}) + 1 &= \Pr[\mathrm{KDM}_0|E_0]\Pr[E_0] + \Pr[\mathrm{KDM}_0|\neg E_0]\Pr[\neg E_0] \\
&\quad + \Pr[\mathrm{KDM}_1|E_1]\Pr[E_1] + \Pr[\mathrm{KDM}_1|\neg E_1]\Pr[\neg E_1] \\
&= \Pr[\mathrm{KDM}_0|E_0]\Pr[E_0] + \Pr[\mathrm{KC}_0|\neg E_0]\Pr[\neg E_0] \\
&\quad + \Pr[\mathrm{KDM}_1|E_1]\Pr[E_1] + \Pr[\mathrm{KC}_1|\neg E_1]\Pr[\neg E_1] \\
&\leq +1 \cdot \epsilon + \Pr[\mathrm{KC}_0|\neg E_0]\Pr[\neg E_0] \\
&\quad 1 \cdot \Pr[E_1] + \Pr[\mathrm{KC}_1|\neg E_1]\Pr[\neg E_1] \\
&= \epsilon + \Pr[\mathrm{KC}_0|E_0]\Pr[E_0] + \Pr[\mathrm{KC}_0|\neg E_0]\Pr[\neg E_0] \\
&\quad + \Pr[\mathrm{KC}_1|E_1]\Pr[E_1] + \Pr[\mathrm{KC}_1|\neg E_1]\Pr[\neg E_1] \\
&= \epsilon + \mathbf{Adv}^{\text{kc-cca}}_{\mathrm{BC}}(\Xi^e, \Xi^d, \mathcal{A}) + 1
\end{aligned}$$

where for the second equality we used that games KDM and KC have the same output conditioned on $\neg E$; the inequality uses $\Pr[E_0] \leq \epsilon$ by Lemma 4.6; and the second to last equality follows from $\mathcal{B}$'s behavior in case of E happening, thus $\Pr[\mathrm{KC}_0|E_0] = 0$ and $\Pr[\mathrm{KC}_1|E_1] = 1$. This concludes this direction of the proof.

5

KCA-SECURE BLOCKCIPHERS

In this chapter we study the feasibility of achieving security against key-correlated attacks in the ideal-cipher model. In this setting the adversary has oracle access to the ideal cipher in both the forward and backward direction.

5.1 KCA SECURITY OF THE IDEAL CIPHER

We begin by extending the standard notion of CCA security to the KCA setting, where key-dependent messages may be enciphered under related keys. Consider game $\text{KC-CCA}^{\Xi^e,\Xi^d,\mathcal{A}}_{\text{Block}(k,n)}$ shown in Fig. 5.1. The KC-CCA advantage of an adversary $\mathcal{A}$ against the ideal cipher with key length k and block length n is defined by

$$\mathbf{Adv}^{\text{kc-cca}}_{\text{Block}(k,n)}(\Xi^e,\Xi^d,\mathcal{A}) := 2\cdot\Pr\left[\text{KC-CCA}^{\Xi^e,\Xi^d,\mathcal{A}}_{\text{Block}(k,n)}\right]-1\ .$$

We now prove a feasibility theorem for the ideal cipher that generalizes and extends those of both Bellare and Kohno [BK03] and of Farshim et al. [FKV17]. We define four conditions on CDF sets, which exclude trivial attacks.

Unpredictability. The unpredictability advantage of an adversary $\mathcal{A}$ against a CDF set Ξ is defined as

$$\mathbf{Adv}^{\text{up}}_{\Xi}(\mathcal{A}) := \Pr\left[\xi(K)=(K_0,M_0) : (\xi,(K_0,M_0)) \twoheadleftarrow \mathcal{A}; K \twoheadleftarrow \mathcal{K}\right]\ ,$$

where we require that $\xi \in \Xi$. Informally, we say Ξ is unpredictable if the above advantage is "small" for every "reasonable" $\mathcal{A}$.

We also define a multi-shot version of this game where $\mathcal{A}$ outputs a list L_1 of candidates for ξ and a list L_2 of predicted values (K_0,M_0). Adversary $\mathcal{A}$ wins if some ξ in L_1 evaluates to some (K_0,M_0) in L_2 when run on a random key K. Denoting the advantage in this game by $\mathbf{Adv}^{\text{m-up}}_{\Xi}(\mathcal{A})$, a simple guessing argument shows that for any $\mathcal{A}$ there exists $\mathcal{B}$ with

$$\mathbf{Adv}^{\text{m-up}}_{\Xi}(\mathcal{A}) \ \le\ \ell_1\cdot\ell_2\cdot\mathbf{Adv}^{\text{up}}_{\Xi}(\mathcal{B})\ ,$$

where ℓ_1 and ℓ_2 are upper bounds on the sizes of L_1 and L_2 respectively. [1]

1. Single-shot adversary $\mathcal{B}$ runs any given multi-shot $\mathcal{A}$ to obtain two lists, picks one entry at random from each list and outputs the pair. Then the success probability of $\mathcal{B}$ is at least $1/(\ell_1\ell_2)$ times that of $\mathcal{A}$.

We require that $\xi^e \in \Xi^e$ and $\xi^d \in \Xi^d$ for all queries ξ^e, ξ^d.

Game $\text{KC-CCA}^{\Xi^e,\Xi^d,\mathcal{A}}_{\text{Block}(k,n)}$:	Proc. $\text{KCE}_{\text{NC}}(\xi^e)$:	Proc. $\text{KCD}_{\text{EC}}(\xi^d)$:
$b \twoheadleftarrow \{0,1\}$	$(K,M) \leftarrow \xi^e(K^*)$	$(K,C) \leftarrow \xi^d(K^*)$
$(\mathsf{iE},\mathsf{iD}) \twoheadleftarrow \text{Block}(k,n)$	if $(K,M) \in \mathsf{ML}$: return $\perp$	if $(K,C) \in \mathsf{CL}$: return $\perp$
$(\mathsf{iE}',\mathsf{iD}') \twoheadleftarrow \text{Block}(k,n)$	$C \leftarrow \mathsf{iE}(K,M)$	$M \leftarrow \mathsf{iD}(K,C)$
$K^* \twoheadleftarrow \{0,1\}^k$	if $b = 0$: $C \leftarrow \mathsf{iE}'(K,M)$	if $b = 0$: $M \leftarrow \mathsf{iD}'(K,C)$
$b' \twoheadleftarrow \mathcal{A}^{\mathsf{iE},\mathsf{iD},\text{KCE}_{\text{NC}},\text{KCD}_{\text{EC}}}$	$\mathsf{CL} \leftarrow \mathsf{CL} : (K,C)$	$\mathsf{ML} \leftarrow \mathsf{ML} : (K,M)$
return $(b' = b)$	return C	return M

Figure 5.1 – Game defining (Ξ^e, Ξ^d)-KC-CCA security of the ideal cipher with key length k and block length n.

KEY-UNPREDICTABILITY. The key-unpredictability advantage of an adversary $\mathcal{A}$ against a CDF set Ξ is defined as

$$\mathbf{Adv}^{\text{kup}}_{\Xi}(\mathcal{A}) := \Pr\left[\xi|_1(K) = K_0 : (\xi, K_0) \twoheadleftarrow \mathcal{A}; K \twoheadleftarrow \mathcal{K}\right] ,$$

where we require that $\xi \in \Xi$. Here $\xi|_1(K)$ denotes the projection to the first coordinate of $\xi(K)$. Informally we say Ξ is key-unpredictable if the above advantage is "small" for every "reasonable" $\mathcal{A}$.

We also define a multi-shot version of this game where $\mathcal{A}$ outputs a list L_1 of candidates ξ and a list L_2 of predicted values K_0. Adversary $\mathcal{A}$ wins if for a random key K, for some ξ in L_1 we have $\xi|_1(K) = K_0$ for some K_0 in L_2. If we denote the advantage in this game by $\mathbf{Adv}^{\text{m-kup}}_{\Xi}(\mathcal{A})$, again we have that there exists a single-shot $\mathcal{B}$ such that

$$\mathbf{Adv}^{\text{m-kup}}_{\Xi}(\mathcal{A}) \;\leq\; \ell_1 \cdot \ell_2 \cdot \mathbf{Adv}^{\text{kup}}_{\Xi}(\mathcal{B}) ,$$

where ℓ_1 and ℓ_2 are upper bounds on the sizes of L_1 and L_2 respectively. Note that key-unpredictability implies unpredictability. The converse does not hold as $\xi\colon K \mapsto (0, K)$ is unpredictable but not key-unpredictable.

CLAW-FREENESS. This requires that it is hard to find two distinct functions that output the same on a random input. Formally, the claw-free advantage of an adversary $\mathcal{A}$ against a CDF set Ξ is defined as

$$\mathbf{Adv}^{\text{cf}}_{\Xi}(\mathcal{A}) := \Pr\left[\xi \neq \xi' \wedge \xi(K) = \xi'(K) : (\xi, \xi') \twoheadleftarrow \mathcal{A}; K \twoheadleftarrow \mathcal{K}\right] ,$$

where we require that $\xi, \xi' \in \Xi$. We say Ξ is claw-free if the above advantage is "small" for every "reasonable" $\mathcal{A}$. Once again, a multi-shot version of the game with lists of candidates L_1 and L_2 for ξ and ξ' can be defined, and as before there exists $\mathcal{B}$ with

$$\mathbf{Adv}^{\text{m-cf}}_{\Xi}(\mathcal{A}) \;\leq\; \ell_1 \cdot \ell_2 \cdot \mathbf{Adv}^{\text{cf}}_{\Xi}(\mathcal{B}) .$$

CROSS-KEY-CLAW-DETECTABILITY. We introduce a new notion and call a pair of CDF sets (Ξ^e, Ξ^d) cross-key-claw-detectable (xkcd) [2] if there is

2. Please do not confuse this notion with the (popular :) web-comic xkcd.com.

an efficient detection algorithm Det such that for any $\mathcal{A}$ the advantage below is small.

$$\mathbf{Adv}^{\text{xkcd}}_{\Xi^e,\Xi^d,\mathsf{Det}}(\mathcal{A}) :=$$
$$\Pr\left[\begin{array}{l|l} \mathsf{Det}(\xi^e,C,\xi^d,M,1) \neq \left[(\xi^e|_1(K),C) = \xi^d(K)\right] \vee \\ \mathsf{Det}(\xi^e,C,\xi^d,M,2) \neq \left[(\xi^d|_1(K),M) = \xi^e(K)\right] & \begin{array}{l} (\xi^e,C,\xi^d,M) \twoheadleftarrow \mathcal{A}; \\ K \twoheadleftarrow \mathcal{K} \end{array} \end{array}\right]$$

We require that $\xi^e \in \Xi^e$ and $\xi^d \in \Xi^d$. Roughly speaking, xkcd provides a way to treat legality of queries across encryption and decryption queries in the KCA game. For example, in an illegal decryption query the output of an earlier encryption query (stored by the game in list CL, cf. Fig. 4.1) is sent to the decryption oracle through a CDF function whose key component matches that used in encryption. Det allows us to decide when such cases occur and deal with them appropriately in the security analysis.

A generalization of this notion considers lists L_1 for pairs (ξ^e, C) and L_2 for (ξ^d, M) and $\mathcal{A}$ wins if Det fails for any given tuple (ξ^e, C, ξ^d, M) in the product of the lists. Again, a guessing argument shows that for every $\mathcal{A}$ there exists $\mathcal{B}$ with

$$\mathbf{Adv}^{\text{m-xkcd}}_{\Xi^e,\Xi^d,\mathsf{Det}}(\mathcal{A}) \;\leq\; \ell_1 \cdot \ell_2 \cdot \mathbf{Adv}^{\text{xkcd}}_{\Xi^e,\Xi^d,\mathsf{Det}}(\mathcal{B})\;.$$

REMARK. We can modify claw-freeness to claw-freeness of keys over the union $\Xi^e \cup \Xi^d$. This would immediately imply xkcd, as Det can always return "No". But in this case the reach of our feasibility result below will not extend to the KDM setting, since under KDM a single key is used in encryption and decryption and hence they always collide.

We will now state and prove our first feasibility result, which identifies a set of sufficient conditions on the pair (Ξ^e, Ξ^d) so that the ideal cipher is KCA secure with respect to it.

Theorem 5.1 (KCA security of the ideal cipher)**.** *Fix a key length k and block length n and let Ξ^e and Ξ^d be two sets of CDFs with signature $\xi\colon \{0,1\}^k \to \{0,1\}^k \times \{0,1\}^n$. Then for any (Ξ^e,Ξ^d)-KC-CCA adversary $\mathcal{A}$ against the ideal cipher making at most q direct ideal cipher queries, q_e encryption and q_d decryption queries, there are (multi-shot) adversaries $\mathcal{B}_1,\ldots,\mathcal{B}_5$ such that*

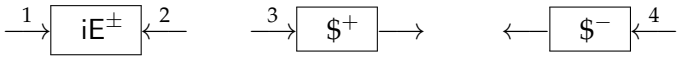
$$\begin{aligned}\mathbf{Adv}^{\text{kc-cca}}_{\text{Block}(k,n)}(\Xi^e,\Xi^d,\mathcal{A}) \leq\; & \mathbf{Adv}^{\text{m-kup}}_{\Xi^e}(\mathcal{B}_1) + \mathbf{Adv}^{\text{m-kup}}_{\Xi^d}(\mathcal{B}_2) + \mathbf{Adv}^{\text{m-cf}}_{\Xi^e}(\mathcal{B}_3) \\ & + \mathbf{Adv}^{\text{m-cf}}_{\Xi^d}(\mathcal{B}_4) + \mathbf{Adv}^{\text{m-xkcd}}_{\Xi^e,\Xi^d,\mathsf{Det}}(\mathcal{B}_5) \\ & + \left(qq_e + qq_d + q_e^2 + q_d^2 + q_eq_d\right)/2^n\;.\end{aligned}$$

Algorithm $\mathcal{B}_1$ outputs two lists of sizes at most $\ell_1 \leq q_e$ and $\ell_2 \leq q$; $\mathcal{B}_2$ outputs lists of sizes at most $\ell_1 \leq q_d$ and $\ell_2 \leq q$; $\mathcal{B}_3$'s lists are of sizes at most $\ell_1, \ell_2 \leq q_e$; $\mathcal{B}_4$'s lists are of sizes at most $\ell_1, \ell_2 \leq q_d$, and $\mathcal{B}_5$'s lists of sizes at most $\ell_1 \leq q_e$ and $\ell_2 \leq q_d$.

Proof. The idea of the proof is that we will gradually modify KC-CCA of Fig. 5.1 so that KCEnc and KCDec oracles run with two *independent and forgetful random oracles*, denoted by $\$^+$ and $\$^-$ respectively, and further the validity checks are performed by the detection algorithm Det. These modifications ensure that we arrive at a game which does not depend on the explicit key-correlated keys, messages, or ciphertexts.

This is the key goal to bear in mind throughout the proof.

Pictorially, in this intermediate game with forgetful oracles the adversary can make queries that correspond to 1 (corresponding to iE), 2 (corresponding to iD), 3 and 4:



We now describe the transitional steps; the details can be found in Figs. 5.2 and 5.3.

Game_0 : This is the KC-CCA game with respect to $b = 1$ and where the four oracles lazily sample the ideal cipher. The same ideal cipher is used in iE and iD as those used in KCEnc and KCDec oracles. Consistency is ensured via shared lists T^+, T^-, Im^+, and Im^- used in lazy sampling.

Game_1 : This game differs from Game_0 in that different lists are used in iE and iD and in KCEnc and KCDec. The game still ensures consistency across the lists and hence there are no functional changes. This game also sets a flag $\mathsf{Bad}_{\mathrm{kup}}$ preparing us to decouple the two lists. More precisely, whenever a call to iE or iD needs to use the lists of KCEnc and KCDec, or vice versa, $\mathsf{Bad}_{\mathrm{kup}}$ is set. This game also sets a bad flag $\mathsf{Bad}_{\mathrm{col}_1}$ whenever it samples a ciphertext (resp. plaintext) that was sampled before and hence appears on Im_1^+ or Im_2^+ (resp. Im_1^- or Im_2^-).

Game_2 : This game omits the code after $\mathsf{Bad}_{\mathrm{kup}}$ or $\mathsf{Bad}_{\mathrm{col}_1}$ is set. Thus Game_1 and Game_2 are identical until $\mathsf{Bad}_{\mathrm{kup}}$ or $\mathsf{Bad}_{\mathrm{col}_1}$ is set. We defer the analysis of this event to a later game. As a result the two ideal ciphers used in iE and iD and KCEnc and KCDec are decoupled.

Game_3 : This game checks for repeat queries to the ideal cipher used within the KCEnc and KCDec oracles. (Note that the queries to these oracles are assumed, without loss of generality, to be distinct.) The game sets a flag $\mathsf{Bad}_{\mathrm{cf}}$ when this takes place. This game also sets a bad flag $\mathsf{Bad}_{\mathrm{col}_2}$ whenever it samples a ciphertext (resp. plaintext) that was sampled before in a previous KCEnc or KCDec query (resp., also in a previous KCEnc and KCDec query). By dropping the code after $\mathsf{Bad}_{\mathrm{cf}}$ and $\mathsf{Bad}_{\mathrm{col}_2}$, we transition to a game where the ideal cipher within KCEnc and KCDec act as forgetful random oracles.

Game_4 : This games uses the detection algorithm Det to check whether or not queries to KCEnc or those to KCDec are valid. More explicitly:

(1) For each decryption query ξ^d, KCDEC runs $\mathsf{Det}(\xi^e, C, \xi^d, 0^n, 1)$ for all previous queries ξ^e to KCENC, which were answered with C. (Intuitively, Det checks whether $\xi^d(K) = (\xi^e|_1(K), C)$; the input 0^n is arbitrary.)

(2) Analogously, for each query ξ^e, the KCENC oracle runs $\mathsf{Det}(\xi^e, 0^n, \xi^d, M, 2)$ for all previous queries ξ^d to KCDEC which resulted in output M. If Det (which intuitively checks if $\xi^e(K) = (\xi^d|_1(K), M)$) returns "Yes" for any of these, KCENC returns $\perp$, else it answers randomly.

Note that this game introduces no functional changes since it still computes the validity of a query using the lists. However it sets a bad flag $\mathsf{Bad}_{\text{xkcd}}$ when the validity check determined by Det is different from the (true) value of validity determined using the lists ML and CL. This then prepares us to drop the explicit check using the lists and only use Det in the next game.

Game_5 : This game drops the code after $\mathsf{Bad}_{\text{xkcd}}$, and so is identical to Game_4 unless $\mathsf{Bad}_{\text{xkcd}}$ has been set.

Note that in Game_5 the iE and iD oracles implement an ideal cipher whereas the KCENC and KCDEC oracles implement two forgetful random oracles, except that sometimes they might return $\perp$ if the Det says that a query is invalid. To signify these, in what follows we refer to KCENC by \$ and KCDEC by $\$^-$. We also use iE^+ for iE and iE^- for iD.

Note also that the above games are all identical unless one of the bad flags is set. Thus we analyze the probability of setting any of the bad flags in the final game Game_5, where the behavior of the oracles are independent of the secret key K^*.

We start by analyzing the probability of setting $\mathsf{Bad}_{\text{kup}}$ and $\mathsf{Bad}_{\text{col}_1}$. Flag $\mathsf{Bad}_{\text{kup}}$ can be set under iE if a query (K, M) to iE appears on T_2^+. Since only KCENC and KCDEC write to T_2^+, this would be the case if (a) $(K, M) = \xi^e(K^*)$ for some query ξ^e to KCENC or (b) $(K, M) = (\xi^d|_1(K^*), M')$ for M' a randomly chosen output of KCDEC.

In case (a) we can build an adversary that wins the multi-shot *unpredictability* of Ξ^e by simulating the oracles as in Game_5 (without the need for K^*) and outputting two lists L_1 and L_2 consisting of all tuples (K, M) that were submitted to iE and ξ^e that were submitted to KCENC. These lists are of sizes q and q_e respectively.

The 'trick' that allows us to simulate without K^* stems from the fact that in Game_5 all ciphertexts and messages are generated independently of $\xi^e(K^*)$ and $\xi^d(K^*)$.

In case (b), however, we build an adversary that wins the multi-shot *key unpredictability* of Ξ^d (note the exponent d). The four oracles are simulated as in Game_5 (once again without the need for K^*), but now the adversary outputs two lists L_1 and L_2 consisting of all keys K for tuples (K, M) submitted to iE and ξ^e that were submitted to KCDEC. These lists are of sizes q and q_d respectively.

We now look at the probability of setting $\mathsf{Bad}_{\text{col}_1}$. A straightforward analysis shows that this probability is upper bounded by $q(q_e + q_d)/2^n$.

$\underline{\mathsf{Game}_0^{\Xi^e,\Xi^d,\mathcal{A}}}$:
$b \leftarrow 1$; $K^* \twoheadleftarrow \{0,1\}^k$
$(\mathsf{iE},\mathsf{iD}) \twoheadleftarrow \mathrm{Block}(k,n)$
$b' \twoheadleftarrow \mathcal{A}^{\mathsf{iE},\mathsf{iD},\mathrm{KCE}\textsc{nc},\mathrm{KCD}\textsc{ec}}$
return $(b' = 1)$

$\underline{\text{Proc. } \mathsf{iE}(K,M)}$:
if $\mathsf{T}^+[K,M]$: return $\mathsf{T}^+[K,M]$
else $C \twoheadleftarrow \{0,1\}^n \setminus \mathsf{Im}^+[K]$
$\mathsf{T}^+[K,M] := C$
$\mathsf{T}^-[K,C] := M$
$\mathsf{Im}^+[K] \leftarrow \mathsf{Im}^+[K] : C$
$\mathsf{Im}^-[K] \leftarrow \mathsf{Im}^-[K] : M$
return C

$\underline{\text{Proc. } \mathsf{iD}(K,C)}$:
if $\mathsf{T}^-[K,C]$: return $\mathsf{T}^-[K,C]$
else $M \twoheadleftarrow \{0,1\}^n \setminus \mathsf{Im}^-[K]$
$\mathsf{T}^-[K,C] := M$
$\mathsf{T}^+[K,M] := C$
$\mathsf{Im}^-[K] \leftarrow \mathsf{Im}^-[K] : M$
$\mathsf{Im}^+[K] \leftarrow \mathsf{Im}^+[K] : C$
return M

$\underline{\text{Proc. } \mathrm{KCE}\textsc{nc}(\xi^e)}$:
$(K,M) \leftarrow \xi^e(K^*)$
if $(K,M) \in \mathsf{ML}$: return $\bot$
if $\mathsf{T}^+[K,M]$: return $\mathsf{T}^+[K,M]$
else $C \twoheadleftarrow \{0,1\}^n \setminus \mathsf{Im}^+[K]$
$\mathsf{T}^+[K,M] := C$
$\mathsf{T}^-[K,C] := M$
$\mathsf{Im}^+[K] \leftarrow \mathsf{Im}^+[K] : C$
$\mathsf{Im}^-[K] \leftarrow \mathsf{Im}^-[K] : M$
$\mathsf{CL} \leftarrow \mathsf{CL} : (K,C)$
return C

$\underline{\text{Proc. } \mathrm{KCD}\textsc{ec}(\xi^d)}$:
$(K,C) \leftarrow \xi^d(K^*)$
if $(K,C) \in \mathsf{CL}$: return $\bot$
if $\mathsf{T}^-[K,C]$: return $\mathsf{T}^-[K,C]$
else $M \twoheadleftarrow \{0,1\}^n \setminus \mathsf{Im}^-[K]$
$\mathsf{T}^-[K,C] := M$
$\mathsf{T}^+[K,M] := C$
$\mathsf{Im}^-[K] \leftarrow \mathsf{Im}^-[K] : M$
$\mathsf{Im}^+[K] \leftarrow \mathsf{Im}^+[K] : C$
$\mathsf{ML} \leftarrow \mathsf{ML} : (K,M)$
return M

$\underline{\mathsf{Game}_1^{\Xi^e,\Xi^d,\mathcal{A}}}$:
$b \leftarrow 1$; $K^* \twoheadleftarrow \{0,1\}^k$
$(\mathsf{iE},\mathsf{iD}) \twoheadleftarrow \mathrm{Block}(k,n)$
$b' \twoheadleftarrow \mathcal{A}^{\mathsf{iE},\mathsf{iD},\mathrm{KCE}\textsc{nc},\mathrm{KCD}\textsc{ec}}$
return $(b' = 1)$

$\underline{\text{Proc. } \mathsf{iE}(K,M)}$:
if $\mathsf{T}_1^+[K,M]$: return $\mathsf{T}_1^+[K,M]$
if $\mathsf{T}_2^+[K,M]$: $\boxed{\mathsf{Bad}_{\mathrm{kup}} \leftarrow \top}$
  return $\mathsf{T}_2^+[K,M]$
else $C \twoheadleftarrow \{0,1\}^n \setminus \mathsf{Im}_1^+[K]$
if $C \in \mathsf{Im}_2^+[K]$: $\boxed{\mathsf{Bad}_{\mathrm{col}_1} \leftarrow \top}$
  $C \twoheadleftarrow \{0,1\}^n \setminus \mathsf{Im}_1^+[K] \cup \mathsf{Im}_2^+[K]$
$\mathsf{T}_1^+[K,M] := C$
$\mathsf{T}_1^-[K,C] := M$
$\mathsf{Im}_1^+[K] \leftarrow \mathsf{Im}_1^+[K] : C$
$\mathsf{Im}_1^-[K] \leftarrow \mathsf{Im}_1^-[K] : M$
return C

$\underline{\text{Proc. } \mathsf{iD}(K,C)}$:
if $\mathsf{T}_1^-[K,C]$: return $\mathsf{T}_1^-[K,C]$
if $\mathsf{T}_2^-[K,C] \neq \bot$: $\boxed{\mathsf{Bad}_{\mathrm{kup}} \leftarrow \top}$
  return $\mathsf{T}_2^-[K,C]$
else $M \twoheadleftarrow \{0,1\}^n \setminus \mathsf{Im}_1^-[K]$
if $M \in \mathsf{Im}_2^-[K]$: $\boxed{\mathsf{Bad}_{\mathrm{col}_1} \leftarrow \top}$
  $M \twoheadleftarrow \{0,1\}^n \setminus \mathsf{Im}_1^-[K] \cup \mathsf{Im}_2^-[K]$
$\mathsf{T}_1^-[K,C] := M$
$\mathsf{T}_1^+[K,M] := C$
$\mathsf{Im}_1^-[K] \leftarrow \mathsf{Im}_1^-[K] : M$
$\mathsf{Im}_1^+[K] \leftarrow \mathsf{Im}_1^+[K] : C$
return M

$\underline{\text{Proc. } \mathrm{KCE}\textsc{nc}(\xi^e)}$:
$(K,M) \leftarrow \xi^e(K^*)$
if $(K,M) \in \mathsf{ML}$: return $\bot$
if $\mathsf{T}_2^+[K,M]$: return $\mathsf{T}_2^+[K,M]$
if $\mathsf{T}_1^+[K,M]$: $\boxed{\mathsf{Bad}_{\mathrm{kup}} \leftarrow \top}$
  return $\mathsf{T}_1^+[K,M]$
else $C \twoheadleftarrow \{0,1\}^n \setminus \mathsf{Im}_2^+[K]$
if $C \in \mathsf{Im}_1^+[K]$: $\boxed{\mathsf{Bad}_{\mathrm{col}_1} \leftarrow \top}$
  $C \twoheadleftarrow \{0,1\}^n \setminus \mathsf{Im}_2^+[K] \cup \mathsf{Im}_1^+[K]$
$\mathsf{T}_2^+[K,M] := C$
$\mathsf{T}_2^-[K,C] := M$
$\mathsf{Im}_2^+[K] \leftarrow \mathsf{Im}_2^+[K] : C$
$\mathsf{Im}_2^-[K] \leftarrow \mathsf{Im}_2^-[K] : M$
$\mathsf{CL} \leftarrow \mathsf{CL} : (K,C)$
return C

$\underline{\text{Proc. } \mathrm{KCD}\textsc{ec}(\xi^d)}$:
$(K,C) \leftarrow \xi^d(K^*)$
if $(K,C) \in \mathsf{CL}$: return $\bot$
if $\mathsf{T}_2^-[K,C]$: return $\mathsf{T}_2^-[K,C]$
if $\mathsf{T}_1^-[K,C]$: $\boxed{\mathsf{Bad}_{\mathrm{kup}} \leftarrow \top}$
  return $\mathsf{T}_1^-[K,C]$
else $M \twoheadleftarrow \{0,1\}^n \setminus \mathsf{Im}_2^-[K]$
if $M \in \mathsf{Im}_1^-[K]$: $\boxed{\mathsf{Bad}_{\mathrm{col}_1} \leftarrow \top}$
  $M \twoheadleftarrow \{0,1\}^n \setminus \mathsf{Im}_2^-[K] \cup \mathsf{Im}_1^-[K]$
$\mathsf{T}_2^-[K,C] := M$
$\mathsf{T}_2^+[K,M] := C$
$\mathsf{Im}_2^-[K] \leftarrow \mathsf{Im}_2^-[K] : M$
$\mathsf{Im}_2^+[K] \leftarrow \mathsf{Im}_2^+[K] : C$
$\mathsf{ML} \leftarrow \mathsf{ML} : (K,M)$
return M

$\underline{\mathsf{Game}_2^{\Xi^e,\Xi^d,\mathcal{A}}}$:
$b \leftarrow 1$; $K^* \twoheadleftarrow \{0,1\}^k$
$(\mathsf{iE},\mathsf{iD}) \twoheadleftarrow \mathrm{Block}(k,n)$
$b' \twoheadleftarrow \mathcal{A}^{\mathsf{iE},\mathsf{iD},\mathrm{KCE}\textsc{nc},\mathrm{KCD}\textsc{ec}}$
return $(b' = 1)$

$\underline{\text{Proc. } \mathsf{iE}(K,M)}$:
if $\mathsf{T}_1^+[K,M]$: return $\mathsf{T}_1^+[K,M]$
if $\mathsf{T}_2^+[K,M]$: $\mathsf{Bad}_{\mathrm{kup}} \leftarrow \top$
  $\boxed{\text{// return } \mathsf{T}_2^+[K,M]}$
$C \twoheadleftarrow \{0,1\}^n \setminus \mathsf{Im}_1^+[K]$
if $C \in \mathsf{Im}_2^+[K]$: $\mathsf{Bad}_{\mathrm{col}_1} \leftarrow \top$
  $\boxed{\text{// } C \twoheadleftarrow \{0,1\}^n \setminus \mathsf{Im}_1^+[K] \cup \mathsf{Im}_2^+[K]}$
$\mathsf{T}_1^+[K,M] := C$
$\mathsf{T}_1^-[K,C] := M$
$\mathsf{Im}_1^+[K] \leftarrow \mathsf{Im}_1^+[K] : C$
$\mathsf{Im}_1^-[K] \leftarrow \mathsf{Im}_1^-[K] : M$
return C

$\underline{\text{Proc. } \mathsf{iD}(K,C)}$:
if $\mathsf{T}_1^-[K,C]$: return $\mathsf{T}_1^-[K,C]$
if $\mathsf{T}_2^-[K,C] \neq \bot$: $\mathsf{Bad}_{\mathrm{kup}} \leftarrow \top$
  $\boxed{\text{// return } \mathsf{T}_2^-[K,C]}$
$M \twoheadleftarrow \{0,1\}^n \setminus \mathsf{Im}_1^-[K]$
if $M \in \mathsf{Im}_2^-[K]$: $\mathsf{Bad}_{\mathrm{col}_1} \leftarrow \top$
  $\boxed{\text{// } M \twoheadleftarrow \{0,1\}^n \setminus \mathsf{Im}_1^-[K] \cup \mathsf{Im}_2^-[K]}$
$\mathsf{T}_1^-[K,C] := M$
$\mathsf{T}_1^+[K,M] := C$
$\mathsf{Im}_1^-[K] \leftarrow \mathsf{Im}_1^-[K] : M$
$\mathsf{Im}_1^+[K] \leftarrow \mathsf{Im}_1^+[K] : C$
return M

$\underline{\text{Proc. } \mathrm{KCE}\textsc{nc}(\xi^e)}$:
$(K,M) \leftarrow \xi^e(K^*)$
if $(K,M) \in \mathsf{ML}$: return $\bot$
if $\mathsf{T}_2^+[K,M]$: return $\mathsf{T}_2^+[K,M]$
if $\mathsf{T}_1^+[K,M]$: $\mathsf{Bad}_{\mathrm{kup}} \leftarrow \top$
  $\boxed{\text{// return } \mathsf{T}_1^+[K,M]}$
$C \twoheadleftarrow \{0,1\}^n \setminus \mathsf{Im}_2^+[K]$
if $C \in \mathsf{Im}_1^+[K]$: $\mathsf{Bad}_{\mathrm{col}_1} \leftarrow \top$
  $\boxed{\text{// } C \twoheadleftarrow \{0,1\}^n \setminus \mathsf{Im}_2^+[K] \cup \mathsf{Im}_1^+[K]}$
$\mathsf{T}_2^+[K,M] := C$
$\mathsf{T}_2^-[K,C] := M$
$\mathsf{Im}_2^+[K] \leftarrow \mathsf{Im}_2^+[K] : C$
$\mathsf{Im}_2^-[K] \leftarrow \mathsf{Im}_2^-[K] : M$
$\mathsf{CL} \leftarrow \mathsf{CL} : (K,C)$
return C

$\underline{\text{Proc. } \mathrm{KCD}\textsc{ec}(\xi^d)}$:
$(K,C) \leftarrow \xi^d(K^*)$
if $(K,C) \in \mathsf{CL}$: return $\bot$
if $\mathsf{T}_2^-[K,C]$: return $\mathsf{T}_2^-[K,C]$
if $\mathsf{T}_1^-[K,C]$: $\mathsf{Bad}_{\mathrm{kup}} \leftarrow \top$
  $\boxed{\text{// return } \mathsf{T}_1^-[K,C]}$
$M \twoheadleftarrow \{0,1\}^n \setminus \mathsf{Im}_2^-[K]$
if $M \in \mathsf{Im}_1^-[K]$: $\mathsf{Bad}_{\mathrm{col}_1} \leftarrow \top$
  $\boxed{\text{// } M \twoheadleftarrow \{0,1\}^n \setminus \mathsf{Im}_2^-[K] \cup \mathsf{Im}_1^-[K]}$
$\mathsf{T}_2^-[K,C] := M$
$\mathsf{T}_2^+[K,M] := C$
$\mathsf{Im}_2^-[K] \leftarrow \mathsf{Im}_2^-[K] : M$
$\mathsf{Im}_2^+[K] \leftarrow \mathsf{Im}_2^+[K] : C$
$\mathsf{ML} \leftarrow \mathsf{ML} : (K,M)$
return M

Figure 5.2 – Game_0, Game_1, and Game_2 used to prove the KCA-security of the ideal cipher.

$\underline{\mathsf{Game}_3^{\Xi^e,\Xi^d,\mathcal{A}}}$:
Unmodified

Proc. $\mathsf{iE}(K,M)$:
Unmodified
Proc. $\mathsf{iD}(K,C)$:
Unmodified

Proc. $\textsc{KCEnc}(\xi^e)$:
$(K,M) \leftarrow \xi^e(K^*)$
if $(K,M) \in \mathsf{ML}$: return $\bot$
if $\mathsf{T}_2^+[K,M]$: $\boxed{\mathsf{Bad}_{\mathsf{cf}} \leftarrow \top}$
$\boxed{\text{// return } \mathsf{T}_2^+[K,M]}$
if $\mathsf{T}_1^+[K,M]$: $\mathsf{Bad}_{\mathsf{kup}} \leftarrow \top$
// return $\mathsf{T}_1^+[K,M]$
$C \twoheadleftarrow \{0,1\}^n$
if $C \in \mathsf{Im}_2^+[K]$: $\boxed{\mathsf{Bad}_{\mathsf{col}_2} \leftarrow \top}$
$\boxed{\text{// } C \twoheadleftarrow \{0,1\}^n \setminus \mathsf{Im}_2^+[K]}$
if $C \in \mathsf{Im}_1^+[K]$: $\mathsf{Bad}_{\mathsf{col}_1} \leftarrow \top$
// $C \twoheadleftarrow \{0,1\}^n \setminus \mathsf{Im}_2^+[K] \cup \mathsf{Im}_1^+[K]$
$\mathsf{T}_2^+[K,M] := C$
$\mathsf{T}_2^-[K,C] := M$
$\mathsf{Im}_2^+[K] \leftarrow \mathsf{Im}_2^+[K] : C$
$\mathsf{Im}_2^-[K] \leftarrow \mathsf{Im}_2^-[K] : M$
$\mathsf{CL} \leftarrow \mathsf{CL} : (K,C)$
return C

Proc. $\textsc{KCDec}(\xi^d)$:
$(K,C) \leftarrow \xi^d(K^*)$
if $(K,C) \in \mathsf{CL}$: return $\bot$
if $\mathsf{T}_2^-[K,C]$: $\boxed{\mathsf{Bad}_{\mathsf{cf}} \leftarrow \top}$
$\boxed{\text{// return } \mathsf{T}_2^-[K,C]}$
if $\mathsf{T}_1^-[K,C]$: $\mathsf{Bad}_{\mathsf{kup}} \leftarrow \top$
// return $\mathsf{T}_1^-[K,C]$
$M \twoheadleftarrow \{0,1\}^n$
if $M \in \mathsf{Im}_2^-[K]$: $\boxed{\mathsf{Bad}_{\mathsf{col}_2} \leftarrow \top}$
$\boxed{\text{// } M \twoheadleftarrow \{0,1\}^n \setminus \mathsf{Im}_2^-[K]}$
if $M \in \mathsf{Im}_1^-[K]$: $\mathsf{Bad}_{\mathsf{col}_1} \leftarrow \top$
// $M \twoheadleftarrow \{0,1\}^n \setminus \mathsf{Im}_2^-[K] \cup \mathsf{Im}_1^-[K]$
$\mathsf{T}_2^-[K,C] := M$
$\mathsf{T}_2^+[K,M] := C$
$\mathsf{Im}_2^-[K] \leftarrow \mathsf{Im}_2^-[K] : M$
$\mathsf{Im}_2^+[K] \leftarrow \mathsf{Im}_2^+[K] : C$
$\mathsf{ML} \leftarrow \mathsf{ML} : (K,M)$
return M

$\underline{\mathsf{Game}_4^{\Xi^e,\Xi^d,\mathcal{A}}}$:
Unmodified

Proc. $\mathsf{iE}(K,M)$:
Unmodified
Proc. $\mathsf{iD}(K,C)$:
Unmodified

Proc. $\textsc{KCEnc}(\xi^e)$:
$(K,M) \leftarrow \xi^e(K^*)$
$\boxed{\begin{array}{l}\mathsf{chk} \leftarrow \bot\\ \text{for } (\xi^d,M) \in \mathsf{DL}:\\ \quad \mathsf{chk} \leftarrow \mathsf{chk} \vee \mathsf{Det}(\xi^e,0^n,\xi^d,M,2)\\ \text{if } \mathsf{chk} \neq ((K,M) \in \mathsf{ML}):\\ \quad \mathsf{Bad}_{\mathsf{xkcd}} \leftarrow \top\\ \quad \mathsf{chk} \leftarrow ((K,M) \in \mathsf{ML})\\ \text{if chk: return } \bot\end{array}}$
if $\mathsf{T}_2^+[K,M]$: $\mathsf{Bad}_{\mathsf{cf}} \leftarrow \top$
// return $\mathsf{T}_2^+[K,M]$
if $\mathsf{T}_1^+[K,M]$: $\mathsf{Bad}_{\mathsf{kup}} \leftarrow \top$
// return $\mathsf{T}_1^+[K,M]$
$C \twoheadleftarrow \{0,1\}^n$
if $C \in \mathsf{Im}_2^+[K]$: $\mathsf{Bad}_{\mathsf{col}_2} \leftarrow \top$
// $C \twoheadleftarrow \{0,1\}^n \setminus \mathsf{Im}_2^+[K]$
if $C \in \mathsf{Im}_1^+[K]$: $\mathsf{Bad}_{\mathsf{col}_1} \leftarrow \top$
// $C \twoheadleftarrow \{0,1\}^n \setminus \mathsf{Im}_2^+[K] \cup \mathsf{Im}_1^+[K]$
$\mathsf{T}_2^+[K,M] := C$
$\mathsf{T}_2^-[K,C] := M$
$\mathsf{Im}_2^+[K] \leftarrow \mathsf{Im}_2^+[K] : C$
$\mathsf{Im}_2^-[K] \leftarrow \mathsf{Im}_2^-[K] : M$
$\mathsf{CL} \leftarrow \mathsf{CL} : (K,C)$
$\mathsf{EL} \leftarrow \mathsf{EL} : (\xi^e,C)$
return C

Proc. $\textsc{KCDec}(\xi^d)$:
$(K,C) \leftarrow \xi^d(K^*)$
$\boxed{\begin{array}{l}\mathsf{chk} \leftarrow \bot\\ \text{for } (\xi^e,C) \in \mathsf{EL}:\\ \quad \mathsf{chk} \leftarrow \mathsf{chk} \vee \mathsf{Det}(\xi^e,C,\xi^d,0^n,1)\\ \text{if } \mathsf{chk} \neq ((K,C) \in \mathsf{CL}):\\ \quad \mathsf{Bad}_{\mathsf{xkcd}} \leftarrow \top\\ \quad \mathsf{chk} \leftarrow ((K,C) \in \mathsf{CL})\\ \text{if chk: return } \bot\end{array}}$
if $\mathsf{T}_2^-[K,C]$: $\mathsf{Bad}_{\mathsf{cf}} \leftarrow \top$
// return $\mathsf{T}_2^-[K,C]$
if $\mathsf{T}_1^-[K,C]$: $\mathsf{Bad}_{\mathsf{kup}} \leftarrow \top$
// return $\mathsf{T}_1^-[K,C]$
$M \twoheadleftarrow \{0,1\}^n$
if $M \in \mathsf{Im}_2^-[K]$: $\mathsf{Bad}_{\mathsf{col}_2} \leftarrow \top$
// $M \twoheadleftarrow \{0,1\}^n \setminus \mathsf{Im}_2^-[K]$
if $M \in \mathsf{Im}_1^-[K]$: $\mathsf{Bad}_{\mathsf{col}_1} \leftarrow \top$
// $M \twoheadleftarrow \{0,1\}^n \setminus \mathsf{Im}_2^-[K] \cup \mathsf{Im}_1^-[K]$
$\mathsf{T}_2^-[K,C] := M$
$\mathsf{T}_2^+[K,M] := C$
$\mathsf{Im}_2^-[K] \leftarrow \mathsf{Im}_2^-[K] : M$
$\mathsf{Im}_2^+[K] \leftarrow \mathsf{Im}_2^+[K] : C$
$\mathsf{ML} \leftarrow \mathsf{ML} : (K,M)$
$\mathsf{DL} \leftarrow \mathsf{DL} : (\xi^d,M)$
return M

$\underline{\mathsf{Game}_5^{\Xi^e,\Xi^d,\mathcal{A}}}$:
Unmodified

Proc. $\mathsf{iE}(K,M)$:
Unmodified
Proc. $\mathsf{iD}(K,C)$:
Unmodified

Proc. $\textsc{KCEnc}(\xi^e)$:
$(K,M) \leftarrow \xi^e(K^*)$
$\mathsf{chk} \leftarrow \bot$
for $(\xi^d,M) \in \mathsf{DL}$:
$\quad \mathsf{chk} \leftarrow \mathsf{chk} \vee \mathsf{Det}(\xi^e,0^n,\xi^d,M,2)$
if $\mathsf{chk} \neq ((K,M) \in \mathsf{ML})$:
$\quad \mathsf{Bad}_{\mathsf{xkcd}} \leftarrow \top$
$\boxed{\text{// } \mathsf{chk} \leftarrow ((K,M) \in \mathsf{ML})}$
if chk: return $\bot$
if $\mathsf{T}_2^+[K,M]$: $\mathsf{Bad}_{\mathsf{cf}} \leftarrow \top$
// return $\mathsf{T}_2^+[K,M]$
if $\mathsf{T}_1^+[K,M]$: $\mathsf{Bad}_{\mathsf{kup}} \leftarrow \top$
// return $\mathsf{T}_1^+[K,M]$
$C \twoheadleftarrow \{0,1\}^n$
if $C \in \mathsf{Im}_2^+[K]$: $\mathsf{Bad}_{\mathsf{col}_2} \leftarrow \top$
// $C \twoheadleftarrow \{0,1\}^n \setminus \mathsf{Im}_2^+[K]$
if $C \in \mathsf{Im}_1^+[K]$: $\mathsf{Bad}_{\mathsf{col}_1} \leftarrow \top$
// $C \twoheadleftarrow \{0,1\}^n \setminus \mathsf{Im}_2^+[K] \cup \mathsf{Im}_1^+[K]$
$\mathsf{T}_2^+[K,M] := C$
$\mathsf{T}_2^-[K,C] := M$
$\mathsf{Im}_2^+[K] \leftarrow \mathsf{Im}_2^+[K] : C$
$\mathsf{Im}_2^-[K] \leftarrow \mathsf{Im}_2^-[K] : M$
$\mathsf{CL} \leftarrow \mathsf{CL} : (K,C)$
$\mathsf{EL} \leftarrow \mathsf{EL} : (\xi^e,C)$
return C

Proc. $\textsc{KCDec}(\xi^d)$:
$(K,C) \leftarrow \xi^d(K^*)$
$\mathsf{chk} \leftarrow \bot$
for $(\xi^e,C) \in \mathsf{EL}$:
$\quad \mathsf{chk} \leftarrow \mathsf{chk} \vee \mathsf{Det}(\xi^e,C,\xi^d,0^n,1)$
if $\mathsf{chk} \neq ((K,C) \in \mathsf{CL})$:
$\quad \mathsf{Bad}_{\mathsf{xkcd}} \leftarrow \top$
$\boxed{\text{// } \mathsf{chk} \leftarrow ((K,C) \in \mathsf{CL})}$
if chk: return $\bot$
if $\mathsf{T}_2^-[K,C]$: $\mathsf{Bad}_{\mathsf{cf}} \leftarrow \top$
// return $\mathsf{T}_2^-[K,C]$
if $\mathsf{T}_1^-[K,C]$: $\mathsf{Bad}_{\mathsf{kup}} \leftarrow \top$
// return $\mathsf{T}_1^-[K,C]$
$M \twoheadleftarrow \{0,1\}^n$
if $M \in \mathsf{Im}_2^-[K]$: $\mathsf{Bad}_{\mathsf{col}_2} \leftarrow \top$
// $M \twoheadleftarrow \{0,1\}^n \setminus \mathsf{Im}_2^-[K]$
if $M \in \mathsf{Im}_1^-[K]$: $\mathsf{Bad}_{\mathsf{col}_1} \leftarrow \top$
// $M \twoheadleftarrow \{0,1\}^n \setminus \mathsf{Im}_2^-[K] \cup \mathsf{Im}_1^-[K]$
$\mathsf{T}_2^-[K,C] := M$
$\mathsf{T}_2^+[K,M] := C$
$\mathsf{Im}_2^-[K] \leftarrow \mathsf{Im}_2^-[K] : M$
$\mathsf{Im}_2^+[K] \leftarrow \mathsf{Im}_2^+[K] : C$
$\mathsf{ML} \leftarrow \mathsf{ML} : (K,M)$
$\mathsf{DL} \leftarrow \mathsf{DL} : (\xi^d,M)$
return M

Figure 5.3 – Game_3, Game_4, and Game_5 used to prove the KCA-security of the ideal cipher.

The probability of setting $\mathsf{Bad}_{\mathrm{kup}}$ under iD is analyzed similarly to cases (a) and (b) above via reductions to the multi-shot unpredictability of Ξ^d and the multi-shot key unpredictability of Ξ^e. The probability of setting $\mathsf{Bad}_{\mathrm{col}_1}$ under iD is at most $q(q_e + q_d)/2^n$.[3]

Next we consider the probability of setting $\mathsf{Bad}_{\mathrm{cf}}$ or $\mathsf{Bad}_{\mathrm{col}_2}$. Flag $\mathsf{Bad}_{\mathrm{cf}}$ can be set under KCEnc as a result of a query ξ_0^e if either (a) a previous distinct query ξ_1^e to KCEnc resulted in $\xi_0^e(K^*) = \xi_1^e(K^*)$, or (b), a previous query ξ_1^d to KCDec resulted in $\xi_0^e(K^*) = (\xi_1^d|_1(K*), M)$, where M is a random value chosen by KCDec in a previous query.

In case (a) by collecting the list of ξ^e queried to KCEnc we can break the multi-shot claw-freeness of Ξ^e (simulating the oracles as in Game_5). For case (b) to take place it must be that Det classified the query as valid. However, this leads to setting $\mathsf{Bad}_{\mathrm{xkcd}}$ since M was returned from the KCDec oracle and hence $(\xi_1^d|_1(K^*), M)$ appears on ML but Det did not detect this. We analyze the probability of setting $\mathsf{Bad}_{\mathrm{xkcd}}$ below. The probability of setting $\mathsf{Bad}_{\mathrm{col}_2}$ in KCEnc can be upper bounded by $q_e(q_e + q_d)/2^n$.

The probability of setting $\mathsf{Bad}_{\mathrm{cf}}$ under KCDec is analyzed as above by a reduction to the multi-shot claw-freeness of Ξ^d (or setting $\mathsf{Bad}_{\mathrm{xkcd}}$). The probability of setting $\mathsf{Bad}_{\mathrm{col}_2}$ in KCDec is upper bounded by $q_d(q_e + q_d)/2^n$.

Finally we consider the probability of setting $\mathsf{Bad}_{\mathrm{xkcd}}$. Flag $\mathsf{Bad}_{\mathrm{xkcd}}$ is set exactly when $\xi^e(K^*)$ appears on ML as a result of a query ξ^d to KCDec with response M and Det says it does not, or that it does not not and Det says it does. We can bound the probability of setting $\mathsf{Bad}_{\mathrm{xkcd}}$ under KCEnc by a reduction to the multi-shot xkcd property by collecting all queries ξ^e made to KCEnc and all query-answer pairs (ξ^d, M) made to and received form KCDec in two lists and outputting them lists in the multi-shot xkcd game. The probability of setting of $\mathsf{Bad}_{\mathrm{xkcd}}$ under KCDec is analyzed analogously.

The theorem now follows by putting the bounds established above together.

□

The above theorem shows that key-unpredictability, claw-freeness, and xkcd are sufficient for achieving KCA security in the ideal-cipher model. This raises the question whether or not they are also *necessary*. Key-unpredictability is. Any adversary that can predict the key output of ξ^e can be used to win the KC game as follows:

1. ask for an encryption under $\xi^e(K^*)$ to get a ciphertext C;
2. predict $\xi^e|_1(K^*)$ as K;
3. compute $\mathsf{iD}(K, C)$ to get M;

3. The above cases take care of collisions between wires $(1/2, 3)$, $(1/2, 4)$ in the picture above.

4. compute $\mathsf{iE}(K, M)$ to get C';
5. return $b' := 1$ if $C = C'$ and $b' := 0$ otherwise.

It is easily seen that this attack returns $b' = 1$ with the same probability of predicting keys when $b = 1$. On the other hand, it only succeeds with negligible probability when $b' = 0$ as the encryption oracle and the challenge oracle use independent ideal ciphers. We deal with such attacks in case (b) of $(\$^+, \mathsf{iE}^{\pm})$ in the proof.

Claw-freeness, on the other hand, is not necessary. Consider a CDF set consisting of exactly two functions that collide with probability $1/2$, e.g., $K \mapsto K$ and $K \mapsto \min(K, \overline{K})$. Then a security proof for the ideal cipher can be easily given: guess with probability $1/2$ if $\min(K, \overline{K}) = K$. Next depending on the guess simulate the key-correlated oracles either using the same set of ideal cipher oracles or independent ones. However, one can also exhibit CDF sets that are not claw-free and *do* lead to a KCA on the ideal cipher. Indeed, any set of $k + 1$ functions that has claws with the identity function depending on whether or not K_i, the i-th bit of the key, is 1 can be used to recover the key one bit at a time. In practice this translates to recovering the key by observing the repetition patters in the outputs.

Finally, there is also a pair of CDF sets which are not xkcd and can be used to attack the ideal cipher. Let $\xi^e(K) := (K, 0^n)$ and $\xi^d[i, C](K) := (K \oplus (K_i)^n =: K', C)$, that is, $K' = \overline{K}$ if $K_i = 1$ and $K' = K$ otherwise. Let C be the answer to $\text{KCE}_{\text{NC}}(\xi^e)$. Now for $i = 1, \ldots, k$, by checking whether $\text{KCD}_{\text{EC}}(\xi^d[i, C])$ returns $\perp$ or not, one can again recover the key one bit at a time. In practice this translates to observing the equality pattern of the decrypted ciphertexts with 0^n.

Theorem 5.1 and Proposition 4.1 can be used to recover known feasibility results for RKA and KDM security of the ideal cipher as follows.

RKA SECURITY. An RKA-only CDF set Ξ associated to a set Φ is key-unpredictable iff Φ is unpredictable in the sense of [BK03]. Indeed, a predictor for a related-key derivation function ϕ can be converted to a key-predictor for $\xi\colon K \mapsto (\phi(K), M)$ for any fixed M. Conversely, a key-predictor for $(\phi(K), M)$ in particular predicts $\phi(K)$. Similarly, Ξ is claw-free iff Φ is claw-free in the sense of [BK03]. A claw in Φ can be immediately converted to a claw in Ξ (by adding any message M). Conversely, a claw $((\phi_1(K), M_1), (\phi_2(K), M_2))$ is necessarily also a claw between ϕ_1 and ϕ_2. For the (funny) xkcd property, Det given $((\phi_1, M_1), C, (\phi_2, C_2), M, i)$ needs to check if (1) $\phi_1(K) = \phi_2(K)$ and (2) if $C = C_2$ when $i = 1$ or if $M = M_1$ when $i = 2$. By claw-freeness, (1) would only occur if $\phi_1 = \phi_2$; the second condition on the other hand is trivial to check. Thus, claw-freeness implies xkcd.

KDM SECURITY. A KDM-only Ξ associated to a KDM set Ψ is always key-unpredictable with single-shot advantage $1/2^n$. This is because KDM-only functions leave the random key unmodified, which remains information-theoretically hidden from the adversary in the key-

unpredictability game. Moreover, and similarly to the RKA case, the set Ξ is claw-free iff Ψ is. Once again, this is because KDM-only CDFs do not modify the key component and hence any claw must be on the message part. For the xckd property, given $((id, \psi_1), C, (id, C_2), M, i)$, algorithm Det needs to check (1) if $id(K) = id(K)$, which always holds; and (2) if $C_2 = C$ when $i = 1$ or if $\psi_1(K) = M$ when $i = 2$. The first of these is trivially checkable. For the second, two cases could occur: either $\psi_1(K)$ is a constant function mapping to M_1, in which case Det can easily check if $M_1 = M$; or ψ_1 is non-constant. However, since Ψ contains all constant functions, ψ_1 and M can be used to break claw-freeness. Hence in this case Det rejects. So similarly to the RKA setting, claw-freeness implies xckd.

KCA SECURITY. We now show that the reach of the above feasibility result for the ideal cipher extends beyond RKA and KDM security. Suppose $k = n$ and recall the set

$$\Xi^{\oplus} := \left\{ K \mapsto (K \oplus \Delta_1, \alpha \cdot K \oplus \Delta_2) : \Delta_1, \Delta_2 \in \mathcal{M}, \alpha \in \{0, 1\} \right\} . \quad (5.1)$$

This set captures related keys that are computed as offsets, key-*in*dependent messages when $\alpha = 0$, and key-dependent messages which are offsets of the key when $\alpha = 1$. We identify functions in this set with tuples $(\Delta_1, \Delta_2, \alpha)$.

This set is key-unpredictable with single-shot advantage at most $1/2^n$, since xor-ing with Δ_1 acts as a permutation and K remains information-theoretically hidden. It is also claw-free with single-shot advantage at most $1/2^n$: If the offsets Δ_1 and Δ_1' for two functions ξ, ξ' are different, there are no claws. If they are the same, and $\alpha = \alpha'$, then a claw also implies $\Delta_2 = \Delta_2'$, making the two functions identical. So $\alpha \neq \alpha'$ is necessary for claws, in which case the functions collide on a random key with probability $1/2^n$.

The set is also xkcd with single-shot advantage $2/2^n$ with respect to the following detector (recall that $\mathsf{Det}(\xi^e, C, \xi^d, M, 1)$ should check whether $(\xi^e|_1(K), C) = \xi^d(K)$):

$$\begin{aligned} &\mathsf{Det}_{\oplus}\big((\Delta_1^e, \Delta_2^e, \alpha^e), C, (\Delta_1^d, \Delta_2^d, \alpha^d), M, i\big) : \\ &\qquad \text{if } i = 1 : \ \text{if } \Delta_1^e = \Delta_1^d \wedge \alpha^d = 0 \wedge \Delta_2^d = C \text{: return "Yes"} \\ &\qquad \text{if } i = 2 : \ \text{if } \Delta_1^e = \Delta_1^d \wedge \alpha^e = 0 \wedge \Delta_2^e = M \text{: return "Yes"} \\ &\qquad \text{else return "No"} \end{aligned} \quad (5.2)$$

Consider case $i = 1$ when $\mathsf{Det}_{\oplus}$ is trying to decide if $(K \oplus \Delta_1^e, C) = (K \oplus \Delta_1^d, \alpha^d \cdot K \oplus \Delta_2^d)$. If $\mathsf{Det}_{\oplus}$ returns "Yes", then both the key components and the ciphertext components collide; the answer is thus correct. Suppose it returns "No". If $\Delta_1^e \neq \Delta_1^d$, the key components do not collide and the answer is correct. Suppose $\Delta_1^e = \Delta_1^d$: if $\alpha^d = 1$, the probability that $\alpha^d \cdot K \oplus \Delta_2^d = C$ is $1/2^n$ since $\oplus$ acts a permutation and K is random; if $\alpha^d = 0$, then $\mathsf{Det}_{\oplus}$ checks if $\Delta_2^d = C$ and hence its

answer is correct. Case $i = 2$ is dealt with similarly and we obtain overall single-shot advantage of at most $2/2^n$.

Theorem 5.1 thus implies that the ideal cipher is $(\Xi^{\oplus}, \Xi^{\oplus})$-KC-CCA secure.

5.2 KCA SECURITY OF 3-ROUND EVEN–MANSOUR

In this section we show that in contrast to the two-round case the three-round Even–Mansour cipher with *reuse of keys and independent permutations* $\mathsf{EM}^{\mathsf{P}_1,\mathsf{P}_2,\mathsf{P}_3}[K,K,K,K]$, defined in Sec. 4.4.2, is $(\Xi^{\oplus}, \Xi^{\oplus})$-KC-CCA secure, with $\Xi^{\oplus}$ as defined in (5.1). We note that, for $\Phi^{\oplus}, \Psi^{\oplus}$ as in (4.2), this construction is known to be both $\Phi^{\oplus}$-RK-CCA and $\Psi^{\oplus}$-KDM-CCA secure [FP15; FKV17]. We build on these works to show KC-CCA security in the theorem.

Theorem 5.2 (KCA security of 3-round EM). $\mathsf{EM}^{\mathsf{P}_1,\mathsf{P}_2,\mathsf{P}_3}[K,K,K,K]$ *is* $(\Xi^{\oplus}, \Xi^{\oplus})$-KC-CCA *secure in the random-permutation model for* P_i*. More precisely, for any adversary* $\mathcal{A}$ *against the* $(\Xi^{\oplus}, \Xi^{\oplus})$-KC-CCA *security of* $\mathsf{EM}^{\mathsf{P}_1,\mathsf{P}_2,\mathsf{P}_3}[K,K,K,K]$ *we have*

$$\mathbf{Adv}^{\text{kc-cca}}_{\mathsf{EM}^{\mathsf{P}_1,\mathsf{P}_2,\mathsf{P}_3}[K,K,K,K]}(\Xi^{\oplus}, \Xi^{\oplus}, \mathcal{A}) \leq 2 \cdot \left(qq_e + qq_d + q_e^2 + q_d^2 + q_e q_d\right) / \left(2^n - q\right) ,$$

where q *is the maximum number of queries of* $\mathcal{A}$ *to* $\mathsf{P}_i^{\pm}$ *(over all* $i = 1, 2, 3$*) and* q_e *and* q_d *are the maximum number of encryption and decryption queries of* $\mathcal{A}$*, respectively.*

Proof. To analyze the KC-CCA security of Even–Mansour, our strategy will be similar to that in Theorem 5.1 and those in [FP15; FKV17]. As in these proofs, we replace the last-round permutation P_3^{+} in KCEnc queries with a forgetful random oracle and also replace the first-round permutation P_1^{-} in KCDec with another forgetful random oracle. After these replacements the outputs of KCEnc and KCDec will be fully randomized and independent of the inputs; the game is thus independent of the challenge bit b.

When simulating KCEnc and KCDec with forgetful oracles, we still need to ensure that illegal queries are answered correctly with $\perp$. As in the proof of Theorem 5.1, we do this using a detector. In particular, we use $\mathsf{Det}_{\oplus}$ as in (5.2), which on input $((\Delta_1, \Delta_2, \alpha), C, (\Delta_1', \Delta_2', \alpha'), M, 1)$ returns "Yes" iff $\Delta_1 = \Delta_1'$ and $\alpha' = 0$ and $\Delta_2' = C$ (and "No" otherwise); and on input $((\Delta_1, \Delta_2, \alpha), C, (\Delta_1', \Delta_2', \alpha'), M, 2)$ returns "Yes" iff $\Delta_1 = \Delta_1'$ and $\alpha = 0$ and $\Delta_2 = M$.

(1) For each decryption query ξ^d, KCDec runs $\mathsf{Det}_{\oplus}(\xi^e, C, \xi^d, 0^n, 1)$ for all previous queries ξ^e to KCEnc, which were answered with C. If Det says "Yes" for any of these, $\text{KCDec}_{\oplus}$ returns $\perp$, else it outputs a random value from the domain.

(2) For each encryption query ξ^e, KCEnc runs $\mathsf{Det}_{\oplus}(\xi^e, 0^n, \xi^d, M, 2)$ for all previous queries ξ^d to KCDec, which were answered with M. If $\mathsf{Det}_{\oplus}$ says "Yes" for any of these, KCEnc returns $\perp$, else it answers with a random value.

These replacements, however, raise the question how P_1^+ in KCEnc and P_3^- in KCDec are treated (since their respective counterparts in KCDec and KCEnc were replaced). Previous works simply leave these oracles unchanged. Our proof diverges in this aspect, because the key-dependency of *both* keys and messages gives rise to queries that cannot be guaranteed to be disjoint from the forgetful oracle, or discarded as trivial/illegal as in the KDM setting [FKV17], or not result in inconsistency due to the middle permutation P_2 being replaced in the RKA setting [FP15].

For the KCA setting we will replace P_3^- in KCDec (and analogously for P_1^+ in KCEnc) as follows. When $\mathsf{Det}_\oplus$ says "No" on a query $\xi^d = (\Delta_1^d, \Delta_2^d, \alpha^d)$, the game checks if each previous query $\xi^e = (\Delta_1^e, \Delta_2^e, \alpha^e)$ to KCEnc with output value C satisfies

$$C \oplus \Delta_1^e = \Delta_2^d \oplus \Delta_1^d \,.$$

If so, oracle $\tilde{\mathsf{P}}_3^-$ outputs the input U to $\$^+$ that was used to compute C. If multiple values are found, $\tilde{\mathsf{P}}_3^-$ returns $\perp$. If the latter is not the case, $\tilde{\mathsf{P}}_3^-$ is sampled as an independent permutation and consistent with P_3. Permutation $\mathsf{P}_3^\pm$, when directly queried, is also sampled consistently with $\tilde{\mathsf{P}}_3^-$ on all those entries that are not kept consistent with $\$^+$. More precisely, oracles $\$^+$, $\mathsf{P}_3^\pm$ and $\tilde{\mathsf{P}}_3^-$ write to independent lists L_1, L_2 and L_3. Oracle $\$^+$ checks consistency with none of the lists, oracle $\mathsf{P}_3^\pm$ checks consistency with L_2 and L_3 and oracle $\tilde{\mathsf{P}}_3^-$ checks consistency with L_1, L_2 and L_3.

Oracle P_1^+ in KCEnc is replaced with $\tilde{\mathsf{P}}_1^+$ similarly by checking if $M \oplus \Delta_1^d = \Delta_2^e \oplus \Delta_1^e$ and outputting the input U to $\$^-$ that was used to compute M.

As a result of the above replacements, the following inconsistencies could arise:

(a) Identical queries to a permutation oracle are answered differently due to a replacement.
(b) Different queries to a permutation oracle and its replacement are answered identically (and hence do not respect permutativity).
(c) Multiple candidates for U are found when sampling $\tilde{\mathsf{P}}_3^-$ or $\tilde{\mathsf{P}}_1^+$.

We now bound the probabilities of these events.

MULTIPLE CANDIDATES This event will happen due to: (1) a collision in the outputs of $\$^+$, which happens with probability $1/2^n$ for each pair of queries to KCEnc; or (2) a collision in the output of $\$^+$ and an output entry in $\mathsf{P}_3^\pm$. This event is equivalent to a collision in the outputs of $\$^+$ and $\mathsf{P}_3^\pm$, which pertains to case (b) and is analyzed next.

($\$^+, \mathsf{P}_3^\pm$) A direct $\mathsf{P}_3^\pm$ call in either direction collides with a point queried to P_3^+ due to a query $\xi^e = (\Delta_1, \Delta_2, \alpha)$ to KCEnc. Suppose the direct query results in $V = \mathsf{P}_3^+(U)$ or $U = \mathsf{P}_3^-(V)$. There are now two cases. (a) The inputs collide, i.e.,

$$\mathsf{P}_2\big(\mathsf{P}_1(\Delta_1 \oplus \Delta_2 \oplus (1-\alpha)\cdot K^*) \oplus \Delta_1 \oplus K^*\big) \oplus \Delta_1 \oplus K^* = U \,. \quad (5.3)$$

If $Y := \mathsf{P}_1(\Delta_1 \oplus \Delta_2 \oplus (1-\alpha)\cdot K^*) \oplus \Delta_1 \oplus K^*$ was queried to P_2 then the adversary can compute $K^* = \mathsf{P}_2(Y) \oplus \Delta_1 \oplus U$, which is only possible with probability $1/2^n$, since the key remains information-theoretically hidden. If it was not queried, the probability that (5.3) holds is also $1/(2^n - q)$, since P_2 is a random permutation, where outputs are chosen in a set of size at least $2^n - q$.

The other case is that (b) the outputs collide. If the $\$^+$ query comes after the direct query, the probability of this event is $1/2^n$. Now suppose the $\$^+$ query comes first and outputs C; thus due to the collision we have that $C = V \oplus K^* \oplus \Delta_1$. But in this case the adversary can compute $K^* = V \oplus \Delta_1$, which happens with probability at most $1/2^n$ (as the key remains information theoretically hidden).

$(\$^-, \mathsf{P}_1^{\pm})$ A direct $\mathsf{P}_1^{\pm}$ call in either direction collides with a point queried to $\$^-$ due to a query ξ^d to KCDec. This event is analyzed analogously to case $(\$^+, \mathsf{P}_3^{\pm})$ above.

$(\$^+, \$^+)$ The are two cases: (a) two inputs collide or (b) two outputs collide. In case (a) two distinct calls $\xi^e = (\Delta_1, \Delta_2, \alpha)$ and $(\xi^e)' = (\Delta_1', \Delta_2', \alpha')$ to the KCEnc oracle result in querying $\$^+$ on the same input. For this event to occur we must have that

$$\mathsf{P}_2\Big(\overbrace{\mathsf{P}_1(\underbrace{\Delta_1 \oplus \Delta_2 \oplus (1-\alpha)\cdot K^*}_{=:X}) \oplus \Delta_1 \oplus K^*}^{=:Y}\Big) \oplus \Delta_1$$
$$= \mathsf{P}_2\Big(\underbrace{\mathsf{P}_1(\overbrace{\Delta_1' \oplus \Delta_2' \oplus (1-\alpha')\cdot K^*}^{=:X'}) \oplus \Delta_1' \oplus K^*}_{=:Y'}\Big) \oplus \Delta_1' \,. \quad (5.4)$$

If $Y = Y'$ then $\mathsf{P}_1(X) \oplus \Delta_1 = \mathsf{P}_1(X') \oplus \Delta_1'$ and from (5.4) we get $\Delta_1 = \Delta_1'$. Since P_1 is a permutation, this implies that $X = X'$. If further $\alpha = \alpha'$ then we would also have $\Delta_2 = \Delta_2'$, which means $\xi^e = (\xi^e)'$, contradicting distinctness of queries. Thus $\alpha \neq \alpha'$ and hence $\Delta_2 \oplus \Delta_2' = K^*$ (since $X = X'$), which only happens with probability $1/2^n$ as it leads to guessing the key.

Assume now that $Y \neq Y'$.

— If the adversary does not query P_2 on either Y or Y' then $\mathsf{P}_2(Y)$ or $\mathsf{P}_2(Y')$ would be distributed randomly and independently of the adversary's view. Hence the probability that the above equality holds would be at most $1/(2^n - q)$ for each pair of queries.

— Suppose both Y and Y' are queried to P_2. Then P_1 must have been queried on both X or X', since otherwise the probability of computing Y or Y' would be at most $1/(2^n - q)$. If both X and X' are queried to P_1 and $\alpha = 0$ or $\alpha' = 0$, then for each query the probability of querying X (or X') to P_1 is $1/2^n$, because K^* is information-theoretically hidden. The

remaining case is $\alpha = \alpha' = 1$. Since the adversary knows Y (it was queried to P_2), it can compute $K^* = Y \oplus \mathsf{P}_1(\Delta_1 \oplus \Delta_2) \oplus \Delta_1$. Again, since K^* is information-theoretically hidden, this can only happen with probability $1/2^n$.

In case (b) outputs of $\$^+$ collide with probability $1/2^n$ for each pair of queries.

$(\$^-, \$^-)$ Two distinct $\xi^d, (\xi^d)' \in \Xi^d$ to KCDec result in querying $\$^-$ on the same point. This event is analyzed analogously to case $(\$^+, \$^+)$ above.

$(\$^+, \tilde{\mathsf{P}}_3^-)$ Calls $\xi^e = (\Delta_1^e, \Delta_2^e, \alpha^e)$ to KCEnc and $\xi^d = (\Delta_1^d, \Delta_2^d, \alpha^d)$ to KCDec result in (a) querying $\tilde{\mathsf{P}}_3^-$ on an input matching an output of $\$^+$; or in (b) querying $\tilde{\mathsf{P}}_3^-$ on an input whose output matches an input of $\$^+$. Let C be the reply to query ξ^e. Then the value chosen by $\$^+$ is $C \oplus K^* \oplus \Delta_1^e$. Thus case (a) would happen if

$$C \oplus K^* \oplus \Delta_1^e = \alpha^d \cdot K^* \oplus \Delta_2^d \oplus K^* \oplus \Delta_1^d \,.$$

If $\alpha^d = 1$, then

$$C \oplus K^* \oplus \Delta_1^e = \Delta_2^d \oplus \Delta_1^d \,,$$

which leads to guessing K^* and happens with probability at most $1/2^n$. On the other hand, if $\alpha^d = 0$, we would have that

$$C \oplus \Delta_1^e = \Delta_2^d \oplus \Delta_1^d \,.$$

But for such queries either $\tilde{\mathsf{P}}_3^-$ is sampled to be consistent with $\$^+$ (or it is not called at all when $\mathsf{Det}_\oplus$ returns "Yes").

Let us now look at case (b), which would happen if

$$\mathsf{P}_2\big(\overbrace{\mathsf{P}_1(\underbrace{\Delta_1^e \oplus \Delta_2^e \oplus (1-\alpha^e)\cdot K^*}_{=:X}) \oplus \Delta_1^e \oplus K^*}^{=:Y}\big) \oplus \Delta_1^e \oplus K^* = \tilde{\mathsf{P}}_3^-\big(\Delta_1^d \oplus \Delta_2^d \oplus (1-\alpha^d)\cdot K^*\big) \,. \quad (5.5)$$

If $\Delta_1^d \oplus \Delta_2^d \oplus (1-\alpha^d)\cdot K^*$ was directly queried to P_3^-, an inconsistency of type $(\$^+, \mathsf{P}_3^\pm)$ arises, which we dealt with above. On the other hand, it could be that this value was never queried to P_3^-.

If the KCEnc query comes before KCDec and the inputs are distinct—colliding inputs was case (a) above—$\tilde{\mathsf{P}}_3^-$ chooses a random value, which collides with the input to $\$^+$ with probability at most $1/(2^n - q)$.

Suppose now that the KCEnc query comes after KCDec. Suppose the input to KCEnc is $\xi^e = (\Delta_1^e, \Delta_2^e, \alpha^e)$. The value computed as input to $\$^+$ is independent of the output of $\tilde{\mathsf{P}}_3^-$ unless P_1^+ is kept consistent with $\$^-$ (since otherwise P_1 and P_2 are independent of $\tilde{\mathsf{P}}_3^-$). If independent, collisions happen with probability at most $1/(2^n - q)$. Otherwise, dependency implies that $M \oplus \Delta_1^d = \Delta_1^e \oplus \Delta_2^e$ where M is the output of KCDec. This

query will be marked as illegal unless $\Delta_1^d \neq \Delta_1^e$. In this case the adversary has found a collision of the form

$$\mathsf{P}_2(Y' \oplus K^* \oplus \Delta_1^e) \oplus K^* \oplus \Delta_1^e = U \ ,$$

where $Y' = \mathsf{P}_2^-(U \oplus K^* \oplus \Delta_1^d) \oplus K^* \oplus \Delta_1^d$ is the value input to $\$^-$ and output by P_1^+ due to the consistency check in KCEnc. Rearranging and applying P_2^- to the above equality yields $Y' \oplus K^* \oplus \Delta_1^e = \mathsf{P}_2^-(U \oplus K^* \oplus \Delta_1^e)$. Substituting the expression for Y' we get that $\mathsf{P}_2^-(U \oplus K^* \oplus \Delta_1^d) \oplus \Delta_1^d \oplus \Delta_1^e = \mathsf{P}_2^-(U \oplus K^* \oplus \Delta_1^e)$. Since P_2^- is a permutation, this would happen iff $\Delta_1^d = \Delta_1^e$, which is an illegal query.

$(\$^-, \tilde{\mathsf{P}}_1^+)$ A call ξ^d to KCDec and a call ξ^e to KCEnc result in querying $\$^-$ and $\tilde{\mathsf{P}}_1^+$ on the same point. This event is analyzed analogously to case $(\$^+, \tilde{\mathsf{P}}_3^-)$ above.

$(\tilde{\mathsf{P}}_3^-, \mathsf{P}_3^\pm)$ A call ξ^d to KCDec and a direct query to $\mathsf{P}_3^\pm$ are such that either the inputs or the outputs collide. If the KCDec query comes *after* the direct query, since oracle $\tilde{\mathsf{P}}_3^-$ is sampled to be consistent with P_3, no inconsistency arises. If P_3 comes after KCDec then an inconsistency arises only with an entry on $\tilde{\mathsf{P}}_3^-$ that is also on $\$^+$. But in this case an inconsistency of the form $(\$^+, \mathsf{P}_3^\pm)$ has been found.

$(\tilde{\mathsf{P}}_1^+, \mathsf{P}_1^\pm)$ A call ξ^e to KCEnc and a direct query to $\mathsf{P}_1^\pm$ collide. This case is treated analogously to $(\tilde{\mathsf{P}}_3^-, \mathsf{P}_3^\pm)$.

KCEnc inconsistently rejects If $\mathsf{Det}_\oplus$ does not output an incorrect answer, $\perp$ is the correct answer. On the other hand, as shown at the end of previous section, $\mathsf{Det}_\oplus$ with $i = 1$ outputs an incorrect answer with probability at most $1/2^n$ per inputs checked.

KCDec inconsistently rejects Similarly to the above, this event occurs with probability at most $1/2^n$ per inputs checked.

The theorem follows by putting the above bounds together. □

Remark. A potential strengthening of the above theorem would be to consider a wider class of KCA attacks for which the ideal cipher is known to be secure. One possibility would be the class of key-unpredictable, claw-free, and xkcd-secure CDF sets as shown in Theorem 5.1. Although feasibility of this level of security claim for iterated Even–Mansour ciphers would follow from known indifferentiability [LS13; DSST17], this would require a larger number of rounds and also comes at the cost of lower levels of security. It remains an interesting open question to find the minimal number of rounds needed in the Even–Mansour ciphers that yields (Ξ^e, Ξ^d)-KC-CCA security with respect to *any* pair (Ξ^e, Ξ^d) for which the ideal cipher is also (Ξ^e, Ξ^d)-KC-CCA secure.

6

KCA-SECURE SYMMETRIC ENCRYPTION

We have seen that it is possible to achieve KCA-secure blockciphers. Given this, we now turn our attention to higher-level primitives and investigate the possibility of constructing KCA-secure primitives which rely on blockciphers. A natural first step is to explore symmetric encryption. Specifically, we consider the notion of authenticated encryption with associated data (AEAD).

6.1 KC-AE SECURITY

Let SE be an AEAD scheme as defined in chapter 2. Let Ξ^e and Ξ^d be CDF sets with signatures $\xi^e : \mathcal{K} \longrightarrow \mathcal{K} \times \mathcal{M} \times \mathcal{N} \times \mathcal{H}$ and $\xi^d : \mathcal{K} \longrightarrow \mathcal{K} \times \mathcal{N} \times \mathcal{H}$ respectively. As is in the blockcipher case, Ξ^e is used to generate correlated keys *and* key-correlated messages. In addition to this, for authenticated encryption, we allow *all* inputs to the encryption algorithm to be key-correlated. Unlike the blockcipher case however, we restrict the use of Ξ^d to generate correlated keys, nonces, and headers as it seems unnatural to derive a key-correlated ciphertext for decryption. If $\xi^e \in \Xi^e$ and $\xi^d \in \Xi^d$ we note that the syntactical difference to SE when considering KC-AE security is that the encryption oracle in the scheme only takes ξ^e as input, while the decryption oracle takes (ξ^d, C) as described in the KC-AE$^{\Xi^e,\Xi^d,\mathcal{A}}_{\text{SE}}$ game on the right in Figure 6.1. The KC-AE advantage of an adversary $\mathcal{A}$ against AEAD is defined by

We can extend to key-correlated ciphertexts, but it will come at the cost of requiring stronger assumptions.

$$\mathbf{Adv}^{\text{kc-ae}}_{\text{SE}}(\Xi^e, \Xi^d, \mathcal{A}) := 2 \cdot \Pr\left[\text{KC-AE}^{\Xi^e,\Xi^d,\mathcal{A}}_{\text{SE}}\right] - 1 \,.$$

We require that nonces are not repeated in KCEnc queries.

We can further extend to misuse-resilient authenticated encryption, if we extend to the stronger assumptions as with key-correlated ciphertexts.

Security against key-correlated attacks is a strengthening of the standard notion of AE-security for a symmetric encryption scheme. As the syntax shows, the input to SE consists of K, M, N, and H. A natural question may arise as to which inputs we allow to be key-correlated. Although we would like to provide guarantees against *any* correlated inputs, there are two immediate issues with header security. It can (sometimes) be the case that headers are not private, and if they are heavily key-correlated, an adversary could trivially retrieve the key. For this reason, we require that header data is private. But a more subtle point, as shown in [BK11], is that even with private headers, it is possible for an adversary to perform a full key recovery attack by taking advantage of the pattern of errors returned while list-checking during the decryption process which may be key-dependent. This attack does not apply in our setting, as we

The game defining the KC-AE-security of SE with key-correlated inputs. Once again, the adversary is required not to repeat nonces in its KCENC queries.

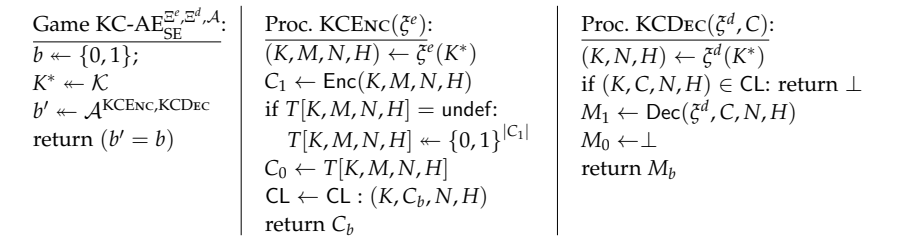


Figure 6.1 – KC-AE-security game for a symmetric encryption scheme SE.

return only one error symbol ($\perp$) for both illegal queries and a failed decryption process, and as such, an adversary cannot use this to gain advantage. Note that even if we define a scheme with different error symbols to denote different types of decryption failures, we can still obtain header security, but at the cost of requiring key-claw-freeness.

6.2 THE HASH-WITH-NONCE TRANSFORM

We introduce the Hash-with-Nonce (HwN) transform with the goal of achieving KCA-security for symmetric encryption. Working in the random-oracle model, HwN converts a conventional AE-secure AEAD scheme SE to a new one, $\overline{\mathsf{SE}} = \mathrm{HwN}[\mathsf{SE}, \mathsf{Gen}] = (\mathsf{Gen}, \overline{\mathsf{Enc}}, \overline{\mathsf{Dec}})$, which we will prove to be KC-AE secure. Gen is its key-generation algorithm, its key length is k and nonce length is n. Its encryption and decryption algorithms are defined as follows, where $\mathsf{H} : \mathcal{K} \times \mathcal{N} \longrightarrow \mathcal{K}$ is a random oracle.

$\overline{\mathsf{Enc}}(K, M, N, H)$:	$\overline{\mathsf{Dec}}(K, C, N, H)$:
$C \leftarrow \mathsf{Enc}(\mathsf{H}(K\|N), M, N, H)$	$M \leftarrow \mathsf{Dec}(\mathsf{H}(K\|N), C, N, H)$
return C	return M

To show that $\overline{\mathsf{SE}}$ is (Ξ^e, Ξ^d)-KC-AE secure it will be convenient to reduce to a multi-user variant of the AE game. Multi-user authenticated encryption (MUAE) allows an adversary to choose the number of keys to target, analogous to the number of users in the scheme. In addition to taking a message, nonce, and header, the encryption oracle takes an index i (indicating user number), and returns encryptions under a key K_i, or a random string of the same length. Decryption always returns the error symbol, except in the case the real algorithms are being used where the message M is returned by decrypting C under K_i. The adversary cannot ask user i to decrypt a ciphertext that it previously obtained from this user. In a scheme with n users, it follows via a simple hybrid argument (see Section 6.5) that any AE-secure AEAD scheme is also MUAE secure.

The convenience in using MUAE lies in the fact that we can leverage the programmability of random oracles to implicitly map the hash values $\mathrm{H}(\xi^e_i|_1(K^*)|N_i)$ of the i-th query ξ^e_i to KCENC, to a new randomly and independently chosen key in the MUAE game. This also

decouples the key used in encryption (i.e., $\mathrm{H}(\xi_i^e|_1(K^*)|N_i)$) from the key-correlated messages $\xi_i^e|_2(K^*)$, key-correlated nonces $\xi_i^e|_3(K^*)$, and key-correlated headers $\xi_i^e|_4(K^*)$. Overall, our reduction proceeds by choosing a K^*, computing the key-correlated message, nonce, and header, faithfully using this key, and implicitly setting $\mathrm{H}(\xi_i^e|_1(K^*)|N)$ to keys K_i in the MUAE game. For this analysis, we need that the keys K_i remain hidden from the adversary's point of view, which we argue is the case as long the queried functions are key-unpredictable. We now state and prove the security guarantees of $\overline{\mathrm{SE}}$

6.3 KCA SECURITY OF HWN

Theorem 6.1 (KCA security of HwN)**.** *Let* SE *be an* AE*-secure* AEAD *scheme. Let* Ξ^e *and* Ξ^d *be two CDF sets. Then for any* (Ξ^e, Ξ^d)-KC-AE *adversary* $\mathcal{A}$ *against the scheme* $\overline{\mathrm{SE}}$ *obtained by applying the* HwN *transform to* SE *that makes* q *random oracle queries,* q_e *encryption queries and* q_d *decryption queries, there are adversaries* $\mathcal{B}$, $\mathcal{C}_1$ *and* $\mathcal{C}_2$ *such that*

$$\mathbf{Adv}^{\text{kc-ae}}_{\overline{\mathrm{SE}}}(\Xi^e, \Xi^d, \mathcal{A}) \leq 2 \cdot \mathbf{Adv}^{\text{mu-ae}}_{\mathrm{SE}}(\mathcal{B}) + 2 \cdot \mathbf{Adv}^{\text{m-kup}}_{\Xi^e}(\mathcal{C}_1) + 2 \cdot \mathbf{Adv}^{\text{m-kup}}_{\Xi^d}(\mathcal{C}_2)$$

where adversary $\mathcal{B}$ *makes at most* $q_e + q_d$ *queries to* INIT*, at most* q_e *queries to encryption, and at most* q_d *queries to decryption, adversary* $\mathcal{C}_1$ *outputs two lists of sizes at most* q_e *and* q*, and adversary* $\mathcal{C}_2$ *outputs two lists of sizes at most* q_d *and* q*.*

Proof. The proof proceeds via a sequence of games as shown in Figure 6.2 and Figure 6.3. The three games in Figure 6.2 run according to the KC-AE game with the HwN transform applied, and with $b = 1$. When $b = 1$, all oracle queries to KCENC return encryptions of key-derived inputs under related keys. Queries to KCDEC return the plaintext, or the error symbol $\perp$ if the ciphertext is deemed illegitimate. The three games in Figure 6.3 run according to the KC-AE game with the HwN transform applied, and with $b = 0$. In this scenario, all oracle queries to KCENC return random ciphertexts, and KCDEC always returns the error symbol $\perp$. All games, adversaries, and transitions are outlined below.

Game_0 describes the KC-AE game with respect to the transformed scheme and $b = 1$.

Game_1 is identical to Game_0 with one conceptual change: it proceeds by separating the book-keeping list for H into cases when H is accessed within KCENC and KCDEC and when directly accessed by the adversary. More precisely, in Game_1, records of adversary $\mathcal{A}$'s queries to the random oracle H will be recorded in a list L, whereas the oracle queries from KCENC and KCDEC will be recorded in a list L' by oracle denoted H′. A bad flag, Bad, is set if H′ is queried on a key/nonce pair that has already been queried by $\mathcal{A}$ to H. In this game, H and H′ jointly implement H

Game_0 shows the KC-AE game with respect to $\overline{\mathsf{SE}}$ and challenge bit 1. Game_1 separates the lists for H and H′. There are no functional changes. Game_2 then fully decouples H and H′.

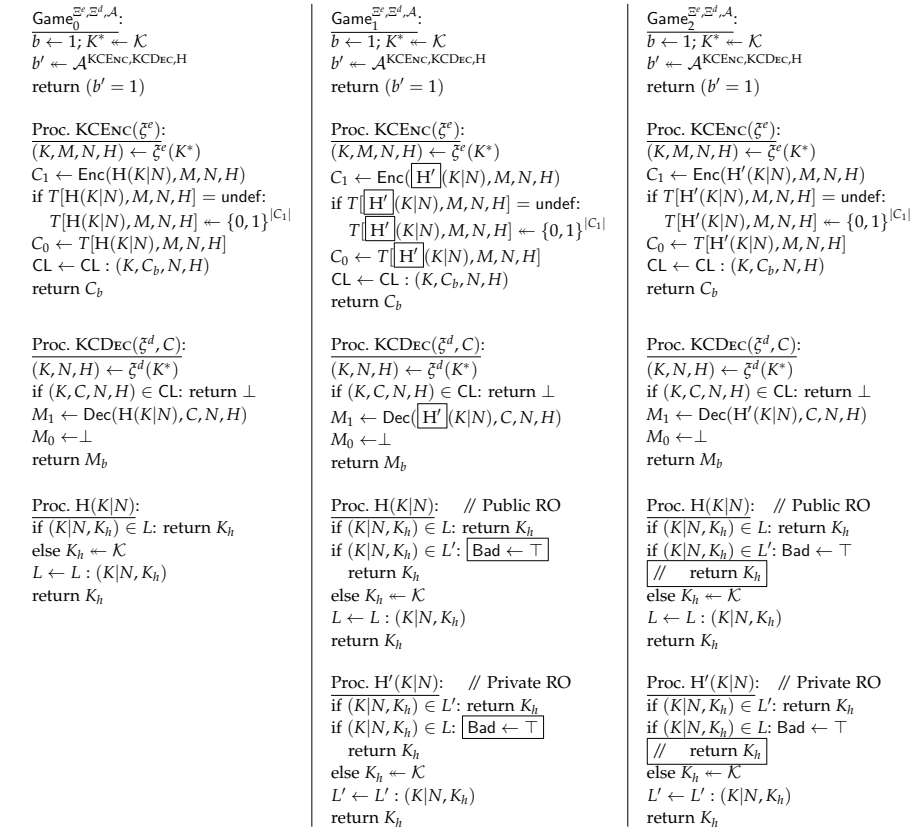


Figure 6.2 – KC-AE to SE reduction with $b = 1$.

in Game_0 and event Bad prepares us to fully decouple H and H′ later:

$$\Pr[\mathsf{Game}_0^{\mathcal{A}}] = \Pr[\mathsf{Game}_1^{\mathcal{A}}] .$$

Game_2 is identical to Game_1 up to the point that Bad is set. If this event occurs within H′, records are no longer returned. By the fundamental lemma of game playing [BR06], we bound the difference between Game_2 and Game_1 as

$$\Pr[\mathsf{Game}_1^{\mathcal{A}}] - \Pr[\mathsf{Game}_2^{\mathcal{A}}] \leq \Pr[\mathsf{Game}_2^{\mathcal{A}} \text{ sets Bad}] .$$

Event Bad corresponds to a non-empty intersection between the two lists L and L'. Below, we will bound $\Pr[\mathsf{Bad}]$ in Game_1 by showing that (1) this probability is close to the probability of setting Bad in Game_2 down to the MUAE security of SE; and (2) the probability of setting Bad in Game_2 is small down to the key unpredictability of Ξ^e and Ξ^d.

$\underline{\mathsf{Game}_3^{\Xi^e,\Xi^d,\mathcal{A}}}$:
$\boxed{b \leftarrow 0}$; $K^* \twoheadleftarrow \mathcal{K}$
$b' \twoheadleftarrow \mathcal{A}^{\text{KCEnc,KCDec,H}}$
return $(b' = 1)$

$\underline{\text{Proc. KCEnc}(\xi^e)}$:
$(K, M, N, H) \leftarrow \xi^e(K^*)$
$C_1 \leftarrow \mathsf{Enc}(\text{H}'(K|N), M, N, H)$
if $T[\text{H}'(K|N), M, N, H] = \mathsf{undef}$:
  $T[\text{H}'(K|N), M, N, H] \twoheadleftarrow \{0,1\}^{|C_1|}$
$C_0 \leftarrow T[\text{H}'(K|N), M, N, H]$
$\mathsf{CL} \leftarrow \mathsf{CL} : (K, C_b, N, H)$
return C_b

$\underline{\text{Proc. KCDec}(\xi^d, C)}$:
$(K, N, H) \leftarrow \xi^d(K^*)$
if $(K, C, N, H) \in \mathsf{CL}$: return $\perp$
$M_1 \leftarrow \mathsf{Dec}(\text{H}'(K|N), C, N, H)$
$M_0 \leftarrow \perp$
return M_b

$\underline{\text{Proc. H}(K|N)}$: // Public RO
if $(K|N, K_h) \in L$: return K_h
if $(K|N, K_h) \in L'$: $\mathsf{Bad} \leftarrow \top$
// return K_h
else $K_h \twoheadleftarrow \mathcal{K}$
$L \leftarrow L : (K|N, K_h)$
return K_h

$\underline{\text{Proc. H}'(K|N)}$: // Private RO
if $(K|N, K_h) \in L'$: return K_h
if $(K|N, K_h) \in L$: $\mathsf{Bad} \leftarrow \top$
// return K_h
else $K_h \twoheadleftarrow \mathcal{K}$
$L' \leftarrow L' : (K|N, K_h)$
return K_h

$\underline{\mathsf{Game}_4^{\Xi^e,\Xi^d,\mathcal{A}}}$:
$b \leftarrow 0$; $K^* \twoheadleftarrow \mathcal{K}$
$b' \twoheadleftarrow \mathcal{A}^{\text{KCEnc,KCDec,H}}$
return $(b' = 1)$

$\underline{\text{Proc. KCEnc}(\xi^e)}$:
$(K, M, N, H) \leftarrow \xi^e(K^*)$
$C_1 \leftarrow \mathsf{Enc}(\text{H}'(K|N), M, N, H)$
if $T[\text{H}'(K|N), M, N, H] = \mathsf{undef}$:
  $T[\text{H}'(K|N), M, N, H] \twoheadleftarrow \{0,1\}^{|C_1|}$
$C_0 \leftarrow T[\text{H}'(K|N), M, N, H]$
$\mathsf{CL} \leftarrow \mathsf{CL} : (K, C_b, N, H)$
return C_b

$\underline{\text{Proc. KCDec}(\xi^d, C)}$:
$(K, N, H) \leftarrow \xi^d(K^*)$
if $(K, C, N, H) \in \mathsf{CL}$: return $\perp$
$M_1 \leftarrow \mathsf{Dec}(\text{H}'(K|N), C, N, H)$
$M_0 \leftarrow \perp$
return M_b

$\underline{\text{Proc. H}(K|N)}$: // Public RO
if $(K|N, K_h) \in L$: return K_h
if $(K|N, K_h) \in L'$: $\mathsf{Bad} \leftarrow \top$
$\boxed{\text{return } K_h}$
else $K_h \twoheadleftarrow \mathcal{K}$
$L \leftarrow L : (K|N, K_h)$
return K_h

$\underline{\text{Proc. H}'(K|N)}$: // Private RO
if $(K|N, K_h) \in L'$: return K_h
if $(K|N, K_h) \in L$: $\mathsf{Bad} \leftarrow \top$
$\boxed{\text{return } K_h}$
else $K_h \twoheadleftarrow \mathcal{K}$
$L' \leftarrow L' : (K|N, K_h)$
return K_h

$\underline{\mathsf{Game}_5^{\Xi^e,\Xi^d,\mathcal{A}}}$:
$b \leftarrow 0$; $K^* \twoheadleftarrow \mathcal{K}$
$b' \twoheadleftarrow \mathcal{A}^{\text{KCEnc,KCDec,H}}$
return $(b' = 1)$

$\underline{\text{Proc. KCEnc}(\xi^e)}$:
$(K, M, N, H) \leftarrow \xi^e(K^*)$
$C_1 \leftarrow \mathsf{Enc}(\boxed{\text{H}}(K|N), M, N, H)$
if $T[\boxed{\text{H}}(K|N), M, N, H] = \mathsf{undef}$:
  $T[\boxed{\text{H}}(K|N), M, N, H] \twoheadleftarrow \{0,1\}^{|C_1|}$
$C_0 \leftarrow T[\boxed{\text{H}}(K|N), M, N, H]$
$\mathsf{CL} \leftarrow \mathsf{CL} : (K, C_b, N, H)$
return C_b

$\underline{\text{Proc. KCDec}(\xi^d, C)}$:
$(K, N, H) \leftarrow \xi^d(K^*)$
if $(K, C, N, H) \in \mathsf{CL}$: return $\perp$
$M_1 \leftarrow \mathsf{Dec}(\boxed{\text{H}}(K|N), C, N, H)$
$M_0 \leftarrow \perp$
return M_b

$\underline{\text{Proc. H}(K|N)}$:
if $(K|N, K_h) \in L$: return K_h
else $K_h \twoheadleftarrow \mathcal{K}$
$L \leftarrow L : (K|N, K_h)$
return K_h

Game_3 changes the challenge bit to 0. Game_4 merges H and H′. Game_5 is functionally identical to Game_4 with merged H and H′. Game_5 is the KC-AE game with respect to $\overline{\text{SE}}$ and challenge bit 0.

Figure 6.3 – KC-AE to SE reduction with $b = 0$.

Game_3 switches the challenge bit to $b = 0$. We show below that any adversarial advantage in distinguishing Game_2 and Game_3 can be converted, via an adversary $\mathcal{B}_1$, against the MUAE game:

$$\Pr[\mathsf{Game}_2^{\mathcal{A}}] - \Pr[\mathsf{Game}_3^{\mathcal{A}}] = \mathbf{Adv}_{\text{SE}}^{\text{mu-ae}}(\mathcal{B}_1) \ .$$

Game_4 reverses the change in Game_2 and merges random oracles H and H′. We have that

$$\Pr[\mathsf{Game}_3^{\mathcal{A}}] - \Pr[\mathsf{Game}_4^{\mathcal{A}}] \leq \Pr[\mathsf{Game}_3^{\mathcal{A}} \text{ sets } \mathsf{Bad}] \ ,$$

where we choose Game_3 to analyze the probability of Bad. Below we construct adversaries to show that this probability is bounded above by the key-unpredictability advantages against Ξ^e and Ξ^d.

Game_5 is identical to Game_4 except for a reversal of the conceptual change seen between Game_0 and Game_1. The random oracles are completely re-coupled, and all queries by $\mathcal{A}$, KCEnc and KCDec are written to a single list L. There are no procedural changes in this game:

$$\Pr[\mathsf{Game}_4^{\mathcal{A}}] = \Pr[\mathsf{Game}_5^{\mathcal{A}}] \ .$$

6.4 THE REDUCTIONS

We now give the details of the bounds for the three transitional changes above.

ADVERSARY $\mathcal{B}_1$ plays the MUAE game and is built based on an adversary $\mathcal{A}$ that attempts to distinguish Game_2 and Game_3 as follows. Adversary $\mathcal{B}_1$ initializes q keys *and chooses* a K^*. It then simulates the encryption and decryption oracles by explicitly computing the actual key, and also message when answering encryption, used using K^*. In doing so, $\mathcal{B}_1$ *implicitly programs* $\mathrm{H}(\xi_i^e(K^*), N_i)$ to K_i, one of the q keys initialized by $\mathcal{B}_1$'s challenger. Since nonces are not reused, and each K_i is chosen randomly, this will constitute a perfect simulation of the random oracle H′. When $b = 0$, this is a perfect simulation of Game_4 environment for $\mathcal{A}$ by $\mathcal{B}_1$. However when $b = 1$, although simulation encryption is good, care must be taken in decryption as decryption of (K, C, N, H) is not allowed in the KC-AE game if (K, C, N, H) was added to list CL. Adversary $\mathcal{B}_1$ thus keeps track of these values and answers with $\perp$ when such a query arises. Note that $\mathcal{B}_1$ once again uses its knowledge K^* and further that this modification does not affect the simulation when $b = 0$. Adversary $\mathcal{B}_1$ continues in this way and returns the bit that $\mathcal{A}$ outputs. Hence,

$$\begin{aligned}\mathbf{Adv}^{\text{mu-ae}}_{\mathrm{SE}}(\mathcal{B}_1) &= \Pr[b' = 1 | b = 1] - \Pr[b' = 1 | b = 0] \\ &= \Pr[\mathsf{Game}_2^{\mathcal{A}}] - \Pr[\mathsf{Game}_3^{\mathcal{A}}] \ .\end{aligned}$$

ADVERSARIES $\mathcal{C}_i$ FOR $i = 1, 2$ are used to show that the probability of Bad is small by the key-unpredictability of the queried functions. Adversaries $\mathcal{C}_i$ for $i = 1, 2$ run $\mathcal{A}$ in Game_3 by *simply simulating its encryption and decryption oracles with* \$ *and* $\perp$ *respectively*. They also lazily sample the random oracles H for $\mathcal{A}$. Adversary $\mathcal{C}_1$ keeps track of all queried functions to KCEnc and adversary $\mathcal{C}_2$ keeps track of all queried functions to KCDec. Both adversaries also keep track of all queries to H. When $\mathcal{A}$ terminates, $\mathcal{C}_i$ simply outputs the list of all queried functions that it keeps track of and H queries its two lists of guesses in the multi-shot key-unpredictability game. (See Figure 6.4 (middle) for the details.) Whenever flag Bad is triggered, either $\mathcal{C}_1$ or $\mathcal{C}_2$ wins the key-unpredictability game against Ξ^e or Ξ^d respectively:

$$\Pr[\mathsf{Game}_3^{\mathcal{A}} \text{ sets Bad}] \leq \mathbf{Adv}^{\text{m-kup}}_{\Xi^e}(\mathcal{C}_1) + \mathbf{Adv}^{\text{m-kup}}_{\Xi^d}(\mathcal{C}_2) \ .$$

ADVERSARY $\mathcal{B}_2$ is used to bound the difference between the probabilities of setting Bad in games Game_3 and Game_4. This is done via another reduction to the MUAE game which attempts to decide the challenge bit by observing if Bad is set. Algorithm $\mathcal{B}_2$

runs $\mathcal{A}$ similarly to algorithm $\mathcal{B}_1$. It however keeps track of the queried functions to KCENC and KCDEC as well as H. It uses its knowledge of K^* to verify if Bad has occurred. When the MUAE challenge bit of $\mathcal{B}_2$ is 1, algorithm $\mathcal{B}_2$ runs $\mathcal{A}$ in an environment identical to that of Game_2; hence the probability of $\mathcal{B}_2$ outputting 0 is exactly that of Bad in Game_2. Similarly, the probability of $\mathcal{B}_2$ outputting 1 is exactly that of Bad occurring in Game_3. Hence

$$\mathbf{Adv}^{\text{mu-ae}}_{\mathrm{SE}}(\mathcal{B}_2) = \Pr[\mathsf{Game}_2^{\mathcal{A}} \text{ sets Bad}] - \Pr[\mathsf{Game}_3^{\mathcal{A}} \text{ sets Bad}] \ .$$

The theorem follows by putting the bounds established above together, and noting that

$$\mathbf{Adv}^{\text{kc-ae}}_{\overline{\mathrm{SE}}}(\Xi^e, \Xi^d, \mathcal{A}) = \Pr[\mathsf{Game}_0^{\mathcal{A}}] - \Pr[\mathsf{Game}_5^{\mathcal{A}}] \ .$$

In the theorem statement, we choose $\mathcal{B}$ to be $\mathcal{B}_1$ or $\mathcal{B}_2$ having the bigger advantage. □

Adversary $\mathcal{B}_1^{\text{INIT,ENC,DEC}}$:
$i \leftarrow 0$; $K^* \twoheadleftarrow \mathcal{K}$
Call $\mathcal{B}_1$.INIT $q_e + q_d$ times
$b' \twoheadleftarrow \mathcal{A}^{\text{KCENC,KCDEC,H}}$
return b'

KCENC(ξ^e):
$(K, M, N, H) \leftarrow \xi^e(K^*)$
if $T[K, N] = \text{undef}$:
  $i \leftarrow i + 1$; $T[K, N] \leftarrow i$
$C \leftarrow \mathcal{B}_1$.ENC$(T[K, N], M, N, H)$
$\mathsf{CL} \leftarrow \mathsf{CL} : (K, C, N, H)$
return C

KCDEC(ξ^d, C):
$(K, N, H) \leftarrow \xi^d(K^*)$
if $T[K, N] = \text{undef}$:
  $i \leftarrow i + 1$; $T[K, N] \leftarrow i$
if $(K, C, N, H) \in \mathsf{CL}$: return $\perp$
$M \leftarrow \mathcal{B}_1$.DEC$(T[K, N], C, N, H)$
return M

H($K|N$):
if $(K|N, K_h) \in L$: return K_h
else $K_h \twoheadleftarrow \mathcal{K}$
$L \leftarrow L : (K|N, K_h)$
return K_h

Adversary $\mathcal{C}_i$:
$K^* \twoheadleftarrow \mathcal{K}$
$b' \twoheadleftarrow \mathcal{A}^{\text{KCENC,KCDEC,H}}$
return $(\mathsf{L}_i, \mathsf{L}')$

KCENC(ξ^e):
$\mathsf{L}_1 \leftarrow \mathsf{L}_1 : \xi^e$
$C \twoheadleftarrow \{0,1\}^{|M|+\tau}$
return C

KCDEC(ξ^d, C):
$\mathsf{L}_2 \leftarrow \mathsf{L}_2 : \xi^d$
return $\perp$

H($K|N$):
if $(K|N, K_h) \in L$:
  return K_h
else $K_h \twoheadleftarrow \mathcal{K}$
$L \leftarrow L : (K|N, K_h)$
$\mathsf{L}' \leftarrow \mathsf{L}' : K$
return K_h

Adversary $\mathcal{B}_2^{\text{INIT,ENC,DEC}}$:
$i \leftarrow 0$; $K^* \twoheadleftarrow \mathcal{K}$
Call $\mathcal{B}_2$.INIT q times
$b' \twoheadleftarrow \mathcal{A}^{\text{KCENC,KCDEC,H}}$
for $(\xi) \in \mathsf{L}_1 \wedge (K, N) \in \mathsf{L}'$
  if $\xi|_1(K^*) = K \wedge \xi|_3(K^*) = N$:
    return 1
return 0

KCENC(ξ^e):
$\mathsf{L}_1 \leftarrow \mathsf{L}_1 : (\xi^e)$
$(K, M, N, H) \leftarrow \xi^e(K^*)$
if $T[K, N] = \text{undef}$:
  $i \leftarrow i + 1$; $T[K, N] \leftarrow i$
$C \leftarrow \mathcal{B}_2$.ENC$(T[K, N], M, N, H)$
$\mathsf{CL} \leftarrow \mathsf{CL} : (K, C, N, H)$
return C

KCDEC(ξ^d, C):
$\mathsf{L}_1 \leftarrow \mathsf{L}_1 : (\xi^d)$
$(K, N, H) \leftarrow \xi^d(K^*)$
if $T[K, N] = \text{undef}$:
  $i \leftarrow i + 1$; $T[K, N] \leftarrow i$
$M \leftarrow \mathcal{B}_2$.DEC$(T[K, N], C, N, H)$
if $(K, C, N, H) \in \mathsf{CL}$: return $\perp$
return M

H($K|N$):
if $(K|N, K_h) \in L$: return K_h
else $K_h \twoheadleftarrow \mathcal{K}$
$L \leftarrow L : (K|N, K_h)$
$\mathsf{L}' \leftarrow \mathsf{L}' : (K, N)$
return K_h

On the left is the description of adversary $\mathcal{B}_1$ simulating the challenger of $\mathcal{A}$ in the KC-AE game whilst interacting in the MUAE environment. In the middle is the description of adversaries $\mathcal{C}_i$ simulating the oracle requests by $\mathcal{A}$ in the KC-AE game when $b = 0$ and whilst running in the key unpredictability environment. On the right is the code for an adversary $\mathcal{B}_2$ simulating $\mathcal{A}$ in the KC-AE game whilst interacting in the MUAE environment.

Figure 6.4 – Code of the adversaries in the KC-AE to MUAE reduction.

REMARK. The above theorem can be extended to a misuse-resilient and key-correlated setting, whereby nonces may be repeated at encryption for a construction similar to HwN that also hashes the header

Game $\text{MUAE}^{\mathcal{A}}_{\text{SE}}$:
$b \twoheadleftarrow \{0,1\}; v \leftarrow 0$
$b' \twoheadleftarrow \mathcal{A}^{\text{Init},\text{Enc},\text{Dec}}$
return $(b' = b)$

Proc. Init():
$v \leftarrow v + 1; K_v \twoheadleftarrow \mathcal{K}$

Proc. Enc(i, M, N, H):
if $\neg(1 \le i \le v)$: return $\perp$
$C_1 \leftarrow \text{Enc}(K_i, M, N, H)$
$C_0 \twoheadleftarrow \{0,1\}^{|C_1|}$
$\text{CL} \leftarrow \text{CL} : (i, C_b, N, H)$
return C_b

Proc. Dec(i, C, N, H):
if $\neg(1 \le i \le v)$: return $\perp$
if $(i, C, N, H) \in \text{CL}$: return $\perp$
$M_1 \leftarrow \text{Dec}(K_i, C, N, H)$
$M_0 \leftarrow \perp$
return M_b

Figure 6.5 – Game $\text{MUAE}^{\mathcal{A}}_{\text{AE}}$ defining the multi-user AE-security of a authenticated encryption scheme SE.

information. A similar proof strategy can be used to analyse its KC-MRAE security at the expense of introducing three extra conditions: (1) the base scheme is MRAE secure; (2) the sets Ξ^e and Ξ^d are claw-free; and (3) (Ξ^e, Ξ^d) is xkcd so that illegal decryption queries can be detected.

6.5 MULTI-USER TO SINGLE-USER REDUCTION FOR AE

This may not seem very important, and you may wonder why it is included, but this is the first reduction I ever wrote, and for that reason, it's super important to me :)

For convenience, the KC-AE to AE proof was shown in the multi-user setting. Here we show how to further reduce from multi-user to single user AE.

MULTI-USER SECURITY. We define the MUAE-security of an authenticated encryption scheme $\text{SE} = (\text{Gen}, \text{Enc}, \text{Dec})$ by considering the game described in Figure 6.5. The MUAE advantage of an adversary $\mathcal{A}$ against an MUAE game is defined by

$$\mathbf{Adv}^{\text{mu-ae}}_{\text{SE}}(\mathcal{A}) := 2 \cdot \Pr\left[\text{MUAE}^{\mathcal{A}}_{\text{SE}}\right] - 1 \,.$$

Note that nonces may not be repeated for each user in Enc.

Theorem 6.2. *Let* SE *be an authenticated encryption scheme. Then for any adversary* $\mathcal{A}$ *attacking the* MUAE*-security of* SE *while making at most* n *queries to* Init*, there exist* AE *adversaries* $\mathcal{B}_1, \ldots, \mathcal{B}_n$ *such that*

$$\mathbf{Adv}^{\text{mu-ae}}_{\text{SE}}(\mathcal{A}) \le \sum_{i=1}^{n} \mathbf{Adv}^{\text{ae}}_{\text{SE}}(\mathcal{B}_i) \,.$$

Proof. The proof proceeds via a standard hybrid argument. We start by defining a sequence of games Game_j (for $0 \le j < n$) as shown in Figure 6.6. Game_j runs similarly to the MUAE game except that if i (the index of the key) is less than or equal to j, then ciphertext C is randomly generated. For $i > j$, the encryption algorithm is used under key K_i. Thus Game_0 is identical to the MUAE game with $b = 1$ and Game_n is identical to the MUAE game with $b = 0$.

$\mathsf{Game}_j^{\mathcal{A}}$:
$\overline{b \leftarrow 1; v \leftarrow 0}$
$b' \twoheadleftarrow \mathcal{A}^{\textsc{Init},\textsc{Enc},\textsc{Dec}}$
return $(b' = 1)$

Proc. $\textsc{Init}()$:
$\overline{v \leftarrow v + 1; K_v \twoheadleftarrow \mathcal{K}}$

Proc. $\textsc{Enc}(i, M, N, H)$:
$\overline{\text{if } \neg(1 \le i \le v) \text{: return } \perp}$
$C \leftarrow \mathsf{Enc}(K_i, M, N, H)$
if $i \le j$: $C \leftarrow \{0,1\}^{|C|}$
$\mathsf{CL} \leftarrow \mathsf{CL} : (i, C, N, H)$
return C

Proc. $\textsc{Dec}(i, C, N, H)$:
$\overline{\text{if } \neg(1 \le i \le v) \text{: return } \perp}$
if $(i, C, N, H) \in \mathsf{CL}$: return $\perp$
if $i \le j$: $M \leftarrow \perp$
$M \leftarrow \mathsf{Dec}(K_i, C, N, H)$
return M

Figure 6.6 – Game_j (with $0 \le j \le n$) showing MUAE game which returns random values when $i \le j$ and an encryption under K_i otherwise.

Let $\mathcal{A}$ be any adversary in the MUAE game. To bound the advantage of $\mathcal{A}$, note that

$$\begin{aligned}\mathbf{Adv}^{\text{mu-ae}}_{\mathrm{SE}}(\mathcal{A}) &:= \Pr[\mathsf{Game}_0^{\mathcal{A}}] - \Pr[\mathsf{Game}_n^{\mathcal{A}}] \\ &= \sum_{j=0}^{n-1} (\Pr[\mathsf{Game}_j^{\mathcal{A}}] - \Pr[\mathsf{Game}_{j+1}^{\mathcal{A}}]) \ .\end{aligned}$$

Thus, for $j = 0, \ldots, n-1$ it remains to bound

$$\Pr[\mathsf{Game}_j^{\mathcal{A}}] - \Pr[\mathsf{Game}_{j+1}^{\mathcal{A}}] \ .$$

To this end, we will rely on the underlying security of SE.

Claim 6.3. *For any j, and for any adversary $\mathcal{A}$, there exists a $\mathcal{B}_j$ such that*

$$\Pr[\mathsf{Game}_j^{\mathcal{A}}] - \Pr[\mathsf{Game}_{j+1}^{\mathcal{A}}] = \mathbf{Adv}^{\mathrm{ae}}_{\mathrm{SE}}(\mathcal{B}_j) \ .$$

The key observation is that in Game_j, for the first j keys queried, j uniform strings are returned as the ciphertext and for the remaining keys, $n - j$ encryptions are returned. In Game_{j+1}, $j + 1$ uniform strings and $n - j - 1$ encryptions are returned. Leveraging this difference of one real encryption at the $j + 1$ position, we can construct an AE adversary $\mathcal{B}_j$ to emulate the environment of an MUAE adversary $\mathcal{A}$ to break the single user AE security. Adversary $\mathcal{B}_j$ works as follows.

Adversary $\mathcal{B}_j^{\textsc{Enc},\textsc{Dec}}$:
$\overline{b' \twoheadleftarrow \mathcal{A}^{\textsc{Init},\textsc{Enc},\textsc{Dec}}}$
return b'

$\textsc{Init}()$:
$\overline{\text{if } i = j + 1 \text{: } K_i \leftarrow \perp}$
else $K_i \twoheadleftarrow \mathcal{K}$

$\textsc{Enc}(i, M, N, H)$:
$\overline{C \leftarrow \mathsf{Enc}(K_i, M, N, H)}$
if $i \le j$:
  $C \twoheadleftarrow \{0,1\}^{|C|}$
if $i = j + 1$:
  $C \leftarrow \mathcal{B}_j.\textsc{Enc}(M, N, H)$
return C

$\textsc{Dec}(i, C, N, H)$:
$\overline{\text{if } i \le j : M \leftarrow \perp}$
if $i = j + 1$:
  $M \leftarrow \mathcal{B}_j.\textsc{Dec}(C, N, H)$
else $M \leftarrow \mathsf{Dec}(K_i, C, N, H)$
return M

Figure 6.7 – Adversary $\mathcal{B}$ in the single-user AE game based on a multi-user AE adversary $\mathcal{A}$.

In the AE game that $\mathcal{B}_j$ is playing, if the challenge bit $b = 1$, adversary $\mathcal{A}$'s view of the environment is identical to Game_j. If $\mathcal{B}_j$'s challenge bit $b = 0$, $\mathcal{A}$'s view is identical to Game_{j+1}. When $\mathcal{A}$ outputs

a bit $b = 1$ if the envirnment appears to be that of Game_j, and $b = 0$ if the guess is that Game_{j+1} is observed. $\mathcal{B}_j$ outputs the same bit b and wins the AE game if $\mathcal{A}$ has distingushed correctly. Thus, if $b_{\mathcal{B}}$ is the bit output by $\mathcal{B}_j$ and $b_{\mathcal{A}}$ the bit output by $\mathcal{A}$ we conclude that

$$\begin{aligned}\mathbf{Adv}^{\mathrm{ae}}_{\mathrm{SE}}(\mathcal{B}_j) &= \Pr[b' = 1 | b_{\mathcal{B}} = 1] - \Pr[b' = 0 | b_{\mathcal{B}} = 0] \\ &= \Pr[\mathsf{Game}^{\mathcal{A}}_j] - \Pr[\mathsf{Game}^{\mathcal{A}}_{j+1}] \ .\end{aligned}$$

□

Part III

FAIL AGAIN

Abstract

In a recent paper [AJPS17d], Aggarwal, Joux, Prakash, and Santha (AJPS) describe a simple public-key cryptosystem mimicking NTRU over the integers. This algorithm relies on the properties of Mersenne primes instead of polynomial rings. The security of the AJPS cryptosystem relies on the conjectured hardness of the Mersenne Low Hamming Ratio Assumption, defined in [AJPS17d].

We show that AJPS' security estimates are too optimistic and describe an algorithm allowing to recover the secret key from the public key much faster than foreseen in [AJPS17d].

In particular, our algorithm is *experimentally practical* (within the reach of the computational capabilities of a large organization), at least for the parameter choice $\{n = 1279, h = 17\}$ conjectured in [AJPS17d] as corresponding to a 2^{120} security level. The algorithm is fully parallelizable.

This is joint work with Marc Beunardeau, Rémi Geraud, and David Naccache. The corresponding paper was presented at the 5^{th} International Conference on Cryptology and Information Security in Latin America, Latincrypt 2017, in La Habana, Cuba; and has been published as [BCGN17a].

Further to this, subsequent work by de Boer et al. [BDJW18], led to a revision of the effective hardness, and led Aggarwal et al. to amend the original cryptosystem substantially [AJPS18]. Ferradi and Naccache further suggested slightly improved variants [FN17a] along with several research directions.

We introduce a cryptosystem similar in spirit to the original Aggarwal–Joux–Prakash–Santha cryptosystem (AJPS-1) but which relies on a *different* hardness assumption. As a result, the lattice reduction attack (*à la* [BCGN17a]) does not seem to apply. The resulting construction is conceptually simpler than the "fixed" AJPS cryptosystem (AJPS-ECC), and than Ferradi and Naccache's "high-bandwidth" variant (AJPS-FN-BT). This is joint work with Marc Beunardeau, Rémi Geraud, and David Naccache.

7

ON THE HARDNESS OF THE MERSENNE LOW HAMMING RATIO ASSUMPTION

7.1 INTRODUCTION

A Mersenne prime is a prime of the form $2^n - 1$, where $n > 1$ is itself prime.

In a recent paper [AJPS17d], Aggarwal, Joux, Prakash, and Santha (AJPS) describe an ingenious public-key cryptosystem mimicking NTRU over the integers. This algorithm relies on the properties of Mersenne numbers instead of polynomial rings. This scheme is defined by four algorithms (Setup, Gen, Enc, Dec) described as follows:

— $\mathsf{Setup}(1^\lambda) \to \mathsf{pp}$, which chooses the public parameters $\mathsf{pp} = (n, h)$ so that $p = 2^n - 1$ is prime and so as to achieve a λ-bit security level. In [AJPS17d] the following lower bound is derived

$$\binom{n-1}{h-1} > 2^\lambda$$

which for instance is satisfied by $\lambda = 120, \mathsf{pp} = (n = 1279, h = 17)$.

— $\mathsf{Gen}(\mathsf{pp}) \to (\mathsf{sk}, \mathsf{pk})$, which picks F, G two n-bit strings chosen independently and uniformly at random from all n-bit strings of Hamming weight h, and returns $\mathsf{sk} \leftarrow G$ and $\mathsf{pk} \leftarrow H = F/G \bmod (2^n - 1)$.

— $\mathsf{Enc}(\mathsf{pp}, \mathsf{pk}, b \in \{0, 1\}) \to c$, which picks A, B two n-bit strings chosen independently and uniformly at random from all n-bit strings of Hamming weight h, then computes

$$c \leftarrow (-1)^b (AH + B) \bmod (2^n - 1).$$

— $\mathsf{Dec}(\mathsf{pp}, \mathsf{sk}, c) \to \{\perp, 0, 1\}$, which computes $D = \|Gc \bmod (2^n - 1)\|$ and returns

$$\begin{cases} 0 & \text{if } D \leq 2h^2, \\ 1 & \text{if } D \geq n - 2h^2, \\ \perp & \text{otherwise} \end{cases}$$

We refer the reader to [AJPS17d] for more details on this cryptosystem which does not require further overview because we directly attack the public key to infer the secret key.

In particular, security rests upon the conjectured intractability of the following problem:

MERSENNE LOW HAMMING RATIO ASSUMPTION. The *Mersenne Low Hamming Ratio Assumption* states that given an n-bit Mersenne prime $p = 2^n - 1$ and an integer h, the advantage of any probabilistic polynomial time adversary attempting to distinguish between $F/G \bmod p$ and R is at most $\frac{\text{poly}(n)}{2^\lambda}$, where R is a uniformly random n-bit string, and (F, G) are independently chosen n-bit strings each having Hamming weight h.

We will argue that (F, G) can be *experimentally* computed from H, at least for the parameter choice $\{n = 1279, h = 17\}$ conjectured in [AJPS17d] as corresponding to a 2^{120} security level.

7.2 OUTLINE OF THE ANALYSIS

The analysis uses the Lenstra–Lenstra–Lovász lattice basis reduction algorithm (LLL, [LLL82]). We do not recall here any internal details of LLL but just the way in which it can be used to solve a linear equation with k unknowns when the total size of the unknowns is properly bounded.

7.2.1 *Using LLL to Spread Information*

Let $x_1, \ldots, x_k \in \mathbb{N}^*$ be k unknowns. Let $p \in \mathbb{N}$ be a modulus and $a_0, \ldots, a_k \in \mathbb{N}$. Consider the equation:

$$a_0 = \sum_{i=1}^{k} a_i x_i \bmod p.$$

Our goal is to use the LLL algorithm to $x_1, \ldots, x_k$ if $\prod_{i=1}^{k} x_i < p$.

In particular, LLL can be adapted to provide any uneven split of sizes between the x_i as long as the sum of those sizes does not exceed the size of p. More details on the theoretical analysis of LLL in that setting and variants are given in [NS01, Sec. 3.2] and [Jou09, Chap. 13], in the context of generalised knapsack problems.

7.2.2 *Partition and Try*

The first observation that attracted our attention is that the size [1] of F (and G) has an unusually small expectation $\sigma(n, h)$:

$$\sigma(n, h) = n \left(1 + \frac{(1 - \frac{h}{n})^{n+1} - 1}{\frac{h}{n}(n+1)} \right)$$

The difference in size between $n = 1279$ and $\sigma(1279, 17)$ is not huge [2] and cannot be immediately exploited. However, the same phenomenon

1. That is, the length of a number, once its leading zeros are discarded.
2. $1279 - \sigma(1279, 17) \approx 75$ bits.

also occurs at the least significant bits and further shortens the expected nonzero parts of F and G by 70 bits.

Similarly, assume that in the key generation procedure, both F and G happen to have bits set to 1 only in their lower halves. When this (rare event) happens, we can directly apply LLL to H to recover F and G. We call this event T.

Is that event rare? Since F and G are chosen at random, T happens with probability at least 2^{-2h}. While T's probability is not cryptographically negligible, this pre-attack only allows to target one key out of 2^{2h}. For the first suggested parameter set ($\lambda = 120$), one public key out of 67 million can be attacked in this fashion and its F and G recovered, i.e., a total break. The question is hence, can this phenomenon be extended to any key? and if so, at what cost? In particular, can we sacrifice work to increase the size of the vulnerable key space? The answers to these questions turn out to be positive, as we will explain hereafter.

RANDOM PARTITIONS. Instead of a fixed partition of $\{0, \ldots, n-1\}$, we can sample random partitions, for instance by sampling (without replacement) m positions, which are interpreted as boundaries between regions of zeros and regions that possibly contain a 1. The total number of regions, $m+1$, determines the dimension of the lattice being reduced.

For the sake of simplicity we consider *balanced* partitions.

BALANCED PARTITIONS. A partition of $\{0, \ldots, n-1\}$ into $m/2$ type 1 blocks and $m/2+1$ type 2 blocks is *balanced* if the total size of the type 1 blocks and the total size of the type 2 blocks differ by at most one.

Since n is odd, we must accept a ± 1 excess.

A randomly sampled partition is not necessarily a balanced partition, to rectify this, we use rejection sampling to ensure the balancing property. We call the sought-after property of these partitions a *correct partition*.

There is room for improvement here as well, since rejection sampling is a very inefficient approach. Nevertheless it will be sufficient for our discussion, and any approach to generating such partitions would work without impacting the analysis.

CORRECT PARTITIONS. Let X be a binary string of length n. A partition of X into $m/2$ type 1 blocks and $m/2+1$ type 2 blocks is *correct for* X if the type 2 blocks are completely made of zeros.

Figure 7.1 illustrates the partitions that we are interested in on a simple example. Also note that the notion of correct partitions does not put any constraint on type 1 blocks, which may contain zeros or not; since they are not guaranteed to be zero we refer to them as "non-zero" blocks. Accordingly, blocks of type 2 in a correct partition are referred to as "zero" blocks. In Figure 7.1 a black square in F or G represents a 1, while white squares represent 0s. The partitions f and g are balanced and correct for F and G respectively, with "zero" blocks coloured white, and "non-zero" blocks coloured black. The vertical dashed lines show how F and G align with their respective partitions.

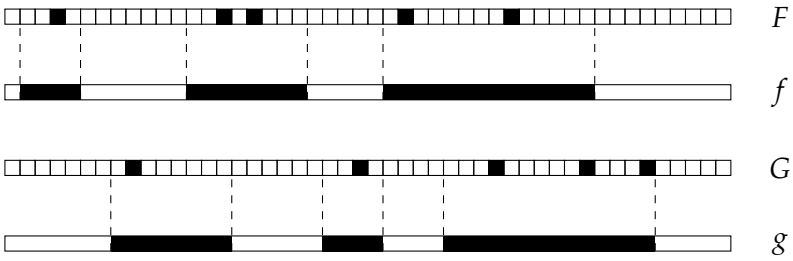


Figure 7.1 – Illustration of *balanced* and *correct* partitions of strings with low Hamming weight.

The observation at the beginning of this section is that using a balanced partition that is correct for F and another one that is correct for G, we can recover F and G from H.

Since F and G are unknown, we cannot construct a correct partition from them directly; but the probability that a random balanced partition is correct for F (resp. G) is lower bounded by 2^{-h}. Assuming that F and G are independent, which they should be according to the key generation procedure, we found a correct partition for *both* F and G with a probability of 2^{-2h}.

We ignore the fact that we sample without replacement here, as $h \ll n$. Under this conservative approximation, all the bits are sampled uniformly and independently, and may fall with probably 1/2 either in a type 1 or a type 2 block.

Remark. We may also consider imbalanced partitions which allow an extra speed-up for a subtle reason: Given that the unknowns found by LLL have a low Hamming density, the odds that these numbers naturally begin by a sequence of zeros (and are hence shorter than expected) is high. The interesting point is that the total length of such natural gains sums up and allows to unbalance the partition in favor of type 1 blocks. Consider the analogy of a fishing boat that can carry up to 1000 kilograms of fish. The fishermen fishes with 3 nets having maximal capacities of 200, 300 and 500 kilograms each. Because waters are sparse in fish, the nets are expected to catch only 70% of their maximal capacity. Hence, we see that larger nets (285, 428, 714) can be used to optimize the boat's fishing capacity. However, unlike the boat, with LLL fish cannot be thrown back to the water and... excess weight sinks the boat (the attack fails). Hence if this speed-up strategy is used, we need to catch more than normal but not be too greedy. Note as well that if all variables end by at least ℓ trailing (LSB) zeros then these $m\ell$ zeros add-up to the gain as well (because there is no constant term in the equation a division of all variables by 2 has no effect on the solution's correctness). We did not exploit nor analyze these tricks in detail.

TRYING PARTITIONS. The attack then consists in sampling a balanced partition, running LLL, and checking whether the values of F and G obtained from the reduction have the correct Hamming weight and yield H by division. Concretely, the matrix to be reduced is obtained as follows from the partitions f of F and g of G:

1. Compute the size of the non-zero blocks in f and g, we call these sizes $\mathbf{u} = \{u_i\}$ and $\mathbf{v} = \{v_i\}$ respectively, with $i = 0, \ldots, m/2 - 1$. Let $w = \max_i\{u_i, v_i\}$.
2. Construct the vector $\mathbf{s} = s_i$ as follows:

$$s_i = \begin{cases} 2^{w-v_i} & \text{if } i < m/2 \\ 2^{w-u_i} & \text{if } m/2 \leq i < m \end{cases}$$

3. Construct the vector $\mathbf{a} = \{a_j\}$ as follows: let f_i (resp. g_i) denote the starting position of the non-zero blocks in F (rep. G), and set

$$a_j = \begin{cases} H \times 2^{g_i} \bmod p & \text{if } j < m/2 \\ p - 2^{f_i} & \text{if } m/2 \leq j < m \end{cases}$$

4. Choose an integer K, and assemble the matrix $\mathbf{M}$ as follows:

$$\mathbf{M} = \begin{pmatrix} \text{diag}(\mathbf{s}) & K\mathbf{a} \\ 0 & Kp \end{pmatrix}$$

   where $\text{diag}(\mathbf{s})$ is the diagonal matrix whose diagonal entries are given by $\mathbf{s}$. The coefficient K is a tuning parameter, which we set to 2^{1200}.
5. Finally, we use LLL on $\mathbf{M}$ (using the Mathematica command `LatticeReduce`) and recover the reduced row of the matrix that complies with the Hamming density of F and G. This row is expected to give the values of the non-zero blocks of F and G, and we can check its correctness by computing its Hamming weight, and checking that the ratio of the candidate values modulo p yield H.

By the above analysis, a given partition is correct with probability 2^{-2h}, which for $\lambda = 120$ is only 2^{-34}; if we can run LLL reasonably fast, which is the case for $m = 16$, an efficient attack happens to be within the reach of a well-equipped organization. Experimental evidence indeed suggests the feasibility of the attack, see Section 7.3.

Remark. For larger security parameters λ, the ratio h/n deduced from the analysis in [AJPS17d] asymptotically vanishes. It should be checked if this influences imbalanced partition finding to the attacker's relative advantage for larger values of λ. We did not explore this avenue left to the reader as a potential research question.

7.3 PUTTING IT TOGETHER

To illustrate the attack's feasibility, we fix a random tape in a deterministically verifiable way and implement our algorithm (see Figure 7.2).

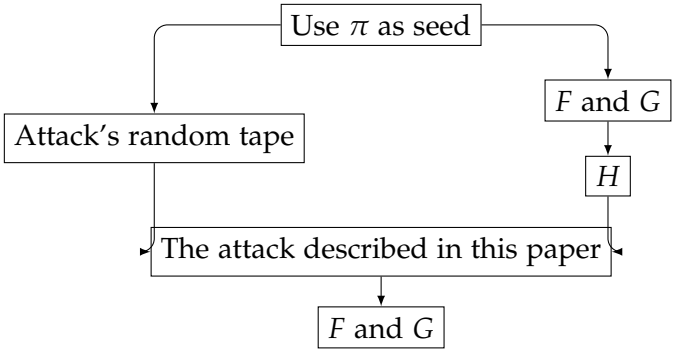


Figure 7.2 – Deriving the attack's random tape from a verifiable source in a deterministic way, as well as the keys.

Proc Gen:
$n, h \leftarrow \mathsf{pp}$
$I_1 = \{i_1, \ldots, i_h\} \leftarrow \text{sample}(\{0, \ldots, n-1\}, h)$
$I_2 = \{i_1, \ldots, i_h\} \leftarrow \text{sample}(\{0, \ldots, n-1\}, h)$
$F \leftarrow \sum_{i \in I_1}^{h} 2^i$
$G \leftarrow \sum_{i \in I_2}^{h} 2^i$
return $(\mathsf{sk} = G, \mathsf{pk} = F \cdot G^{-1} \bmod p)$

Figure 7.3 – The key generation procedure $\mathsf{Gen}(\mathsf{pp})$ for Mersenne-based encryption.

We generated a nothing-up-our-sleeves key with the procedure of Figure 7.3. The sample(S, h) procedure selects h indices without replacement in the range S. It is implemented by returning the h first entries of a deterministic Fisher–Yates shuffle of S. The randomness in sample(S, h) is simulated by iterating the SHA256 function, starting with the seed given by the ASCII representation of the 100 first decimals of π:

```
3141592653589793238462643383279502884197169399375105820974944592307816406286208998628034825342117068
```

In a real attack we would simply use a fast non-cryptographic random number generator, but the above choice serves the purpose of reproducibility.

This gives the following (in hexadecimal notation, the zero MSBs have not been written):

$I_1 = \{\mathtt{33}, \mathtt{47}, \mathtt{8e}, \mathtt{95}, \mathtt{a1}, \mathtt{134}, \mathtt{19f}, \mathtt{1ab}, \mathtt{1ac}, \mathtt{1ce}, \mathtt{25d}, \mathtt{301}, \mathtt{30a}, \mathtt{3ee}, \mathtt{444}, \mathtt{46b}, \mathtt{471}\}$
$I_2 = \{\mathtt{89}, \mathtt{b5}, \mathtt{de}, \mathtt{116}, \mathtt{141}, \mathtt{1dd}, \mathtt{1de}, \mathtt{2ae}, \mathtt{322}, \mathtt{37a}, \mathtt{388}, \mathtt{38a}, \mathtt{3f9}, \mathtt{48c}, \mathtt{48d}, \mathtt{4e9}, \mathtt{4f2}\}$

$F =$ 208000000001000000000000000000004000000000000000000000
000000000000000000000000000000004020000000000000000000
000000000000000000200000000000000000000000000000000040
000001800800000000000000000000000010000000000000000000
000000000002002040000000000000008000080000000000000000

$G =$ 402000000000000000000000300000000000000000000000000000
000020000000000000000000000000005000400000000000000000
004000000000000000000000000000400000000000000000000000
000000000000000000000000060000000000000000000000000000
000040000000002000000000200000000000000000000000000000
000000000002000000000400000000

$H =$ 1610fecf11dbd70f5d09da1244a85c3aa7aed7de75a6d1fe4e988b5f6
6d66e1bc27d46afd96800ff8b2b67316dff1046b88d205e620ba78a81
3c15f47ab8a7d2a8f7eb12fe0fcff882307d92d4c0f9296a7cf4390ce
3140e11e4b7c802fa67d3a8517d30b00980380bdf8992ed6a2d3f74e2
5f14bae21786672bddae4f2bf897f38741cdc10b319f8272d42f738cd
296d4907331518c3439621aefad5c3d1a7c

7.3.1 *Recovering F and G from H*

FINDING A WINNING PARTITION. At this step, we generate random balanced partitions and try LLL on the resulting decomposition. Doing so we quickly find the following partitions

$f = \{2a, bf, 134, 1ec, 233, 253, 25a, 270, 2ee, 32d, 3e4, 41e, 42b, 4a7, 4f6, 4fd\}$
$g = \{7c, 142, 1d0, 22a, 289, 2c8, 2de, 2e7, 2eb, 33c, 372, 3a0, 3da, 3ff, 48a, 4fd\}$

respectively for F and G, which upon lattice reduction yield candidates of the correct Hamming weight. Their ratio indeed gives H; however one may debate our claim that this partition was found at random and argue that we constructed it from our prior knowledge of F and G.

To counter this argument and insist that finding partitions is reasonably easy, we derived them *deterministically from the same seed as the key*. To achieve this, we proceed as follows: we draw two independent sets of $m/2 - 1$ indices in the range $[0, n/2]$, which gives the sizes of the zero blocks and the non-zero blocks. This guarantees that the partitions are balanced. The randomness used for this sampling is obtained by iterating SHA256 as for key generation.

As in the example above, we construct partitions for $m = 16$ — this choice is not dictated by probability (as the likelihood to find a correct partition is in theory independent of m), but rather by a trade-off between the cost of LLL and the number of partitions explored. It

is possible for instance to start with $m = 2$ partitions, then $m = 3$, and so forth, but we settled for a random search which is easier to implement.

We found the following partition for F at run #1,152,006 (in 116 s):

$$f = \{\texttt{27}, \texttt{b2}, \texttt{10e}, \texttt{13c}, \texttt{198}, \texttt{1cf}, \texttt{24b}, \texttt{27b}, \texttt{2ac}, \texttt{30f}, \texttt{3e1}, \texttt{456}, \texttt{45a}, \texttt{4ba}, \texttt{4d6}, \texttt{4fd}\}$$

Experiments with random partitions show that this timing is quite variable and follows a Poisson distribution, with a correct partition being typically found earlier, with an average of 2^{17} tries.

Recovering F alone took about two minutes. Given that we have a totally deterministic random tape, we regard our experiment as legitimately reflecting reality. Because F and G are independent, this brings the total effort to about the square of this number, i.e. about 2^{34} attempts to get both partitions with certainty. Each of these attempts must also involve one LLL, which is the main cost factor.

Using the same sequence, run #64,249 gave a partition for G too (in 7.6 s):

$$g = \{\texttt{7b}, \texttt{11c}, \texttt{13b}, \texttt{181}, \texttt{1cc}, \texttt{1e1}, \texttt{284}, \texttt{2e6}, \texttt{318}, \texttt{329}, \texttt{36f}, \texttt{3e5}, \texttt{3f1}, \texttt{404}, \texttt{476}, \texttt{4fd}\}$$

Finally, note that the task is fully parallelizable and would benefit from running on several independent computers, a remark that we will later use in our final workfactor estimates.

COMPUTING THE SECRET KEY Running our program as explained in Section 7.2, we recover F, G, and confirm that $H = F/G \bmod p$.

7.3.2 *Predicting the Total Execution Time*

Putting all the above figures together and assuming no further algorithmic improvements, the total expected effort is:

$$\frac{(\text{LLL\_Time} + 2 \times \text{Partition\_Time}) \times \text{Average\_Partition\_Tries}^2}{\text{Number\_of\_Processors}}$$

Where, in our basic scenario $\text{Average\_Partition\_Tries} = 2^h$.

We performed LLL in Mathematica using the `LatticeReduce` function, which took less than a second in the worst case on a simple laptop. We safely assume that this figure can be divided by 10 using a dedicated and optimized code. We also assume that a credible attacker can, for example, very easily afford buying or renting 150 TILE-Gx72 multicore processors.

$$\frac{\frac{1}{10} \times 1{,}152{,}006 \times 64{,}249}{150 \times 72} \times \frac{1}{60 \times 60 \times 24} \approx 7 \text{ days } 22 \text{ hours.}$$

Hence, according to the evidence exhibited in this paper, breaking a 1279-bit key takes a week using 150 currently available multicore processors (e.g. TILE-Gx72).

7.4 CONCLUSION

While we did not formally evaluate efficiency nor asymptotic complexities, our quick and dirty experiments clearly suffice to show that key recovery is fast and within reach. An obvious countermeasure consists in increasing parameter sizes. Hence a precise re-evaluation of parameter sizes and safety margins of the Mersenne Low Hamming Ratio Assumption seems in order.

More systemic protections may consist in modifying the definition of H (and possibly the underlying cryptosystem) which is clearly a very interesting open problem.

Nonetheless the beautiful idea of Aggarwal, Joux, Prakash, and Santha exploiting the fact that arithmetics modulo Mersenne numbers is (somewhat) Hamming-weight preserving, is very elegant and seems very rich in possibilities and potential cryptographic applications.

8

PUBLIC-KEY CRYPTOSYSTEMS BASED ON A NEW COMPLEXITY ASSUMPTION

8.1 INTRODUCTION

In 2017, Aggarwal, Joux, Prakash, and Santha [AJPS17a; AJPS17b] introduced a new public-key cryptosystem, inspired by NTRU [HPS98] but conceptually much simpler, and tentatively immune to some of the most classical attacks against NTRU. Since public-key cryptosystems are relatively rare, Aggarwal et al.'s construction (henceforth AJPS-1, following [FN17b]) garnered much attention from the cryptographic community. In a matter of weeks, it was found that AJPS-1's initial security estimates were optimistic, and a modified scheme with larger parameters was proposed [AJPS17b]. Section 8.2 recalls the construction and history of these cryptosystems, which we refer to as Mersenne-based cryptosystems, in more details.

In this paper, we suggest a further modification of the underlying hardness assumption, which are conjectured to be unaffected by attacks against AJPS-1, yet which enable the construction of similarly-elegant encryption schemes. The new assumption, dubbed "projected Mersenne", and a corresponding public-key encryption scheme are introduced in Section 8.4.

8.2 PRELIMINARIES

NOTATIONS. We denote by $\|x\|$ the Hamming weight of x, and by $\mathfrak{H}_{n,w}$ the set of all n-bit strings of Hamming weight h. The notation $x \twoheadleftarrow X$ means that x is the result of uniformly sampling from the set X. Unless stated otherwise, log refers to the natural logarithm, and $\log_2$ to the base 2 logarithm. The symbols $\oplus$ and $\wedge$ stand for the binary XOR and AND operations, respectively. We denote the concatenation of x and y by $x\|y$. A q-ary error correcting code with block length d, dimension k, and minimum Hamming distance δ will be denoted $[d, k, \delta]_q$. We use the shorthand notation $\mathbb{Z}_p$ to denote $\mathbb{Z}/p\mathbb{Z}$. Algorithms are given as input the (unary) representation of the security parameter λ. PPT stands for probabilistic polynomial time.

8.3 PRIOR WORK

8.3.1 *The Mersenne Low Hamming Ratio Assumption*

In particular, if $2^n - 1$ is prime, then so is n.

Recall that a Mersenne number is an integer of the form $2^n - 1$ for some n, and that a Mersenne prime is a Mersenne number which is prime.

The use of a Mersenne prime *is not necessary for the scheme's correctness, and in fact no attack is currently known if p is a Mersenne composite. The conservative choice of a Mersenne* prime *is recommended to avoid potentially unforeseen attacks exploiting the factorisation of p, cf. [AJPS17b, §8].*

MERSENNE LOW HAMMING RATIO SEARCH PROBLEM (MLHR). Let $p = 2^n - 1$ be a Mersenne prime. Given $n, w \in \mathbb{N}$ and $h \in \mathbb{Z}_p$, find $f, g \in \mathbb{Z}_p$ such that $\|f\| = \|g\| = w$ and $f/g = h \bmod p$, under the promise that such a couple exists.

A brute-force attack on MLHR tries all possible couples $\{f, g\}$, which corresponds to a security level of

$$\lambda = \binom{n-1}{w-1} \approx w \cdot \log n \text{ bits.}$$

A quantum variant of this search, exploiting the generic speed-ups provided by Grover's algorithm, correspondingly halves λ. Should these attacks be optimal — as initially suggested by Aggarwal et al. — the MLHR would enable the construction of conceptually-simple and computationally-efficient post-quantum secure public-key cryptosystems.

8.3.2 *The Aggarwal–Joux–Prakash–Santha Cryptosystem (AJPS-1)*

The original AJPS-1 scheme [AJPS17a] is defined by the following algorithms:

— $\mathsf{Setup}(1^\lambda) \to \mathsf{pp}$. Outputs the public parameters $\mathsf{pp} = \{n, h\}$, so that in particular $p = 2^n - 1$ is prime. The choice of n and w is such that the cryptosystem achieves some λ-bit security level.

— $\mathsf{Gen}(\mathsf{pp}) \to \{\mathsf{sk}, \mathsf{pk}\}$. This algorithm generates the private and public keys. It samples $\{F, G\} \twoheadleftarrow \mathfrak{H}^2_{n,w}$, and returns:

$$\begin{aligned} \mathsf{sk} &\leftarrow G \\ \mathsf{pk} &\leftarrow H = F/G \bmod p \end{aligned}$$

— $\mathsf{Enc}(\mathsf{pp}, \mathsf{pk}, m) \to C$. This algorithm takes as input the public parameters pp, the public key pk, and a message $m \in \{0, 1\}$. It samples $\{A, B\} \twoheadleftarrow \mathfrak{H}^2_{n,w}$, and computes:

$$C \leftarrow (-1)^m (AH + B) \bmod p.$$

— $\mathsf{Dec}(\mathsf{pp}, \mathsf{sk}, C) \to \{\perp, 0, 1\}$. This algorithm computes $d \leftarrow \|G \cdot C \bmod p\|$ and returns:

$$\begin{cases} 0 & \text{if } d \leq 2w^2, \\ 1 & \text{if } d \geq n - 2w^2, \\ \perp & \text{otherwise.} \end{cases}$$

Table 8.1 – Synoptic comparison of NTRU and AJPS-1.

	NTRU [HPS98]	AJPS-1 [AJPS17a]
Ciphertext space	$R = \mathbb{Z}[X]/(X^N - 1)$	$R = \mathbb{F}_2[X]/(X^p - 1)$, $p = 2^n - 1$
Message space	$M \in R_M$	$M \in \{0, 1\}$
Private key	$F \in R_F$	$F \in \mathfrak{H}_{n,w}$
Public key	$H = G/F_q \bmod q$ $(G \in R_G)$	$H = G/F$ $(G \in \mathfrak{H}_{n,w})$
Encryption	$C = prH + M \bmod q$ $(r \in R_r)$	$C = (-1)^M(AH + B)$ $(A, B \in \mathfrak{H}_{n,w})$
Decryption	$M = F_p^{-1}(FC \bmod q) \bmod p$	$M = \begin{cases} 0 & \text{if } \lVert FC \rVert \leq 2w^2 \\ 1 & \text{if } \lVert FC \rVert \geq n - 2w^2 \\ \perp & \text{otherwise} \end{cases}$

NTRU stands for N-th Degree Truncated Polynomial Ring Units.

SIMILARITIES WITH THE NTRU CRYPTOSYSTEM. Mersenne-based cryptosystems are reminiscent of NTRU [HPS98], which owes its name to the polynomial ring $R = \mathbb{Z}[X]/(X^N - 1)$ in which operations are performed. In comparison, Mersenne-based cryptosystems work in $\mathbb{Z}_p \simeq \mathbb{F}_2[X]/(X^p - 1)$, where $p = 2^n - 1$ is prime. [1]

Table 8.1 shows the parallels between the two cryptosystems. Notations for NTRU follow [HPS98], except R_f, R_g, R_r, R_m which are subsets of R having a prescribed number of coefficients set to -1 and 1.

The hard problem underlying NTRU is the closest-vector problem (CVP) in some special convolution modular lattices; namely, f and g form a relatively short vector in a known lattice constructed from q and h. Parameters for NTRU must be chosen to resist lattice reduction attacks (e.g., [CS97; KF17]). In the original version of their paper [AJPS17a], Aggarwal et al. consider and then dismiss two possible types of attack that could be better than brute force, inspired by NTRU cryptanalysis: a combinatorial meet-in-the-middle attack, which is claimed to fail due to the presence of "approximate collisions"; and a lattice-based attack, claimed to fail due to the presence of "parasitic vectors".

BEUNARDEAU–CONNOLLY–GÉRAUD–NACCACHE ATTACK. The latter claim was rapidly challenged, when a faster experimental attack using lattice reduction was discovered by Beunardeau et al. [BCGN17b] as presented in Chapter 7, which successfully recovered private keys for the initially suggested $\lambda = 128$ bit security level parameters. This attack runs in time $(2 + \delta + o(1))^{2w}$, for some very small constant $\delta > 0$ [BDJW17], thereby collapsing the security of the original AJPS construction to about $2w$ bits.

DE BOER–DUCAS–JEFFERY–DE WOLF ATTACK. The former claim was also challenged by de Boer et al. [BDJW17], who showed how to circumvent the "approximate collision" problem by leveraging locality-

1. Bernstein et al. [BCLV16] argue against the use of such rings for NTRU.

sensitive hashing. This resulted in a meet-in-the-middle attack, whose complexity is approximately

$$\binom{n/2}{w/2} \approx \binom{n}{w}^{1/2} \approx \frac{1}{2} w \log n.$$

A quantum version of this algorithm has runtime about

$$\binom{n/3}{w/3} \approx \binom{n}{w}^{1/3} \approx \frac{1}{6} w \log n.$$

Pointing out similar work for the related NTRU cryptosystem [Buh98], de Boer et al. conjecture that a combination of the MITM approach with lattice reduction could lead to an even faster attack, reminiscent for instance of Howgrave-Graham's [How07].

8.3.3 *Aggarwal–Joux–Prakash–Santha with Error Correction (AJPS-ECC)*

To thwart the effect of the above attacks, Aggarwal et al. proposed a new version of their cryptosystem.

The new version accomodates larger parameters and also improves somewhat the cryptosystem's bandwidth. As it makes use of an error correction scheme ECC $= \{\mathcal{D}, \mathcal{E}\}$, we refer to it as AJPS-ECC (following [FN17b]). The Setup algorithm is unmodified. The other algorithms are modified as follows:

— $\mathsf{Gen}(\mathsf{pp}) \to \{\mathsf{sk}, \mathsf{pk}\}$. Sample $\{F, G\} \twoheadleftarrow \mathfrak{H}^2_{n,w}$, $R \twoheadleftarrow \{0,1\}^n$ and return:

$$\begin{aligned} \mathsf{sk} &\leftarrow F \\ \mathsf{pk} &\leftarrow \{R, T\} = \{R, F \cdot R + G \bmod p\} \end{aligned}$$

— $\mathsf{Enc}(\mathsf{pp}, \mathsf{pk}, M) \to C$. Sample $\{A, B_1, B_2\} \twoheadleftarrow \mathfrak{H}^3_{n,w}$, and compute

$$C \leftarrow \begin{cases} C_1 = A \cdot R + B_1 \bmod p \\ C_2 = (A \cdot T + B_2 \bmod p) \oplus \mathcal{E}(M) \end{cases}$$

— $\mathsf{Dec}(\mathsf{pp}, \mathsf{sk}, C) \to \{\perp, M\}$ is modified accordingly and returns

$$\mathcal{D}((F \cdot C_1 \bmod p) \oplus C_2).$$

An analysis of the parameter choices for ECC and for the cryptosystem, including some additional discussion not relevant for our purpose, can be found in the updated paper by Aggarwal et al. [AJPS17b]. The careful reader will notice that AJPS-ECC relies for security on *different assumption* than that of the hardness of MLHR search; however, Aggarwal et al. point out that slight modifications to the attack presented in Chapter 7 also apply to this modified scheme and choose the parameters accordingly.

8.3.4 *Ferradi–Naccache (AJPS-FN-BT)*

An interesting collection of variants is described by Ferradi and Naccache [FN17b]. Noticing that some of the random coins used during encryption may be recovered, Ferradi and Naccache suggest turning this into a feature, thereby increasing the cryptosystem's bandwidth. However, the security of most of these variants is not discussed in depth. The core idea can be found in Ferradi and Naccache's "bivariate" variant AJPS-FN-BT2.

AJPS-FN-BT2 relies on the availability of an efficient function $\mathsf{Solve}_{x,y}$ which finds a low Hamming weight solution to a given Diophantine equation of the form $\alpha x + \beta y + \gamma = 0$ for given parameters α, β, γ. They suggest implementing this function as a heuristic-based backtracking algorithm. Using this, it becomes possible to recover the values A and B used during encryption, which have low Hamming weight. One possibility is to use A and B to design a key-encapsulation mechanism as follows:

— Setup and Gen are identical to those of AJPS-1, except that we additionally agree on a block cipher $\mathrm{BC} : \{0,1\}^\lambda \times \{0,1\}^\lambda \to \{0,1\}^\lambda$, a cryptographic hash function $\mathrm{H}_2 : \{0,1\}^* \to \{0,1\}^\lambda$, and a cryptographic hash function $\mathrm{H}_1 : \{0,1\}^* \to \mathfrak{H}^2_{n,w}$.

— $\mathsf{Enc}(\mathsf{pp}, \mathsf{pk}, M) \to C$ is modified as follows. Sample $r \twoheadleftarrow \{0,1\}^\lambda$ and compute $\{A, B\} \twoheadleftarrow \mathrm{H}_1(r \| M)$. Then compute $K \leftarrow \mathrm{H}_2(A \| B)$. Finally, output

$$C = \{C_1, C_2\} = \{AH + B \bmod p,\ \mathrm{BC}_k(r \| m)\}\,.$$

— $\mathsf{Dec}(\mathsf{pp}, \mathsf{sk}, C) \to \{\perp, M\}$ is modified as follows: first recover

$$\{A, B\} \leftarrow \mathsf{Solve}_{x,y}\,[GC_1 = Fx + Gy \bmod p]\,.$$

In case of failure, return $\perp$. Otherwise, compute $K \leftarrow \mathrm{H}_2(A \| B)$, and recover $u \leftarrow \mathrm{BC}_k^{-1}(C_2)$. If $\mathrm{H}_1(u) \neq \{A, B\}$ then return $\perp$. Otherwise return M.

The correctness of this scheme (and other variants in [FN17b]) is not formally analysed but is backed by numerical simulations.

8.4 THE PROJECTED-MERSENNE CRYPTOSYSTEM

8.4.1 *The Projected-Mersenne assumption*

We introduce the following problem:

PROJECTED-MERSENNE LOW HAMMING RATIO SEARCH PROBLEM. Let $p = 2^n - 1$ be a Mersenne prime. Given $n, w, d \in \mathbb{N}$, $M = 2^d - 1$, and $h \in \mathbb{Z}_p$, find $f, g \in \mathbb{Z}_p$ such that

1. $\|g\| = w$

2. $\|f \wedge M\| = 1$
3. $f/g = h \bmod p$

under the promise that such a couple exists.

This search problem can be solved by brute-force enumeration much like MLHR, as it suffices to find g, i.e., find one in $\binom{n-1}{w-1} \approx 2^{w \log n}$ possibilities. We introduce the following assumption:

We choose to explicit only one parameter, namely the random numerator's size. The other parameter is the denominator's Hamming weight, which will be the same throughout our different variants, and therefore will not be explicitly noted.

a-Projected Mersenne assumption. The *a-projected Mersenne assumption* states that given a Mersenne prime $p = 2^n - 1$, an integer a in $\mathsf{poly}(\lambda)$, any PPT distinguisher has a negligible chance to distinguish between R/G and R', where $R \twoheadleftarrow \{0, \ldots, 2^a - 1\}$, $G \twoheadleftarrow \mathfrak{H}_{n,w}$, and $R' \twoheadleftarrow \mathbb{Z}_p$.

where the distinguishing advantage is defined as usual:

For a PPT distinguisher $\mathcal{D}$ that outputs a bit $b \in 0, 1$, the distinguishing advantage to distinguish between two random variables X and Y is defined as:

$$\Delta^{\mathcal{D}}(X;Y) = |\Pr[\mathcal{A}(X) = 1] - \Pr[\mathcal{D}(Y) = 1]|$$

8.4.2 *Projected-Mersenne Encryption*

We now define our encryption scheme.

— $\mathsf{Setup}(1^\lambda) \to \mathsf{pp}$. Choose $p = 2^n - 1$ a Mersenne prime, and parameters w, a, b, c, d so as to achieve a λ-bit security level. Additional constraints on a, b, c, d to ensure correctness are discussed below. We also agree on an error-correcting code $\mathrm{ECC} = (\mathcal{E}, \mathcal{D})$ with codewords of size d bits.

A possibility is to use BCH codes [Hoc59; BRC60] which are efficient and give fine control over the code's parameters, or Reed–Solomon codes [RS60] which are MDS.

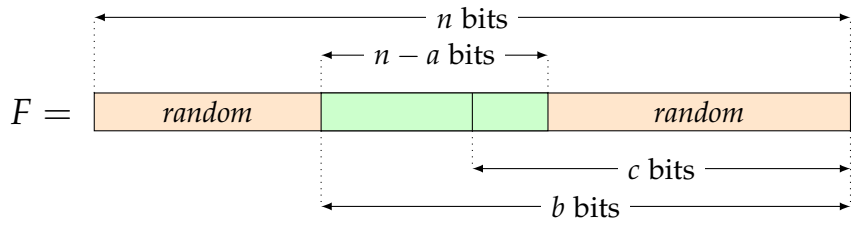
— $\mathsf{Gen}(\mathsf{pp}) \to \{\mathsf{sk}, \mathsf{pk}\}$. Sample $G \twoheadleftarrow \mathfrak{H}_{n,w}$ and a random a-bit number R that has large Hamming weight (e.g., $\|R\| = a/2$). Let $F \leftarrow R \cdot 2^b + 2^c \bmod p$. Note that, in general, $F \notin \mathfrak{H}_{n,w}$. An illustration of F's structure is given in Figure 8.1. The Gen algorithm returns

$$\begin{aligned} \mathsf{sk} &\leftarrow G \\ \mathsf{pk} &\leftarrow F/G \bmod p \end{aligned}$$

— $\mathsf{Enc}(\mathsf{pp}, \mathsf{pk}, M) \to C$. Sample $B \twoheadleftarrow \mathfrak{H}_{n,w}$ and return $C \leftarrow \mathcal{E}(M) \cdot \mathsf{pk} + B \bmod p$.

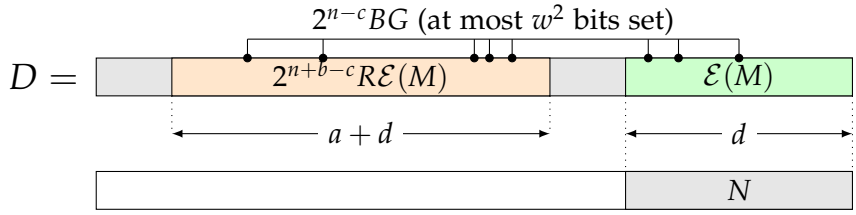
— $\mathsf{Dec}(\mathsf{pp}, \mathsf{sk}, C) \to \{M, \perp\}$. First compute $D \leftarrow 2^{n-c} C \cdot \mathsf{sk} \bmod p$. This should be

$$\begin{aligned} D &= 2^{n-c} C \mathsf{sk} \\ &= 2^{n-c} \left(\mathcal{E}(M) \cdot H + B\right) \cdot G \bmod p \\ &= 2^{n-c} \mathcal{E}(M) F + 2^{n-c} BG \bmod p \\ &= 2^{n-c} \mathcal{E}(M) \cdot \left(2^b R + 2^c\right) + 2^{n-c} BG \bmod p \\ &= \mathcal{E}(M) + 2^{n+b-c} R \mathcal{E}(M) + 2^{n-c} BG \bmod p \end{aligned}$$



A central region of size $n - a$ contains only a single set bit. Note that this figure and the following are not to scale, we give concrete parameters later.

Figure 8.1 – An illustration of the structure in F, as used in our Gen algorithm.



For appropriately chosen parameters, projecting by N only retains a noisy version of $\mathcal{E}(M)$.

Figure 8.2 – An illustration of the structure in D, as used in our Dec algorithm.

Let $N = 2^d - 1$, then the algorithm outputs $\mathcal{D}(N \wedge D)$. Figure 8.2 illustrates this process.

8.4.3 *Correctness*

The correctness of this scheme is based upon two facts. The first is that we can appropriately choose a, b, c, d so that the first and second term in the expanded expression of D are disjoint (as depicted in Figure 8.2). When this is the case, masking by N removes the $2^{n+b-c}R\mathcal{E}(M)$ term. The conditions for this to happen are easily found by inspecting Figures 8.1 and 8.2:

$$n \leq a + 2d \qquad \text{and} \qquad a + b - n < c < b$$

Assuming an ECC is given with block length d, we can choose the following parameters, which correspond to maximising a:

$$a = n - 2d \qquad \text{and} \qquad b \twoheadleftarrow \{0, \ldots, n\} \qquad \text{and} \qquad c = b - d \bmod p.$$

The second key fact is that BG has low Hamming weight, namely at most w^2. This is not affected by multiplication by a power of 2, and therefore $2^{n-c}BD \wedge M$ has Hamming weight at most $\eta = \min(d, w^2)$.[2] If ECC can correct at least η errors, then the decoding algorithm succeeds.[3] Even in the case that ECC can only correct $t < \eta$ errors,

2. If ECC is a linear code, then $w^2 \leq d - k + 1$, or in other terms, $k \leq d - w^2 + 1$. Therefore, $\eta = d$.

3. In fact, the noise considered here is *additive*, and may result in more than w^2 bits being affected due to the carry. We may choose a stronger error correction capacity to account for such unlikely events.

there is still a non-zero probability that Dec successfully recovers M, which corresponds to the events where all $\eta - t$ bits lay between 2^d and $2^n - 1$; this probability is roughly $(1 - 2^{d-n})^{\eta-t}$.

8.4.4 *Semantic Security*

As described above, the cryptosystem is vulnerable to a trivial chosen ciphertext attack. Assume the attacker gets a challenge encryption C^* of M_b with $b \in \{0, 1\}$ being the challenge bit they have to guess. Indeed, we get

$$C^* = \mathcal{E}(M) \cdot \mathsf{pk} + B \bmod p$$

Thanks to the knowledge of the public key they will be able to recover the randomness (and break the semantic security). They compute the encrpytion with the randomness being set to 0.

$$C_0 = \mathcal{E}(M_0) \cdot \mathsf{pk} + B \bmod p$$
$$C_1 = \mathcal{E}(M_1) \cdot \mathsf{pk} + B \bmod p$$

We then subtract the ciphertexts:

$$C^* - C_0 = (\mathcal{E}(M_b) - \mathcal{E}(M_0)) \cdot \mathsf{pk} + B \bmod p$$
$$C^* - C_1 = (\mathcal{E}(M_b) - \mathcal{E}(M_1)) \cdot \mathsf{pk} + B \bmod p$$

One of these equations is equal to B and has low hamming weight, which allows the adversary to distinguish. This phenomenon is common to other cryptosystems, such as the McEliece code-based encryption scheme. Therefore our system is inherently not CCA-secure. We can treat this problem by using a key encapsulation mechanism which encrypts a random message, and derive the randomness used in the encryption by hashing the random message. AJPS uses the same method, but they only need it for chosen ciphertext attacks.

8.5 KEY ENCAPSIATION MECHANISM

To keep consistency with AJPS, we use the same notation to define KEM to ease comparison.

KEY ENCAPSULATION MECHANISM SYNTAX. A key encapsulation mechanism KEM is a tuple of PPT algorithms $(\mathsf{Gen}, \mathsf{Encaps}, \mathsf{Decaps})$ such that:

1. The key generation algorithm Gen takes as input the security parameter 1^λ and outputs a public/private key pair $(\mathsf{pk}, \mathsf{sk})$.
2. The encapsulation algorithm Encaps takes as input a public key pk and the security parameter 1^λ. It outputs a ciphertext C and a key $k \in \{0,1\}^{\ell(\lambda)}$ where ℓ is the key length. We write $(C, K) \leftarrow \mathsf{Encaps}_{\mathsf{pk}}(1^\lambda)$

3. The deterministic decapsulation algorithm Decaps takes as input a private key sk and a ciphertext C, and outputs a key K_0 or an error symbol $\perp$ denoting failure. We write this as $K_0 := \mathsf{Decaps}_{\mathsf{sk}}(C)$.

It is required that with all but negligible probability over $(\mathsf{sk}, \mathsf{pk})$ output by $\mathsf{Gen}(1^\lambda)$, if $\mathsf{Encaps}_{\mathsf{pk}}(1^\lambda)$ outputs (c, k) then $\mathsf{Decaps}_{\mathsf{sk}}(C)$ outputs K_0.

KEM SEMANTIC SECURITY. The key-encapsulation mechanism KEM $(\mathsf{Gen}, \mathsf{Encaps}, \mathsf{Decaps})$ is said to be semantically secure if for any probabilistic polynomial time distinguisher, given the public key pk, the advantage for distinguishing (C, K_0) and (C, K_1), where $(C, K_0) \twoheadleftarrow \mathsf{Encaps}(\mathsf{pk})$ and K_1 is uniform and independent of C is negligible in λ.

KEM SEMANTIC SECURITY UNDER CHOSEN CIPHERTEXT ATTACK.

The key-encapsulation mechanism $\mathsf{KEM} = (\mathsf{Gen}, \mathsf{Encaps}, \mathsf{Decaps})$ is said to be secure under chosen ciphertext attacks if for any probabilistic polynomial time distinguisher that is given access to the decapsulation oracle and the public key pk, the advantage for distinguishing (C, K_0) and (C, K_1), where $(C, K_0) \twoheadleftarrow \mathsf{Encaps}(\mathsf{pk})$ and K_1 is uniform and independent of C is negligible in λ under the assumption that the distinguisher does not query the oracle with C.

We now describe our KEM. Its purpose is to avoid encrpytion with randomness set to 0 as in the attack against our encyption scheme. To achieve this, we will sample a key at random, and use it with a random oracle to generate the randomness to encrypt the key. Let H be a random oracle from the key space $\{0,1\}^\lambda$ be the random tape of our encryption scheme (ie. low hamming weight numbers).

— Gen remains unchanged from the encryption scheme.
— $\mathsf{Encaps}(\mathsf{pk})$ draws uniformly at random a key K, produces the ciphertext $\mathsf{Enc}(K) \cdot \mathsf{pk} + \mathrm{H}(K)$ and the key K.
— $\mathsf{Decaps}(\mathsf{sk}, C)$ produces the key $K' = \mathsf{Dec}(\mathsf{sk}, C)$, reencrypts its own randomness $C' = \mathsf{Enc}(K') \cdot \mathsf{pk} + \mathrm{H}(K')$, and checks $C = C'$. If $C \neq C'$ output $\perp$, else output K.

Our KEM is trivially $1 - \delta$-correct with δ negligible in λ from the correctness of the encryption scheme.

8.5.1 *Security Analysis*

In this section we show that the KEM's semantic security (Definition 8.5) relies on the Projected- Mersenne Assumption (Definition 8.4.1), and discuss the attacks that can apply to this assumption.

8.5.1.1 *Semantic Security of the Key Encapsulation Mechanism*

Theorem 8.1. *[Semantic Security] Our KEM is semantically secure (Definition 8.5) under the a-Projected-Mersenne Assumption (Definition 8.4.1), with a as defined in Figure 8.1.*

Before proving Theorem 8.1 we give a few lemmas that we will use later. The main thing to prove is that the public key is indistinguishable from random, which is shown in Lemma 8.5.

Lemma 8.2. *Given a PPT computable function f on two random variables X and Y , if there is no PPT distinguisher D that can distinguish between X and Y with non negligible advantage, then there is no PPT distinguisher D' that can distinguish between X and Y with non negligible advantage.*

The proof of Lemma 8.2 is well known and can be found easily.

Lemma 8.3. *If $x \in \mathbb{Z}_p^*$ is of hamming weight 1, then $x - 1$ is of hamming weight 1.*

Proof. Since the multiplicative group $\mathbb{Z}_p^*$ is of order $p - 1$, we have $x^{-}1 = x^{p-2}$. Since we work modulo a Mersenne prime, multipliying by a number of hamming weight 1 is equivalent to shifting. Therefore taking the product of two numbers of hamming weight 1 is also of hamming weight 1. Since $x^{-}1$ is the product of numbers of hamming weight 1, it is itself of hamming weight 1. □

Lemma 8.4 also gives an intuition about our security. For example doing the same computation on the public key of AJPS-1, we would get a much smaller number of contributions (17/2 for 128-bits of security).

Lemma 8.4. *Every bit of the public key is the sum of an average of $a/2$ bits of R omitting the contribution of the carry*[4].

Proof. First we notice that although $1/G$ is not random looking (as shown in [BCGN17c]), it has a random hamming weight ($n/2$ on average). Indeed $1/G = G^p - 2$. So its low hamming weight increases since we perform approximately n squarings to come to the inverse. Second, since R is of size a, every bit of the public key is influenced by a copy of R if a bit in the a bits preceding it is set to 1 in $1/G$. Combining the two observations gives the result. □

The intuition of the security of our scheme is in this lemma. The message is written using this 2^c in the public key. We basically show that dividing by G makes that the 2^c that will contain the message masked by R.

Lemma 8.5. *Assuming the a-Projected-Mersenne Assumption (Def. 8.4.1), given a Mersenne prime $p = 2^{n-1}$, an integer a output by* Setup*, any PPT disinguisher has a negligible chance to distinguish between*

$$\mathsf{pk}R'$$

where pk *is the public key generated from* Gen *and $R' \twoheadleftarrow \mathbb{Z}_p$*

Here we see the reduction is not tight : for 128-bits of security the probability is $1 - 2^{69}$.

Proof. Let $\mathsf{pk} = 2^b \cdot R/G + 2^c \cdot 1/G \bmod p$. Applying Lemma 8.2 with f being the division by 2^b, the adversary tries to distinguish $\text{chal} = R/G + \frac{2^{c-b}}{G}$. By Lemma 8.3, 2^{c-b} is of hamming weight 1. So we can rewrite $\text{chal} = \frac{R+2^i}{G}$ for some i. Applying Lemma 8.4, with overwhelming probability, every additional 1 coming from $2^i/G$ will be in a copy of R. Since there are as many copies of R as there are 1s coming from $2^i/G$ (this number being $||1/G||$, the hamming weight of $1/G$), we can rewrite the challenge:

4. We omit the the carry's influence to simplify analysis: we only need a bound.

$$chal = \sum_{i=0}^{||1/G||} 2^{u_i} R_i$$

for some u_i with $R_i = R + 2^x$ and with $x \twoheadleftarrow 0, a-1$. We now show that we can replace one R_i by R. Since x is taken at random with overwhelming probability an R_i is indistinguishable from R since their distributions are statistically close. Indeed $R \twoheadleftarrow [0, 2^{a-1}]$ and $R_i \twoheadleftarrow [2^x, 2^x + 2^{a-1}]$. X is on average $a/2$, and the statistical distance is $a/2$. With overwhelming probability the statistical distance is negligible.

Once again the reduction is not tight. One could try to make it tighter by choosing which R goes with which 1, to minimize x

Since there are polynomially many R_is, one can replace them one by one to get from chal to R/G while staying indistinguishable. □

Lemma 8.6. *Assuming the a-Projected Mersenne Assumption (Def. 8.4.1), given a Mersenne prime $p = 2^{n-1}$, an integer a, any PPT distinguisher has a negligible chance to distinguish between G/R and R' whereR $\twoheadleftarrow [0, 2^{a-1}]$, $G \twoheadleftarrow \mathfrak{H}_{n,w}$ and $R' \twoheadleftarrow \mathbb{Z}_p$.*

Proof. This is shown by applying Lemma 8.2 with f being the modular inversion. □

We can now prove our main theorem 8.1.

Proof. For any PPT distinguisher D, we have by the triangle inequality:

$$\begin{aligned}\delta^D((\mathsf{pk}, C); (\mathsf{pk}, R)) \leq\ & \delta^D((\mathsf{pk}, C); (R, \mathsf{Enc}(M)R + B)) \\ & + \delta^D((R, \mathsf{Enc}(M)R + B); (R, R')) \\ & + \delta^D((R, R'); (\mathsf{pk}, R))\end{aligned}$$

where pk is a public key genrated from Gen, M is drawn at random, $B \twoheadleftarrow \mathfrak{H}_{n,w}$, $C = \mathsf{Enc}(M)\mathsf{pk} + B$ is a ciphertext, and $R, R' \twoheadleftarrow \mathbb{Z}_p$. This suffices to show the semantic security.

We now have to show that the three bounding terms are negligible:

— $\delta^D((\mathsf{pk}, C); (R, \mathsf{Enc}(M)R + B))$. By Lemma 8.5 we have that pk is indistinguishable from a random, applying Lemma 8.2 with $f(X) = (X, \mathsf{Enc}(M)X + B)$ with M a message drawn at random and $B \twoheadleftarrow \mathfrak{H}_{n,w}$

— $\delta^D((R, \mathsf{Enc}(M)R + B); (R, R'))$. By Lemma 8.6 we have that B/R is indistinguishable from random, applying Lemma 8.2 with $f(X) = (R, X \cdot R' + \mathsf{Enc}(M))$ we get that

$$\delta^D((R, \mathsf{Enc}(M)R + B); (R, R \cdot R' + \mathsf{Enc}(M)))$$

is negligible. By observing that $R \cdot R' + \mathsf{Enc}(M)$ is uniformly distributed, $\delta^D((R, \mathsf{Enc}(M)R + B); (R, R'))$ is negligible.

— $\delta^D((R, R'); (\mathsf{pk}, R))$. This is shown to be negligible by Lemma 8.5 and Lemma 8.2 with $f(X) = (X, R)$ where $R \twoheadleftarrow \mathbb{Z}_p$

□

8.5.1.2 *Chosen Ciphertext Security*

We now show that security holds against chosen ciphertext (Definition 8.5). For this we only need to show that the queries will not help the adversary, and then conclude with semantic security. The key point is that the decapsulation oracle to which the adversary has access will answer with overwhelming probability if the ciphertext are not made 'honestly'. Since once the key is fixed and the random oracle called with it the encapsulation procedure is deterministic, the adversary can simulate it easily.

Theorem 8.7. *[Semantic Security under Chosen Ciphertext Attack] Our KEM is semantically secure under chosen ciphertext attack (Definition 8.5) under the a-Projected-Mersenne Assumption (Definition 8.4.1), with a as defined in Figure 8.1.*

Proof. There are two cases, either the random oracle was called on the answer of a query to the decapsulation mechanism, or it was not (ie. the adversary tries 'malicious' queries).

— If the adversary queries Decaps with a ciphertext, and receives K, which was previously queried to the random oracle, then simulation is trivial.

— The probability of the second event is $\Pr[K \leftarrow \mathsf{Decaps}(C \twoheadleftarrow \mathbf{Adv})] = \Pr[\mathsf{Enc}(\mathsf{Dec}(C)) \cdot \mathsf{pk} + \mathrm{H}(K) = C \twoheadleftarrow \mathbf{Adv}] \leq \binom{n}{w}$ since it requires guessing the random oracle response. The second event is therefore negligible.

□

8.5.2 *Attacks on the Underlying Assumption*

Due to the similarity with AJPS, it is natural to discuss the attacks that are most efficient against it, and to measure to what extent such attacks apply to our new construction.

8.5.2.1 *Lattice Attacks*

As with other versions of Mersenne encryption the attack presented in Chapter 7 is also applicable here and has an experimental cost of finding the right partitions for the low Hamming weight. We then set $w = \lambda$.

8.5.2.2 *Brute Force Attacks*

A brute force exhaustion of sk is always possible, and takes an effort of $\binom{n-1}{w-1}$. Thus the bare minimum requirement for security is that this quantity exceeds 2^{λ}

8.5.2.3 *Meet-in-the-Middle Attacks*

The key result of de Boer et al. is backed by [BDJW17, Lemma 3.1], which assumes that F has constant small Hamming weigth w. Without this assumption, the likelihood that a locality-sensitive hash function H is "good" for g does not have a lower bound, so that the meet-in-the-middle attack is no longer guaranteed to succeed. We can compare simulations from [BDJW17, Appendix A.1] with the same experiment on our scheme, which shows that de Boer et al.'s Heuristic 3.2, which is reasonable against AJPS-1, does not hold for our scheme.

8.5.3 *Conclusion*

Altough our scheme is vulnerable to the same attacks as AJPS, it is simpler in the sense that we do not need two ciphertexts. We hoped that the size of the random R would have twart our lattice attack. This is not the case experimentally, but since the analysis of our attack is not complete, there is still hope that our scheme is of interest. We also think that it is simpler to analyse our assumption than AJPS's.

Part IV

FAIL BETTER

Abstract

The provably secure Schnorr signature scheme is popular and efficient. However, each signature requires a fresh modular exponentiation, which is typically a costly operation. As the increased uptake in connected devices revives the interest in resource-constrained signature algorithms, we introduce a variant of Schnorr signatures that mutualises exponentiation efforts.

Combined with precomputation techniques (which would not yield as interesting results for the original Schnorr algorithm), we can amortise the cost of exponentiation over several signatures: these signatures share the same nonce. Sharing a nonce is a deadly blow to Schnorr signatures, but is not a security concern for our variant, dubbed ReSchnorr.

ReSchnorr is provably secure, asymptotically-faster than Schnorr when combined to efficient precomputation techniques, and experimentally 2 to 6 times faster than Schnorr for the same number of signatures when using 1 MB of static storage.

This is joint work with Marc Beunardeau, Rémi Geraud, David Naccache, and Damien Vergnaud. This work was accepted at the 22nd European Symposium on Research in Computer Security, ESORICS 2017, Oslo (Norway) and published as [BCGNV17].

9

REUSING NONCES IN SCHNORR SIGNATURES

9.1 INTRODUCTION

The increased popularity of lightweight implementations invigorates the interest in resource-preserving protocols. Interestingly, this line of research was popular in the late 1980's, when smart-cards started performing public-key cryptographic operations (e.g. [FS87]). Back then, cryptoprocessors were expensive and cumbersome, and the research community started looking for astute ways to identify and sign with scarce resources.

In this work we revisit a popular signature algorithm published by Schnorr in 1989 [Sch90] and seek to lower its computational requirements assuming that the signer is permitted to maintain some read-only memory. This storage allows for time-memory trade-offs, which are usually not very profitable for typical Schnorr parameters.

We introduce a new signature scheme, ReSchnorr, which is provably secure in the random oracle model (ROM) under the assumption that the *partial discrete logarithm problem* (see below) is intractable. This scheme can benefit much more from precomputation techniques, which results in faster signatures.

Implementation results confirm the benefits of this approach when combining efficient precomputation techniques, when enough static memory is available (of the order of 250 couples of the form (x, g^x)). We provide comparisons with Schnorr for several parameters and pre-computation schemes.

INTUITION. The Schnorr signature algorithm uses a large prime modulus p and a smaller prime modulus q dividing $p-1$. The security of the signature scheme relies on the discrete logarithm problem in a subgroup of order q of the multiplicative group of the finite field $\mathbb{Z}_p$ (with $q \mid p-1$). Usually the prime p is chosen to be large enough to resist index-calculus methods for solving the discrete-log problem (*e.g.* 3072 bits for a 128-bit security level), while q is large enough to resist the square-root algorithms [Sha71] (e.g. 256 bits for 128-bit security level).

The intuition behind our construction is to consider a prime p such that $p-1$ has *several* different factors q_i large enough to resist these birthday attacks, i.e.

$$p = 1 + 2\prod_{i=1}^{\ell} q_i$$

then several "orthogonal" Schnorr signatures can share the same commitment component $r = g^k \bmod p$. This is not the case with standard Schnorr signatures where, if a k is reused then the secret signing key is revealed.

It remains to decide how r can be computed quickly. In the original Schnorr protocol k is sampled uniformly at random from $\mathbb{Z}_q$. However, to be secure, our construction requires that k is chosen from the larger set $\mathbb{Z}_{p-1}$. which means that a much higher effort is required to compute r. Here we cut corners by generating an r with pre-computation techniques which allow an exponentiation to be sub-linear. The trick is that once the exponentiation is sub-linear, we are more effective in our setting than in the original Schnorr setting.

We start by reminding how the original Schnorr signature scheme works and explain our extension assuming that k is randomly drawn from $\mathbb{Z}_{p-1}$. We then present applications of our construction, by comparing several pre-processing schemes.

9.2 SCHNORR SIGNATURES

Schnorr signatures [Sch90] are an offspring ElGamal signatures [ElG86] which are provably secure in the Random Oracle Model under the assumed hardness of solving generic instances of the Discrete Logarithm Problem (DLP) [PS96]. For convenience, we restate the definition from Chapter 2 for ease of comparison here.

SCHNORR SYNTAX. The Schnorr signature scheme is a tuple of algorithms defined as follows:

— $\mathsf{Setup}(1^\lambda)$: Large primes p, q are chosen, such that $q \geq 2^\lambda$ and $p - 1 = 0 \bmod q$. A cyclic group $\mathbb{G} \subset \mathbb{Z}_p$ of prime order q is chosen, in which it is assumed that the DLP is hard, along with a generator $g \in \mathbb{G}$. A hash function $\mathrm{H} : \{0,1\}^* \to \mathbb{G}$ is chosen. Public parameters are $\mathsf{pp} = (p, q, g, \mathbb{G}, \mathrm{H})$.

— $\mathsf{Gen}(\mathsf{pp})$: Pick an integer x uniformly at random from $[2, q-1]$ as the signing key sk, and publish $y \leftarrow g^x$ as the public key pk.

— $\mathsf{Sign}(\mathsf{pp}, \mathsf{sk}, M)$: Pick k uniformly at random in $\mathbb{Z}_q^*$, compute $r \leftarrow g^k \bmod q$, $e \leftarrow \mathrm{H}(M, r)$, and $s \leftarrow k - ex \bmod q$. Output $\sigma \leftarrow \{r, s\}$ as a signature.

— $\mathsf{Verify}(\mathsf{pp}, \mathsf{pk}, M, \sigma)$: Let $(r, s) \leftarrow \sigma$, compute $e \leftarrow \mathrm{H}(M, r)$ and return True if $g^s y^e = r$, and False otherwise.

SCHNORR SECURITY. We recall the strong [1] EUF-CMA security notion: A signature scheme Σ is *secure against existential forgeries in a chosen-message attack* (strongly EUF-CMA-secure) if the advantage of any

1. In contrast to the *weak* version, the adversary is allowed to forge for a message that they have queried before, provided that their forgery is *not* an oracle response.

Game $\mathsf{EUF\text{-}CMA}^{\mathcal{A}}_{\Sigma}(\lambda)$:
$(\mathsf{sk}, \mathsf{pk}) \twoheadleftarrow \mathsf{Gen}(1^\lambda)$
$(M^*, \sigma^*) \leftarrow \mathcal{A}^{\mathsf{Sign}(\cdot),\mathsf{Verify}(\cdot,\cdot),H(\cdot)}(1^\lambda)$
if $(M^*, \sigma^*) \notin \mathsf{L}$
    return $\mathsf{Verify}(\mathsf{pk}, M^*)$
return $\bot$

$\mathsf{Sign}(M)$:
$\sigma \twoheadleftarrow \mathsf{Sign}(\mathsf{sk}, M)$
$\mathsf{L} \leftarrow \mathsf{L} \cup \{M, \sigma\}$
return σ

$\mathsf{Verify}(M, \sigma)$:
return $\mathsf{Verify}(\mathsf{pk}, M, \sigma)$

Figure 9.1 – The strong EUF-CMA experiment for digital signature schemes.

PPT adversary $\mathcal{A}$ against the EUF-CMA game defined in Figure 9.1 is negligible:

$$\mathbf{Adv}^{\mathsf{euf\text{-}cma}}_{\Sigma,\mathcal{A}}(\lambda) = \Pr\left[\mathsf{EF}^{\mathcal{A}}_{\Sigma}(\lambda) = 1\right] \in \mathrm{Negl}(\lambda)$$

9.3 RESCHNORR SIGNATURES

Our construction relies on using a prime p of the form mentioned in the introduction. This is not a trivial change, and requires care as we discuss below.

Technically, our construction is a *stateful* signature scheme (see e.g. [KL07, Chapter 12]), in which we simultaneously sign only one message and keep a state corresponding to the values k, g^k and the index i for the current prime number. However, it is more compact and convenient to describe it as a signature for ℓ simultaneous messages.

RESCHNORR SIGNATURE SCHEME. Similar to the Schnorr signature scheme, Reschnorr is a tuple of algorithms (Setup, Gen, Sign, and Verify), which we define as follows:

— $\mathsf{Setup}(1^\lambda)$: Generate ℓ primes $q_1, \ldots, q_\ell$ of size $\geq 2^\lambda$ and ℓ groups $\mathbb{G}_1, \ldots, \mathbb{G}_\ell$ respectively of order $q_1, \ldots q_\ell$ such that the DLP is hard in the respective $\mathbb{G}_i$, and such that $p = 1 + 2\prod q_i$ is prime. This is easily achieved by selecting $(\ell - 1)$ primes q_i and varying the last one until p is prime. Choose a cryptographic hash function $H : \{0,1\}^* \to \{0,1\}^{q_1}$. The hash function will be used to produce elements of $\mathbb{Z}_{q_i}$. For this we will denote by H_i the composition of H and a conversion function from $\{0,1\}^{q_1}$ to $\mathbb{Z}_{q_i}$ [2] Finally, choose g a generator of the group $\mathbb{Z}_p^*$ of order $p - 1$. The public parameters are therefore

$$\mathsf{pp} = \left(p, \{q_i\}_{i=1}^{\ell}, H, g, \{G_i\}_{i=1}^{\ell}\right).$$

— $\mathsf{Gen}(\mathsf{pp})$: The signer chooses $x \twoheadleftarrow \mathbb{Z}_{p-1}^*$ and computes $y \leftarrow g^x \bmod p$. The key $\mathsf{sk} = x$ is kept private to the signer, while the verification key $\mathsf{pk} = y$ is made public.

2. This conversion function can read the string as a binary number and reduce it mod q_i for example.

— $\mathsf{Sign}(\mathsf{pp}, \mathsf{sk}, m_1, \ldots, m_\ell)$: The signer chooses $k \twoheadleftarrow \mathbb{Z}_p$, such that $k \neq 0 \bmod q_i$ for all i, and computes $r \leftarrow g^k \bmod p$.
The signer can now sign the ℓ messages m_i as:

$$\rho_i \twoheadleftarrow \{0,1\}^\lambda, \quad e_i \leftarrow H_i(m_i, r, \rho_i), \quad \text{and} \quad s_i \leftarrow k - e_i x \bmod q_i$$

outputting the ℓ signatures $\sigma_i = \{r, s_i, \rho_i\}$—or, in a more compact form,

$$\sigma = \{r, s_1, \ldots, s_\ell, \rho_1, \ldots, \rho_\ell\}.$$

— $\mathsf{Verify}(\mathsf{pp}, \mathsf{pk}, m_i, (r, s_i, \rho_i), i)$: Verifying a signature is achieved by slightly modifying the original Schnorr scheme: First check that $s_i \in \{0, \ldots q_i - 1\}$ and compute $e_i \leftarrow H_i(m_i, r, \rho_i)$, then observe that for a correct signature [3]:

$$(g^{s_i} y^{e_i})^{\frac{p-1}{q_i}} = r^{\frac{p-1}{q_i}} \bmod p.$$

The signature is valid if and only if this equality holds, otherwise the signature is invalid (see Section 9.3).

Remark. Note that unlike Schnorr, in the Sign algorithm we add a random ρ_i for a signature to make the argument of the hash function unpredictable. This will be useful for the proof of Theorem 9.1 in the ROM.

Remark. Note also that one almost recovers the original Schnorr construction for $\ell = 1$—the only differences being in the verification formula, where both sides are squared in our version, and the addition of a fresh random to hash.

ReSchnorr correctness. The ReSchnorr signature scheme is correct.

Proof. Let g, y, r, s_i, and ρ_i be as generated by the Gen and Sign algorithms for a given message m_i. We check that,

$$\left(\frac{(g^{s_i} y^{e_i})^{s_i}}{r}\right)^{\frac{p-1}{q_i}} = 1 \bmod p.$$

By the definition of s_i, there exists $\lambda \in \mathbb{Z}$ such that $g^{s_i} = g^{k - e_i x + \lambda q_i}$, hence

$$g^{s_i} y^{e_i} g^{-k} = g^{\lambda q_i} \bmod p.$$

Raising this to the power of $\frac{p-1}{q_i}$ we get $g^{\lambda(p-1)} = 1$ since the order the multiplicative group $\mathbb{Z}_p^*$ is $p - 1$. □

3. One can note, $\frac{p-1}{q_i} = 2q_1 \cdots q_{i-1} q_{i+1} \cdots q_\ell$.

RESCHNORR SECURITY.

To aid in the proof of security, we introduce the following problem which we call the partial discrete logarithm problem (PDLP). Intuitively it corresponds to solving a discrete logarithm problem in the subgroup of our choice.

PDLP. Let $\ell \geq 2$ be an integer, $q_1, \ldots, q_\ell$ distinct prime numbers and $q = q_1 \ldots q_\ell$. Let $\mathbb{G}$ be a group of order q and g a generator of $\mathbb{G}$. Given $g, q, q_1, \ldots, q_l$, and $y = g^x$, the *partial discrete logarithm problem* (PDLP) consists in finding $i \in [\ell]$ and $x_i \in \mathbb{Z}_{q_i}$ such that $x_i = x \bmod q_i$.

In our context, we are chiefly interested in a subgroup of order q of a multiplicative group of a finite field $\mathbb{Z}_p^*$, where q divides $p-1$—ideally, $q = (p-1)/2$. The best known algorithms to solve the PDLP are index-calculus based methods in $\mathbb{Z}_p^*$ and square-root algorithms in subgroups of prime order q_i for some $i \in [\ell]$. With p of bit-size 3072, $q = (p-1)/2$, $\ell = 12$ and $q_1, \ldots, q_\ell$ of bit-size 256, we conjecture that solving the PDLP requires about 2^{128} elementary operations. In Section 9.3.1, we provide security argument in the generic group model on the intractability of the PDLP for large enough prime numbers $q_1, \ldots, q_\ell$.

Theorem 9.1 (Existential unforgeability)**.** *ReSchnorr is provably* EUF-CMA*-secure assuming the hardness of solving the PDLP, in the ROM.*

To prove this result, we will exhibit a reduction from an efficient EUF-CMA forger to an efficient PDLP solver. To that end we first show a sequence of indistinguishability results between the output distributions of

— Our signature algorithm $\mathsf{Sign} = \mathsf{Sign}_0$ on user inputs.
— A modified algorithm Sign_1 (see Figure 9.2), where the hash of user inputs is replaced by a random value. This situation is computationally indistinguishable from the previous one in the ROM.
— A modified algorithm Sign_2 (see Figure 9.2), that has no access to the signing key x. The output distribution of this algorithm is identical to the output of Sign_1 (Theorem 9.2).

Then we use the forking lemma [PSoo; BNo6] to show that an efficient EUF-CMA-adversary against Sign_2 can be used to construct an efficient PDLP solver. Finally we leverage the above series of indistinguishably results to use an adversary against Sign_0. Let CRT (for Chinese Remainder Theorem) be the isomorphism that maps $\mathbb{Z}_{q_1} \times \cdots \times \mathbb{Z}_{q_\ell} \times \mathbb{Z}_2$ to $\mathbb{Z}_{p-1}$.

Theorem 9.2. *The output distributions of* Sign_1 *and* Sign_2 *are identical.*

Proof. This theorem builds on several intermediate results described in **??**. We denote δ the output distribution of Sign_1 and δ' the output distribution of Sign_2. The structure of the proof is the following:

$\underline{\mathsf{Sign}_1}$:
$\rho \twoheadleftarrow \{0,1\}^\lambda$
$k \twoheadleftarrow \mathbb{Z}_p \setminus \left(\bigcup_{i=1}^{\ell} \{q_i, 2q_i, \ldots, p-1\}\right)$
$r \leftarrow g^k \bmod p$
for $i = 1$ to ℓ:
    $e_i \twoheadleftarrow \mathbb{Z}_{q_i}$
    $s_i \leftarrow k - e_i x \bmod q_i$
    $\rho_i \twoheadleftarrow \{0,1\}^\lambda$
return $(r, e_1, \ldots, e_\ell, s_1, \ldots, s_\ell, \rho_1 \ldots, \rho_\ell)$

$\underline{\mathsf{Sign}_2}$:
for $i = 1$ to ℓ
    $e_i \twoheadleftarrow \mathbb{Z}_{q_i}$
    $s_i \twoheadleftarrow \mathbb{Z}_{q_i}$
    $\rho_i \twoheadleftarrow \{0,1\}^\lambda$
$a \twoheadleftarrow \{0,1\}$
$b \twoheadleftarrow \{0,1\}$
$S \leftarrow \mathsf{CRT}(s_1, \ldots, s_\ell, a)$
$E \leftarrow \mathsf{CRT}(e_1, \ldots, e_\ell, b)$
$r \leftarrow g^S y^E$
for $i = 1$ to ℓ
    if $r \neq 1 \bmod q_i$,
    Else abort
return
$(r, e_1, \ldots, e_\ell, s_1, \ldots, s_\ell, \rho_1 \ldots, \rho_\ell)$

Figure 9.2 – The algorithms used in Theorem 9.2, as part of the proof of Theorem 9.1 (ReSchnorr signatures are EUF-CMA).

Theorem 9.3 shows that the output of Sign_2 is a subset of the output of Sign_1.

Theorem 9.4 shows that in Sign_1 there is a unique random tape per output.

Theorem 9.5 shows that in Sign_2 there are exactly two random tapes per output.

Theorem 9.7 shows that there are twice as many random tapes possible for Sign_2 than for Sign_1

This demonstrates that by uniformly choosing the random tape, the resulting distributions for Sign_1 and Sign_2 are identical, which is the uniform distribution on the set of valid signatures.

Lemma 9.3. *Every tuple of δ' is a valid signature tuple. Therefore $\delta' \subseteq \delta$.*

Proof. Let $(r, e_1, \ldots, e_\ell, s_1, \ldots, s_\ell, \rho_1, \ldots, \rho_\ell) \in \delta'$. Let $i \in [\ell]$. By the Chinese Remainder Theorem we have:

$$S = s_i \bmod q_i \quad \text{and} \quad E = e_i \bmod q_i.$$

So there exists $\lambda, \mu \in \mathbb{Z}$ such that

$$S = s_i + \lambda q_i \quad \text{and} \quad E = e_i + \mu q_i.$$

Hence:

$$\begin{aligned} r^{\frac{p-1}{q_i}} &= \left(g^S y^E\right)^{\frac{p-1}{q_i}} \\ &= \left(g^{s_i + \lambda q_i} y^{e_i + \mu q_i}\right)^{\frac{p-1}{q_i}} \\ &= \left(g^{s_i} y^{e_i}\right)^{\frac{p-1}{q_i}} g^{\lambda(p-1)} y^{\mu(p-1)} \\ &= \left(g^{s_i} y^{e_i}\right)^{\frac{p-1}{q_i}} \end{aligned}$$

The last equality holds since the order of the multiplicative group $\mathbb{Z}_p^*$ is $p-1$, and this concludes the proof with the fact that $r \neq 1 \bmod q_i$. □

Lemma 9.4. *There is exactly one random tape upon which* Sign_1 *can run to yield each particular tuple of* δ.

of Theorem 9.4. Let $k, e_1, \ldots, e_\ell, \rho_1, \ldots, \rho_\ell$ and $k', e'_1, \ldots, e'_\ell, \rho'_1, \ldots, \rho'_\ell$ be random choices of δ that both yield $(r, e_1, \ldots, e_\ell, s_1, \ldots, s_\ell, \rho_1, \ldots, \rho_\ell)$. It is immediate that $e_i = e'_i$ and $\rho_i = \rho'_i$ for all $i \in [\ell]$. Also since $g^k = g^{k'}$, g is of order $p-1$ and since k and k' are in $[p]$ then $k = k'$. □

Lemma 9.5. *There are exactly two random tapes over* $k, \rho_1, \ldots, \rho_\ell, e_1, \ldots, e_\ell$ *that output each tuple of* δ'.

Proof. Let $e_1, \ldots, e_\ell, s_1, \ldots, s_\ell, a, b, \rho_1, \ldots, \rho_\ell$ and $e'_1, \ldots, e'_\ell, s'_1, \ldots, s'_\ell, a', b', \rho'_1, \ldots, \rho'_\ell$ be random choices that both give $(r, e_1, \ldots, e_\ell, s_1, \ldots, s_\ell, \rho_1, \ldots, \rho_\ell)$. It is immediate that $e_i = e'_i$, $s_i = s'_i$, and $\rho_i = \rho'_i$ for all $i \in [\ell]$. Let S, S', E, and E' be the corresponding CRT images. We have $g^S y^E = g^{S'} y^{E'}$, which is $g^{S+xE} = g^{S'+xE'}$, and $S + xE = S' + xE' \bmod (p-1)$. Since x is odd (it is invertible mod $p-1$), it follows that $S+E$ and $S'+E'$ have the same parity. Therefore $a+b = a'+b' \bmod 2$ and we have two choices: $a = b$, or $a = 1-b$, both of which are correct.

□

Lemma 9.6. $\#\left(\mathbb{Z}_p \setminus \left(\bigcup_{i=1}^{\ell} \{q_i, 2q_i, \ldots, p-1\}\right)\right) = 2\prod_{i=1}^{\ell}(q_i - 1)$.

Proof. The number of invertible elements $\bmod p$ is $\prod_{i=1}^{\ell}(q_i-1) \times (2-1)$ so the number of invertible mod q_i for all i (and not necessarily for 2) is $2\prod_{i=1}^{\ell}(q_i-1)$. This is exactly the cardinality of the set

$$\left(\mathbb{Z}_p \setminus \left(\bigcup_{i=1}^{\ell} \{q_i, 2q_i, \ldots, p-1\}\right)\right),$$

□

Lemma 9.7. *There are twice as many possible random choices in* δ' *as there are in* δ.

Proof. For the number of random choices in δ we use Theorem 9.6 to count the number of k and then count the number of e_i and get $2\prod_{i=1}^{\ell}(q_i-1) \times \prod_{i=1}^{\ell} q_i$. For δ', having $r \neq 1 \bmod q_i$ is equivalent to having $s_i \neq -e_i x$. Therefore it has the same number of random choices as a distribution picking the s_i from $\mathbb{Z}_{q_i} \setminus \{e_i x\}$ which is $\prod_{i=1}^{\ell} q_i \times \prod_{i=1}^{\ell}(q_i-1) \times 2 \times 2$. □

It follows from the above results that the two distributions are the same, i.e. the uniform distribution over the set of valid signatures. This concludes the proof of Theorem 9.2. □

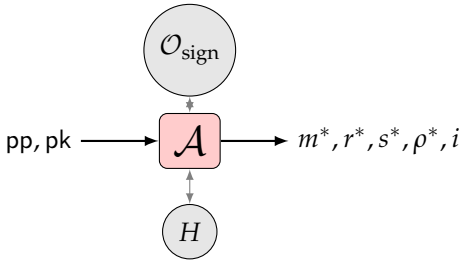


Figure 9.3 – An efficient EUF-CMA adversary $\mathcal{A}$ against ReSchnorr, with random oracle H and a signing oracle $\mathcal{O}$.

$\mathcal{R}$ implements the random oracle as $\mathcal{R}$.H and the signing oracle as $\mathcal{R}$.Sign. The rewinded adversary and oracles are indicated with a prime symbol.

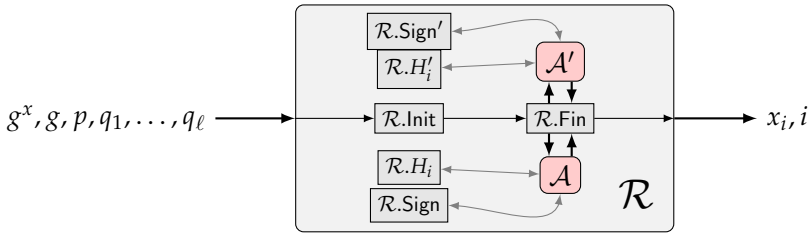


Figure 9.4 – An efficient solver $\mathcal{R}$ for the PDLP, using a polynomial number of queries to $\mathcal{A}$.

Theorem 9.8 (Security under Chosen Message Attack)**.** *An efficient attacker against* Sign_2 *can be turned into an efficient PDLP solver in the ROM.*

Proof. Let $\mathcal{A}$ be an attacker that wins the EUF-CMA game for ReSchnorr, illustrated in Figure 9.3. We construct in **??** an algorithm $\mathcal{R}$ that uses $\mathcal{A}$ to solve the PDLP. $\mathcal{A}'$ is equivalent to $\mathcal{A}$ (with the same random tape which we omit in the notation), the difference being that it interacts with different oracles. Abusing notation we denote by $\mathcal{R}.H_i$ the composition of the hash function and the conversion function. If L is a list of pairs, we denote by $L^{-1}[e]$ the index of the element e in the list, and by $L[i]$ the i-th element of the list. If they cannot (i.e. if e is not in the list, or the list does not have an i-th element) they abort.

The algorithm $\mathcal{R}$ aborts in four possible ways during the simulation (denoted $(\star)$, $(\dagger)$, $(\ddagger)$ and $(\S)$) in **??**. We upper-bound the probability of these events in the following list:

— $(\star)$ This occurs with negligible probability since the ρ is a fresh random which is unpredictable by the adversary.
— $(\dagger)$ This occurs with non overwhelming probability since the adversary is efficient.
— $(\ddagger)$ The element is in the list with non negligible probability because if the adversary forges on an unqueried hash in the ROM, it has a negligible chance to succeed.
— $(\S)$ This happens with non overwhelming probability due to the forking lemma [PSoo].

$\underline{\mathcal{R}.\mathsf{Init}(y = g^x, g, p, q_1, \ldots, q_\ell):}$
$\Sigma \leftarrow \emptyset$
$j \leftarrow 1$
$k \leftarrow 0$
$l \leftarrow 0$
$\mathsf{pk} \leftarrow y$
$\mathsf{pp} \leftarrow \{p, \{q_i\}_{i=1}^{\ell}, g\}$
return $(\mathsf{pk}, \mathsf{pp})$

$\underline{\mathcal{R}.\mathsf{Fin}(\mathsf{pk}, \mathsf{pp}):}$
$(M^*, r^*, s^*, \rho^*, i^*) \twoheadleftarrow \mathcal{A}(\mathsf{pp}, \mathsf{pk})$
$e^* \leftarrow \mathcal{R}.H_{i^*}(M^*, r^* \bmod q_{i^*}, \rho^*)$
$a \leftarrow L^{-1}[((M^*, r^* \bmod q_{i^*}, \rho^*), e^*)]$‡
if not $\mathsf{Verify}_{\mathsf{pp},\mathsf{pk}}(M^*, r^*, s^*, i^*)$
  abort †
$(M'^*, r'^*, s'^*, \rho'^*, i'^*) \twoheadleftarrow \mathcal{A}'(\mathsf{pp}, \mathsf{pk})$
if $i^* \neq i'^*$ then abort §
if $r^* \neq r'^*$ then abort §
$e'^* \leftarrow \mathcal{R}.H_{i^*}(M'^*, r^* \bmod q_{i^*}, \rho'^*)$
if $e^* = e'^*$ then abort §
if not $\mathsf{Verify}_{\mathsf{pp},\mathsf{pk}}(M'^*, r^*, s'^*, i^*)$
  abort †
$\Delta s \leftarrow s^* - s'^*$
$\Delta e \leftarrow e'^* - e^*$
return $(i^*, \Delta s / \Delta e)$

$\underline{\mathcal{R}.H(x):}$
if $\exists (x', h') \in \mathsf{L}$ s.t. $x' = x$
  return h'
Else
  $h \twoheadleftarrow \mathbb{Z}_p$
  $\mathsf{L} \leftarrow \mathsf{L} \cup \{(x, h)\}$
  return h

$\underline{\mathcal{R}.H'(x):}$
$k \leftarrow 0$
if $\exists (x', h') \in L'$ s.t. $x' = x$
  return h'
else
if $i \leq a$
  $(x', h') \leftarrow L.[i]$
  returnh'
  $k \leftarrow k + 1$
  $\mathsf{L}' \leftarrow \mathsf{L}' \cup \{(x, h)\}$
else
  $h \twoheadleftarrow \mathbb{Z}_p$
  $\mathsf{L}' \leftarrow \mathsf{L}' \cup \{(x, h)\}$
  h

$\underline{\mathcal{R}.\mathsf{Sign}'(M):}$
$l \leftarrow 0$
return $[i]$
$l \leftarrow l + 1$

$\underline{\mathcal{R}.\mathsf{Sign}(M):}$
if $j = 1$
  $(r, e_1, \ldots, e_\ell, s_1, \ldots, s_\ell, \rho_1, \ldots, \rho_\ell) \twoheadleftarrow \delta'$
  if $\exists h$ s.t. $((M, r \bmod q_1, \rho_1), h) \in \mathsf{L}$
    abort $\star$
  $\mathsf{L} \leftarrow \mathsf{L} \cup \{((M, r \bmod q_1, \rho_1), e_1)\}$
  $j \leftarrow j + 1 \bmod \ell$
  return $(s_1, r, \rho_1, 1)$
  $\Sigma \leftarrow \Sigma \cup \{(s_1, r, \rho_1, 1)\}$
else
  if $\exists h$ s.t. $((M, r \bmod q_j, \rho_j), h) \in \mathsf{L}$
    abort $\star$
  $\mathsf{L} \leftarrow \mathsf{L} \cup \{(M, r \bmod q_j, \rho_j), e_j\}$
  $j \leftarrow j + 1 \bmod \ell$
  return (s_j, r, ρ_j, j)
  $\Sigma \leftarrow \Sigma \cup \{(s_j, r, \rho_j, j)\}$

Figure 9.5 – An efficient solver for the PDLP, constructed from an efficient EUF-CMA adversary against ReSchnorr.

If $\mathcal{R}$ does not abort, then $\left(g^{s^*} y^{e^*}\right)^{\frac{p-1}{q_{i^*}}} = \left(r^*\right)^{\frac{p-1}{q_{i^*}}} = \left(g^{\tilde{s}^*} y^{\tilde{e}^*}\right)^{\frac{p-1}{q_{i^*}}} \bmod p$. Then $s^* + e^* x = \tilde{s}^* + \tilde{e}^* \bmod q_{i^*}$. It follows that the value returned by $\mathcal{R}$ is equal to $x \bmod q_{i^*}$.
$\mathcal{R}$ succeeds with non negligible probability, as explained earlier. The probability of forking is polynomial in the number of queries to the random oracle, the number of queries to the signature oracle, and ℓ. Note that the reduction is ℓ times looser than [PSoo]. This concludes the proof of Theorem 9.8. □

of Theorem 9.1. Using Theorem 9.2, we can use Sign_0 instead of Sign_2 as a target for the attacker in Theorem 9.8. □

9.3.1 *Generic Security of the Partial Discrete Logarithm Problem*

In this section, we prove that the partial discrete logarithm problem introduced in Section 9.3 is intractable in the generic group model. This model was introduced by Shoup [Sho97] for measuring the exact difficulty of solving classical discrete logarithm problems. Algorithms in generic groups do not exploit any properties of the encodings of group elements. They can access group elements only via a random encoding algorithm that encodes group elements as random bit-strings.

Proofs in the generic group model provide heuristic evidence of some problem hardness when an attacker does not take advantage of group elements' encoding. However, they do not necessarily say anything about the difficulty of specific problems in a concrete group.

Let ℓ be some non-negative integers, let $q_1, \ldots, q_\ell$ be some distinct prime numbers and let $q = q_1 \cdots q_\ell$. We consider a cyclic group $\mathbb{G}$ of (composite) order q generated by g. We assume without loss of generality that $q_1 = \max(q_1, \ldots, q_\ell)$. A classical method [PH78] to solve the partial discrete logarithm problem in $\mathbb{G}$ given $h = g^x \in \mathbb{G}$ is to compute $h^{q_2 \cdots q_\ell}$, an element of order diving q_1 (that belongs to the subgroup generated by $g^{q_2 \cdots q_\ell}$) and to compute its discrete logarithm x_1 in base $g^{q_2 \cdots q_\ell}$ using a square root method such as Shanks "baby-step giant-step" algorithm [Sha71]. It is easy to see that x_1 is equal to $x \bmod q_1$ and is obtained within time complexity $O(\sqrt{q_1} + \log(q_2 \cdots q_\ell))$ group operations.

Our goal is to prove that this time complexity is essentially optimal in the generic group model. Let $\mathcal{A}$ be a generic group adversary that solves the partial discrete logarithm problem in $\mathbb{G}$. As usual, the generic group model is implemented by choosing a random encoding $\sigma : \mathbb{G} \longrightarrow \{0,1\}^m$. Instead of working directly with group elements, $\mathcal{A}$ takes as input their image under σ. This way, all $\mathcal{A}$ can test is string equality. $\mathcal{A}$ is also given access to an oracle computing group multiplication and division: taking $\sigma(g_1)$ and $\sigma(g_2)$ and returning $\sigma(g_1 \cdot g_2)$ and $\sigma(g_1/g_2)$ respectively. Finally, we can assume that $\mathcal{A}$ submits to the oracle only encodings of elements it had previously received. This is because we can choose m large enough so that the probability of choosing a string that is also in the image of σ is negligible.

Theorem 9.9. *Let $\mathcal{A}$ be a generic algorithm that takes as input two encodings $\sigma(g)$ and $\sigma(h)$ (where g is a generator of $\mathbb{G}$ and $h = g^x \in \mathbb{G}$) and makes at most τ group oracle queries, then $\mathcal{A}$'s advantage in outputting a partial discrete logarithm (i, x_i) with $i \in \{1, \ldots, \ell\}$ and $x_i = x \bmod q_i$ is upper-bounded by $O(\tau^2/q_1)$.*

Proof. We consider an algorithm $\mathcal{B}$ playing the following game with $\mathcal{A}$. Algorithm $\mathcal{B}$ picks two bit strings σ_1, σ_2 uniformly at random in $\{0,1\}^m$. Internally, $\mathcal{B}$ keeps track of the encoded elements using elements in the ring $\mathbb{Z}_{q_1}[X_1] \times \cdots \times \mathbb{Z}_{q_\ell}[X_\ell]$. To maintain consistency with the bit strings given to $\mathcal{A}$, $\mathcal{B}$ creates a lists $\mathcal{L}$ of pairs (F, σ) where F is a polynomial vector in the ring $\mathbb{Z}_{q_1}[X_1] \times \cdots \times \mathbb{Z}_{q_\ell}[X_\ell]$ and $\sigma \in \{0,1\}^m$ is the encoding of a group element. The polynomial vector F represents the exponent of the encoded element in the group $\mathbb{Z}_{q_1} \times \cdots \times \mathbb{Z}_{q_\ell}$. Initially, $\mathcal{L}$ is set to

$$\{((1, 1, \ldots, 1), \sigma_1), ((X_1, \ldots, X_n), \sigma_2)\}$$

Algorithm $\mathcal{B}$ starts the game providing $\mathcal{A}$ with σ_1 and σ_2. The simulation of the group operations oracle goes as follows:

GROUP OPERATION: Given two encodings σ_i and σ_j in $\mathcal{L}$, $\mathcal{B}$ recovers the corresponding vectors F_i and F_j and computes $F_i + F_j$ for multiplication (or $F_i - F_j$ for division) termwise. If $F_i + F_j$ (or $F_i - F_j$) is already in $\mathcal{L}$, $\mathcal{B}$ returns to $\mathcal{A}$ the corresponding bit string; otherwise it returns a uniform element $\sigma \xleftarrow{R} \{0,1\}^m$ and stores $(F_i + F_j, \sigma)$ (or $(F_i - F_j, \sigma)$) in $\mathcal{L}$.

After $\mathcal{A}$ queried the oracles, it outputs a pair $(i^*, x_i^*) \in \{1 \ldots, \ell\} \times \mathbb{Z}_{q_{i^*}}$ as a candidate for the partial discrete logarithm of h in base g. At this point, $\mathcal{B}$ chooses uniform random values $x_1, \ldots, x_n \in \mathbb{Z}_{q_1} \times \cdots \times \mathbb{Z}_{q_\ell}$. The algorithm $\mathcal{B}$ sets $X_i = x_i$ for $i \in \{1, \ldots, n\}$.

If the simulation provided by $\mathcal{B}$ is consistent, it reveals nothing about $(x_1, \ldots, x_\ell)$. This means that the probability of $\mathcal{A}$ guessing the correct value for $(i^*, x_i^*) \in \{1, \ldots, \ell\} \times \mathbb{Z}_{q_{i^*}}$ is $1/q_{i^*}$. The only way in which the simulation could be inconsistent is if, after we choose value for $x_1, \ldots, x_n$, two different polynomial vectors in $\mathcal{L}$ happen to produce the same value.

It remains to compute the probability of a collision happening due to a unlucky choice of values. In other words, we have to bound the probability that two distinct vectors F_i, F_j in $\mathcal{L}$ evaluate to the same value after the substitution, namely $F_i(x_1, \ldots, x_n) - F_j(x_1, \ldots, x_n) = 0$. This reduces to bound the probability of hitting a zero of $F_i - F_j$. By the simulation, this happens only if $F_i - F_j$ is a vector of polynomials where at least one coordinate — say the k-th — is a non-constant polynomial (and thus of degree one) denoted $(F_i - F_j)^{(k)}$.

Recall that the Schwartz-Zippel lemma says that, if F is a degree d polynomial in $\mathbb{Z}_{q_k}[X_k]$ and $S \subseteq \mathbb{Z}_{q_k}$ then

$$\Pr[F(x_k) = 0 \bmod q_k] \leq \frac{d}{|S|}$$

where x_k is chosen uniformly from S. Going back to our case, we obtain by applying the Schwartz-Zippel lemma :

$$\Pr[(F_i - F_j)^{(k)}(x_k) = 0 \in \mathbb{Z}_{q_k}] \leq 1/q_k \leq 1/q_1.$$

Therefore, the probability that the simulation provided by $\mathcal{B}$ is inconsistent is upper-bounded by $\tau(\tau - 1)/q_1$ (by the union bound) and the result follows. □

9.4 PROVABLY SECURE PRE-COMPUTATIONS

Often the bottleneck in implementations centers around modular exponentiation. In this section we briefly outline several proposed *pre-computation* techniques, as well as presenting in more detail two pre-computation schemes which were used in our implementation to compare timings between classical Schnorr and ReSchnorr.

9.4.1 *Brief overview*

The problem of computing modular exponentiations is well-known to implementers of both DLP-based and RSA-based cryptosystems. In the specific case that we want to compute $g^x \bmod p$, the following strategies have been proposed but their security is often heuristic:

— Use signed expansions (only applicable to groups where inversion is efficient);
— Use Frobenius expansions or the GLV/GLS method (only applicable to certain elliptic curves);
— Batch exponentiations together, as suggested by M'Raïhi and Naccache [MN96].

The above approaches work for arbitrary values of x. Alternatively, one may choose a particular value of x with certain properties which make computation faster; however there is a possibility that doing so weakens the DLP:

— Choose x with low Hamming weight as proposed by Agnew et al. [AMOV91];
— Choose x to be a random Frobenius expansion of low Hamming weight, as discussed by Galbraith [Gal12, Sec. 11.3];
— Choose x to be given by a random addition chain, as proposed by Schroeppel et al. [SOOS95];
— Choose x to be a product of low Hamming weight integers as suggested by Hoffstein and Silverman [HS03]—broken by Cheon and Kim [CK08];
— Choose x to be a small random element in GLV representation—broken by Aranha et al. [AFGKTZ14];

Finally, a third branch of research uses large amounts of pre-computation to generate random pairs $(x, g^x \bmod p)$. The first effort in this direction was Schnorr's [Sch90], quickly broken by de Rooij [de 97]. Other constructions are due to Brickell et al. [BGMW93], Lim and Lee [LL94], and de Rooij [de 95]. The first provably secure solution is due to Boyko et al. [BPV98], henceforth BPV, which was extended and made more precise by [NSS01; CMT01; NS99]. This refined algorithm is called E-BPV (extended BPV).

9.4.2 *The E-BPV Pre-Computation Scheme*

E-BPV [4] relies on pre-computing and storing a set of pairs $(k_i, g^{k_i} \bmod p)$; then a "random" pair $(r, g^r \bmod p)$ is generated by choosing a subset of the k_i, setting r to be their sum, and computing the corresponding exponential by multiplying the $g^{k_i} \bmod p$.

4. BPV is a special case of E-BPV where $h = 2$. As such they share the same precomputing step.

(E-)BPV.Preprocessing:
$k_1, \ldots, k_n \twoheadleftarrow \mathbb{Z}_p^*$
for $j \in [n]$
$\mathsf{L} \leftarrow \mathsf{L} \cup \{(k_j, K_j = g^{k_j} \bmod p)\}$
return L

E-BPV.GetRandomPair:
pick $S \subseteq [n]$ s.t. $|S| = k$
$(d_i, D_i) \twoheadleftarrow D$
$r \leftarrow 0$
$R \leftarrow 1$
for $j \in S$
$x_j \twoheadleftarrow [h-1]$
$r \leftarrow r + k_j x_j \bmod \phi(p)$
$R \leftarrow R \cdot K_j^{x_j} \bmod p$
return (r, R)

Figure 9.6 – The E-BPV algorithm for generating random pairs $(x, g^x \bmod p)$. The BPV algorithm is a special case of E-BPV for $h = 2$.

To guarantee an acceptable level of security, and resist lattice reduction attacks, the number n of precomputed pairs must be sufficiently large; and enough pairs must be used to generate a new couple.

Nguyen et al. [NSS01] showed that using E-BPV instead of standard exponentiation gives an adversary an advantage bounded by

$$m\sqrt{\frac{K}{\binom{n}{k}(h-1)^k}}$$

with m the number of signature queries by the adversary, (k, n, h) E-BPV parameters, and K the exponent's size. [5]

We fix conservatively $m = 2^{128}$. For ReSchnorr, at 128-bit security, we have $K = P = 3072$. As suggested in [NSS01] we set $n = k$, and constrain our memory:

$$h^k \geq 2^{3400}$$

Optimizing $2k + h$ under this constraint, we find $(h, k) = (176, 455)$. This corresponds to 1087 modular multiplications, i.e., an amortized cost of 90 multiplications per signature, for about 170 kB of storage.

Alternatively, we can satisfy the security constraints by setting $n = 2048$, $h = 100$, $k = 320$, which corresponds to about 770 kB of storage, giving an amortized cost of 62 modular multiplications per signature.

In the implementation (Section 9.5), we solve the constrained optimisation problem to find the best coefficients (i.e., the least number of multiplications) for a given memory capacity.

Remark. To achieve the claimed bounds on modular multiplications, one should not compute $R \leftarrow K_j^{x_j} \bmod p$ directly; rather, an efficient speedup due to Brickell et al. [BGMW93] (BGMW) may be used. To illustrate the importance of this remark, we also give timings for a "naive" implementation in Table 9.3.

5. For Schnorr, the exponent's size is Q; for ReSchnorr, it is P.

Halving storage cost The following idea can halve the amount of storage required for the couples (x, g^x): instead of drawing the values x at random, we draw a master secret s once, and compute $x_{i+1} \leftarrow g^{x_i} \oplus s$ (or, more generally/securely, a PRF with low complexity $x_{i+1} = \mathrm{PRF}_s(g^{x_i})$). Only s, x_0, and the values g^{x_i} need to be stored; instead of all the couples (x_i, g^{x_i}). This remark applies to both BPV and E-BPV.

9.4.3 *Lim and Lee Pre-Computation Scheme*

We also consider a variation on Lim and Lee's fast exponentiation algorithm [LL94]. Their scheme originally computes g^r for r known in advance, but it is easily adapted to the setting where r is constructed on the fly. The speed-up is only linear, however, which ultimately means we cannot expect a sizable advantage over Schnorr. Nevertheless, Lim and Lee's algorithm is less resource-intensive and can be used in situations where no secure E-BPV parameters can be found (e.g., in ultra-low memory settings).

The Lim-Lee scheme (LL) has two parameters, h and v. In the original LL algorithm, the exponent is known in advance, but it is easily modified to generate an exponent on the fly. Intuitively, it consists in splitting the exponent in a "blocks" of size h, and dividing further each block in b sub-blocks of size v. The number of modular multiplications (in the worst case) is $a + b - 2$, and we have to store $(2^h - 1)v$ pairs. The algorithms are given in Figure 9.7.

For a given amount of memory M, it is easy to solve the constrained optimization problem, and we find

$$h_{\mathrm{opt}} = \frac{1}{\ln(2)} \left(1 + W\left(\frac{1+M}{e}\right)\right)$$

where W is the Lambert function. For a memory M of 750 kB, this gives $h \approx 8.6$. The optimal parameters for integers are $h = 9$ and $v = 4$.[6]

Remark. For LL, Figure 9.4.2 on halving storage requirements does not apply, as x need not be stored.

A summary of the properties for the pre-computations techniques E-PBV and LL can be found in Table 9.1.

9.5 IMPLEMENTATION RESULTS

Reschnorr, using the algorithms described in **??**, has been implemented in C using the GMP library. In the interest of timing comparison we have also implemented the classical Schnorr scheme. The

6. In practice, it turns out that $h = v = 8$ performs slightly better, due to various implementation speed-ups possible in this situation

LimLee.Preprocessing(h, v):
$g_0 \leftarrow g$
for $i = 0$ to $h - 1$
  $g_i \leftarrow g_{i-1}^{2^a}$
for $i = 0$ to $2^h - 1$
  let $i = e_{h-1} \dots e_1$ in binary
  $g_{0,i} = g_{h-1}^{e_{h-1}} \cdots g_1^{e_1}$
for $i = 0$ to $2^h - 1$
  for $j = 0$ to $v - 1$
    $g_{j,i} \leftarrow g_{j-1,i}^{2^b}$
    $\mathsf{L} \leftarrow \mathsf{L} \cup \{g_{j,i}\}$
return L

LimLee.GetRandomPair:
$R \leftarrow 1$
$r \leftarrow 0$
for $i = b - 1$ to 0
  $R \leftarrow R^2$
  $r \leftarrow r + r$
for $j = v - 1$ to 0
  $r_{i,j} \twoheadleftarrow \{0, \dots, 2^h - 1\}$
  $R \leftarrow R \times g_{j,r_{i,j}}$
  $r \leftarrow r + r_{i,j}$
return (r, R)

Figure 9.7 – The LL algorithm for generating random pairs $(x, g^x \bmod p)$.

results for several scenarios are outlined in Table 9.2 (at 128-bit security) and Table 9.3 (at 192-bit security). Complete source code and timing framework are available upon request from the authors.

These experiments show that ReSchnorr is faster than Schnorr when at least 250 pairs (i.e., 750 kB at 128-bit security) have been precomputed. This effect is even more markedly visible at higher security levels: ReSchnorr benefits more, and more effectively, from the E-BPV+BGMW optimisation as compared to Schnorr. The importance of combining E-BPV and BGMW is also visible: E-BPV using naive exponentiation does not provide any speed-up.

Schnorr and ReSchnorr achieve identical performance when using Lim and Lee's optimisation, confirming the theoretical analysis. When less than 1 MB of memory is alloted, this is the better choice.

9.5.1 *Heuristic Security*

Several papers describe server-aided precomputation techniques (e.g., [KU16]), which perform exponentiations with the help of a (possibly untrusted) server, i.e., such techniques allow to outsource the computation of $g^x \bmod n$, with public g and n, without revealing x to the server.

Algorithm	Storage	Multiplications	Security
Square-and-multiply	0	$1.5 \log P$	Always
BPV [BPV98]	nP	$k - 1$	$m\sqrt{\frac{P}{\binom{n}{k}}} < 2^{-\lambda}$
E-BPV [NSS01]	nP	$2k + h - 3$	$m\sqrt{\frac{P}{\binom{n}{k}(h-1)^k}} < 2^{-\lambda}$
Lim and Lee [LL94]	$2^h \times v \times P$	$\frac{\log P}{h}(1 + \frac{1}{v}) - 3$	Always

Table 9.1 – Precomputation/online computation trade-offs for ReSchnorr.

Scheme	Storage	Precomp.	Time (per sig.)
Schnorr	–	–	6.14 ms
Schnorr + [NSS01]	170 kB	33 s	105 ms
Schnorr + [NSS01] + [BGMW93]	170 kB	33 s	2.80 ms
Schnorr + [NSS01] + [BGMW93]	750 kB	33 s	2.03 ms
Schnorr + [NSS01] + [BGMW93]	1 MB	34 s	2.00 ms
Schnorr + [NSS01] + [BGMW93]	2 MB	37 s	2.85 ms
Schnorr + [LL94]	165 kB	3 s	949 ns
Schnorr + [LL94]	750 kB	3 s	644 ns
Schnorr + [LL94]	958 kB	3 s	630 ns
Schnorr + [LL94]	1.91 MB	3 s	★ 472 ns
ReSchnorr	–	–	5.94 ms
ReSchnorr + [NSS01]	170 kB	33 s	9.2 ms
ReSchnorr + [NSS01] + [BGMW93]	170 kB	33 s	1.23 ms
ReSchnorr + [NSS01] + [BGMW93]	750 kB	33 s	426 ns
ReSchnorr + [NSS01] + [BGMW93]	1 MB	34 s	371 ns
ReSchnorr + [NSS01] + [BGMW93]	2 MB	37 s	★ 327 ns
ReSchnorr + [LL94]	165 kB	3 s	918 ns
ReSchnorr + [LL94]	750 kB	3 s	709 ns
ReSchnorr + [LL94]	958 kB	3 s	650 ns
ReSchnorr + [LL94]	1.91 MB	3 s	757 ns

Table 9.2 – Timing results for Schnorr and ReSchnorr, at 128-bit security ($P = 3072$, $Q = 256$). Computation was performed on an ArchLinux single-core 32-bit virtual machine with 128 MB RAM. Averaged over 256 runs.

Scheme	Storage	Time (/sig.)
Schnorr	–	35.2 ms
Schnorr + [LL94]	715 kB	508 ns
Schnorr + [NSS01] + [BGMW93]	750 kB	2.08 ms
Schnorr + [NSS01] + [BGMW93]	1.87 MB	1.62 ms
Schnorr + [LL94]	1.87 MB	★ 476 ns
ReSchnorr	–	33.0 ms
ReSchnorr + [LL94]	715 kB	486 ns
ReSchnorr + [LL94]	1.87 MB	467 ns
ReSchnorr + [NSS01] + [BGMW93]	1.87 MB	★ 263 ns

Table 9.3 – Timing results for Schnorr and ReSchnorr, at 192-bit security ($P = 7680$, $Q = 384$). Computation was performed on an ArchLinux single-core 32-bit virtual machine with 128 MB RAM. Averaged over 256 runs.

Interestingly, the most efficient algorithms in that scenario (which of course we could leverage) use parameters provided by Hohenberger and Lysyanskaya [HL05] for E-BPV. A series of papers took these

parameters for granted (including [KU16]), but we should point out that *these are not covered* by the security proof found in [NSS01].

Despite this remark, it seems that no practical attack is known either; therefore if we are willing to relax our security expectations somewhat it is possible to compute the modular exponentiation faster. Namely, a Q-bit exponent can be computed in $O(\log Q^2)$ modular multiplications.

ReSchnorr uses an exponent that is ℓ times bigger than Schnorr, which is amortized over ℓ signatures. Comparing ReSchnorr to Schnorr, the ratio is $\frac{\ell \log(Q)^2}{(\log \ell Q)^2}$. With $Q = 256$ we get a ratio of approximately 5.7.

Note that as Q increases, so does ℓ, and therefore so does the advantage of ReSchnorr over Schnorr in that regime.

9.5.2 *Reduction-Friendly Moduli*

As part of computing $g^k \bmod p$, a very costly operation is the reduction mod p. An interesting question is whether some particular moduli p can be found, for which reduction is particularly easy.

An example of such moduli are those that start with a `1` followed by many `0`.

Example 9.1. *For $P = 3072$ and $Q = 256$, using (in hexadecimal notation)*

$$\Delta_i = \{\texttt{12d}, \texttt{165}, \texttt{1e7}, \texttt{247}, \texttt{2f5}, \texttt{31b}, \texttt{327}, \texttt{34f}, \texttt{3a3}, \texttt{439}, \texttt{56b}, \texttt{4fe7}\}$$

and $q_i = 2^Q + \Delta_i$, we have that p equals:

```
2[60]e0e8[56]18058164[53]1479d1e16e8[51]aa09581f139be[48]3
a9dc2e99b080dd[47]dfe705c4e9b3a45678[43]25a378c4e6b62835f4
01[42]471d330fbde56ef2c80281e[39]5c5388621a308a5425f007648
[37]4e506ba1a5b68dc5faca1155e64[35]270051399124b193e6716e0
8b4408[34]8a07b85ed815e7eac1135861bd67e3
```

where `[x]` denotes a sequence of `x` hexadecimal zeros.

9.6 CONCLUSION

We have introduced a new digital signature scheme variant of Schnorr signatures, that reuses the nonce component for several signatures. Doing so does not jeopardise the scheme's security; attempting to do the same with classical Schnorr signatures would immediately reveal the signing key. However the main appeal of our approach is that precomputation techniques, whose benefits can only be seen for large enough problems, become applicable and interesting. As a result, without loss of security, it becomes possible to sign messages using fewer modular multiplications.

Part V

EPILOGUE

10 CONCLUSION

Let us continue our historical stroll before it comes time to conclude, for now, and put the pen down.

10.1 THE TEENAGE YEARS OF MODERN CRYPTOGRAPHY

The year is 2019. Since you started your journey, there have been even more shocks to the system. You have learned of Cambridge Analytica and their ilk, which made you become increasingly worried about the state of democracy. Digital advertising is the new religion, social networks the sacred spaces, and influencers the prophets. You see that choice, decision, information, only partially reside in the hands of the individual. You've seen the rise of voice-, and facial-recognition software fueling the surveillance state. You've seen this move so quickly, with monumental mistakes, and you worry, because it seems that there are very few who even notice. And you *still* can't send an encrypted email! It's chaos.

Yet, at the same time, you see digital currency and its supporters unanimously wanting anonymity. You see *messengers* gaining popularity and think maybe you don't actually *need* encrypted email. Maybe there are other ways to communicate. Maybe there are other ways to *compute*. You catch machine learning and cryptography's eyes meeting across the room. The lawyers and engineers start to hold hands. You're growing up! It's confusing now, and you're left with more questions than you ever thought possible, but don't worry. This is a period of change, a moment of growth. You will understand more when you're older, but for now, my dear, let's sit down, and think about where we should go from here.

I refuse to write the word 'crypto' pertaining to anything other than cryptography in this thesis.

10.2 OPEN PROBLEMS

10.2.1 *Key-Correlated Security*

One of the greatest lessons learned while working towards the idea of key-correlated attacks was that although related key attacks and key-dependent message attacks have been widely studied, and even the idea of key-dependent messages encrypted under related keys has been mentioned, there are a number of subtleties that went unnoticed for many years. Furthermore, although KCA does not seem very different to RKA + KDM, there were many interesting obstacles that needed to be overcome when trying to deduce the proofs. This work

The most notable obstacle was solved by the xkcd *property :)*

really displayed the richness that can be found in cryptography, and teaches the lesson that we should not content ourselves with following only intuition, more digging is often needed. As such, we named the part 'Try Again'. The advent of a new security notion can open a multitude of directions to follow. We aimed to address as many as possible in the work carried out in Part II, but there are still a number of questions to be addressed:

ANALYSIS OF FEISTEL CONSTRUCTION. In Chapter 5 we looked at the KCA-security of blockciphers and showed that 3-rounds of Even–Mansour are sufficient to achieve KCA-security for a class of functions. One could further extend this analysis by looking at other block cipher models. For example, how many rounds of Feistel are needed to ensure KCA-security.

ANALYSIS OF MODES OF OPERATION. Another direction would be to look at blockciphers in a broader context and ask whether any of the currently used modes of operation are secure under key-correlated attacks.

CONCRETE CONSTRUCTIONS. As well as looking at the broader context, it would be interesting to know if the notion of key-correlated attacks aids in the cryptanalysis of concrete constructions, for example, of AES.

KCX. A small, but technical detail arose while thinking about key-correlated security. Depending on the model, keys are XORed with functions in different ways. It would be interesting to extend the idea of key-correlated attacks to key-correlated XORing.

10.2.2 *New Assumptions and PKE Schemes*

Public-key encryption is fundamental in cryptography. It was noted several times that this is what caused the *revolution* in the 1970's. Today there remain very few PKE schemes, and the hunt to find more is still ongoing. Furthermore, as we move closer to having quantum computing as a reality, there is an urgency to *find* schemes that are resistant to quantum attacks. In 2017, [AJPS17c] described such an encryption scheme. Furthermore, it was based on the hardness of a new assumption. This is doubly exciting because not only do we (potentially) have a new PKE scheme, we also have an entirely new toy from which we can start to think about building other schemes. Two new developments such as this will garner much attention from those wanting to try to break the new toys, but also from those who want to design new things. The work in Chapter 7 aimed at the former and tried to break the scheme. Given the parameters suggested in the original work, we managed to find the secret key, given only the public key, in only a number of days. The work in Chapter 8 aimed at the latter and proposed a modification to the assumption and described

another public-key encryption scheme. Both of these works were quite reactionary and so did not perform very formal analysis. More formal analysis was carried out in [BDJW18] and eventually the AJPS cryptosystem was modified and published as [AJPS18]. This was an exciting time to work in the area and one of the main lessons learned was that even with careful consideration of cryptographic schemes, there can exist small 'tricks' that allow to exploit a particular feature. It was a nice reminder to a young researcher that although sometimes you may strike gold, more often than not, and over and over, you are going to *'fail again'*, and hence the title for this part. In any case, these initial analyses barely scratch the surface of the research opened up in this space, and there are a number of areas left to be addressed:

MORE EFFICIENT ATTACKS. The AJPS encryption scheme is not entirely dissimilar to NTRU constructions, so it would be interesting to see if more of the analysis performed on NTRU also applies to AJPS. In particular whether Lattice attacks and the Meet-in-the-Middle attacks can be combined to gain efficiency, as was done for NTRU. This is a question asked by Adi Shamir following the presentation of [BCGN17a].

QUANTUM ATTACKS. The scheme is supposed to be post-quantum secure. The analysis in [BDJW18] looked at methods based on Grover's algorithm, but leave open the question of whether less generic approaches would work better.

OTHER SCHEMES. Given a new hardness assumption, what other primitives can be build from it? As we saw with the DL and RSA problems, can we build signature schemes to provide integrity with such an assumption?

10.2.3 *Efficient Signature Schemes*

We saw with key-correlated security that working in very general models can lead to some security oversights. One of the most interesting points working towards the results in Part IV is that when we come out of the very general model and start to look at the specific construction, there can also be room for improvement. By taking advantage of the group structure and by choosing our primes in a particular way, we can gain efficiency. The greatest lesson learned here is that it is important to have a global view of how a primitive works, but there's also much to be gained by looking at the specifics and hence the naming of this part *'fail better'*.

Ok, there may have been some artistic license in the naming of these parts...

CAN WE GENERALISE? 'Reusing nonces' as we did with Schnorr signatures can be thought of as somewhat similar to related key notions of security. An interesting question would be to ask whether we can retain some notion of security for Schnorr-like

signature schemes where all inputs are related, or derived, or correlated.

MORE EXPLOITATION Are there other schemes in which we can look at the details, group structure or otherwise, and gain efficiency, or stronger security?

10.3 BEYOND THE TECHNICALITIES

There are so many big ideas, tiny intricacies, and alluring notions, that now, coming to the end of this thesis, I feel that I am only *beginning* to discover the beauty of this field. There are a number of changes that the field is experiencing at the moment. Cryptography is moving beyond the "all or nothing" paradigm it used to be known for; it is no longer the case that an adversary can learn nothing about a message. Actually, maybe to gain functionality, we're happy for them to learn a very controlled *something*. We're seeing the rise of homomorphic and functional encryption, of MPC, and all the ideas that allow us to compute on encrypted data. This opens a can of worms so that now, all of a sudden, we can 'peek' inside the ciphertext. There are huge implications of this for all manner of data analytics and machine learning. It begs the question of *who* do we allow to look at our ciphertexts? How can we enforce this?

And what of quantum computers? How will they affect us? Are we ready? Or even closer still, 5*G* there are attacks, right? But they're still going ahead with it? What about these digital currencies? Will they last? How about the community? Even this is changing. We're no longer just academics or military personnel. We're lawyers, engineers, social scientists, managers. There's a lot to look forward to, and still many questions to answer.

A FINAL REMARK. I have often been told that a popular question asked of PhD students when talking about a piece of scientific work, or research, is *"How do you know when you're done?"* I have thought about this a lot over the years and, being a PhD student, posed my answers as questions:

I quickly learned the first one is a very wrong answer

— "I'm done when.. I think I'm done?"
— "When my advisor tells me I'm done?"
— "When reviewer n° 2 has run out of *stuff* to complain about?"
— "When it's 5:59 am on a Saturday morning and there's a deadline in 1 minute?"
— "When the coffee machine breaks? Or when my laptop dies?"
— "When *am* I done?!"

If there is one thing I have learned throughout the course of this PhD, it's a definite answer to that question. Open your mind enough to an idea, and you'll *never* be done; for the things you truly think about, reward you with a lifetime of problems to solve, questions to ask, answers to twist, papers to write, books to read, students to te...

BIBLIOGRAPHY

[AJPS17a] Divesh Aggarwal, Antoine Joux, Anupam Prakash, and Miklos Santha. *A New Public-Key Cryptosystem via Mersenne Numbers, version 20170530:001542*. Cryptology ePrint Archive, Report 2017/481. https://eprint.iacr.org/2017/481. 2017.

[AJPS17b] Divesh Aggarwal, Antoine Joux, Anupam Prakash, and Miklos Santha. *A New Public-Key Cryptosystem via Mersenne Numbers, version 20171206:004924*. Cryptology ePrint Archive, Report 2017/481. https://eprint.iacr.org/2017/481. 2017.

[AJPS17c] Divesh Aggarwal, Antoine Joux, Anupam Prakash, and Miklos Santha. *A New Public-Key Cryptosystem via Mersenne Numbers*. Cryptology ePrint Archive, Report 2017/481. http://eprint.iacr.org/2017/481. 2017.

[AJPS17d] Divesh Aggarwal, Antoine Joux, Anupam Prakash, and Miklos Santha. *A New Public-Key Cryptosystem via Mersenne Numbers*. Cryptology ePrint Archive, Report 2017/481. http://eprint.iacr.org/2017/481. 2017.

[AJPS18] Divesh Aggarwal, Antoine Joux, Anupam Prakash, and Miklos Santha. "A New Public-Key Cryptosystem via Mersenne Numbers." In: *CRYPTO 2018, Part III*. Ed. by Hovav Shacham and Alexandra Boldyreva. Vol. 10993. LNCS. Springer, Heidelberg, Aug. 2018, pp. 459–482. DOI: 10.1007/978-3-319-96878-0_16.

[AMOV91] Gordon B. Agnew, Ronald C. Mullin, I. M. Onyszchuk, and Scott A. Vanstone. "An Implementation for a Fast Public-Key Cryptosystem." In: *Journal of Cryptology* 3.2 (Jan. 1991), pp. 63–79. DOI: 10.1007/BF00196789.

[AFPW11] Martin R. Albrecht, Pooya Farshim, Kenneth G. Paterson, and Gaven J. Watson. "On Cipher-Dependent Related-Key Attacks in the Ideal-Cipher Model." In: *FSE 2011*. Ed. by Antoine Joux. Vol. 6733. LNCS. Springer, Heidelberg, Feb. 2011, pp. 128–145. DOI: 10.1007/978-3-642-21702-9_8.

[App16] Benny Applebaum. "Garbling XOR Gates "For Free" in the Standard Model." In: *Journal of Cryptology* 29.3 (July 2016), pp. 552–576. DOI: 10.1007/s00145-015-9201-9.

[AFGKTZ14] Diego F. Aranha, Pierre-Alain Fouque, Benoît Gérard, Jean-Gabriel Kammerer, Mehdi Tibouchi, and Jean-Christophe Zapalowicz. "GLV/GLS Decomposition, Power Analysis, and Attacks on ECDSA Signatures with Single-Bit Nonce Bias." In: *ASIACRYPT 2014, Part I*. Ed. by Palash Sarkar and Tetsu Iwata. Vol. 8873. LNCS. Springer, Heidelberg, Dec. 2014, pp. 262–281. DOI: 10.1007/978-3-662-45611-8_14.

[BF15] Manuel Barbosa and Pooya Farshim. "The Related-Key Analysis of Feistel Constructions." In: *FSE 2014*. Ed. by Carlos Cid and Christian Rechberger. Vol. 8540. LNCS. Springer, Heidelberg, Mar. 2015, pp. 265–284. DOI: 10.1007/978-3-662-46706-0_14.

[BCK11] Mihir Bellare, David Cash, and Sriram Keelveedhi. "Ciphers that securely encipher their own keys." In: *ACM CCS 2011*. Ed. by Yan Chen, George Danezis, and Vitaly Shmatikov. ACM Press, Oct. 2011, pp. 423–432. DOI: 10.1145/2046707.2046757.

[BCM11] Mihir Bellare, David Cash, and Rachel Miller. "Cryptography Secure against Related-Key Attacks and Tampering." In: *ASIACRYPT 2011*. Ed. by Dong Hoon Lee and Xiaoyun Wang. Vol. 7073. LNCS. Springer, Heidelberg, Dec. 2011, pp. 486–503. DOI: 10.1007/978-3-642-25385-0_26.

[BDJR97] Mihir Bellare, Anand Desai, Eric Jokipii, and Phillip Rogaway. "A Concrete Security Treatment of Symmetric Encryption." In: *38th FOCS*. IEEE Computer Society Press, Oct. 1997, pp. 394–403. DOI: 10.1109/SFCS.1997.646128.

[BK11] Mihir Bellare and Sriram Keelveedhi. "Authenticated and Misuse-Resistant Encryption of Key-Dependent Data." In: *CRYPTO 2011*. Ed. by Phillip Rogaway. Vol. 6841. LNCS. Springer, Heidelberg, Aug. 2011, pp. 610–629. DOI: 10.1007/978-3-642-22792-9_35.

[BK03] Mihir Bellare and Tadayoshi Kohno. "A Theoretical Treatment of Related-Key Attacks: RKA-PRPs, RKA-PRFs, and Applications." In: *EUROCRYPT 2003*. Ed. by Eli Biham. Vol. 2656. LNCS. Springer, Heidelberg, May 2003, pp. 491–506. DOI: 10.1007/3-540-39200-9_31.

[BN06] Mihir Bellare and Gregory Neven. "Multi-signatures in the plain public-Key model and a general forking lemma." In: *ACM CCS 2006*. Ed. by Ari Juels, Rebecca N. Wright, and Sabrina De Capitani di Vimer-

cati. ACM Press, 2006, pp. 390–399. DOI: 10.1145/1180405.1180453.

[BR93] Mihir Bellare and Phillip Rogaway. "Random Oracles are Practical: A Paradigm for Designing Efficient Protocols." In: *ACM CCS 93*. Ed. by Dorothy E. Denning, Raymond Pyle, Ravi Ganesan, Ravi S. Sandhu, and Victoria Ashby. ACM Press, Nov. 1993, pp. 62–73. DOI: 10.1145/168588.168596.

[BR06] Mihir Bellare and Phillip Rogaway. "The Security of Triple Encryption and a Framework for Code-Based Game-Playing Proofs." In: *EUROCRYPT 2006*. Ed. by Serge Vaudenay. Vol. 4004. LNCS. Springer, Heidelberg, 2006, pp. 409–426. DOI: 10.1007/11761679_25.

[BCLV16] Daniel J. Bernstein, Chitchanok Chuengsatiansup, Tanja Lange, and Christine van Vredendaal. "NTRU Prime." In: *IACR Cryptology ePrint Archive* 2016 (2016), p. 461. URL: http://eprint.iacr.org/2016/461.

[BCGN17a] Marc Beunardeau, Aisling Connolly, Rémi Géraud, and David Naccache. "On the Hardness of the Mersenne Low Hamming Ratio Assumption." In: *Progress in Cryptology - LATINCRYPT 2017 - 5th International Conference on Cryptology and Information Security in Latin America, Havana, Cuba, September 20-22, 2017, Revised Selected Papers*. 2017, pp. 166–174. DOI: 10.1007/978-3-030-25283-0_9. URL: https://doi.org/10.1007/978-3-030-25283-0_9.

[BCGNV17] Marc Beunardeau, Aisling Connolly, Remi Geraud, David Naccache, and Damien Vergnaud. "Reusing nonces in Schnorr signatures." In: *Proceedings of the 22nd European Symposium on Research in Computer Security, ESORICS 2017, Oslo, Norway, September 11–15*. 2017.

[BCGN17b] Marc Beunardeau, Aisling Connolly, Rémi Géraud, and David Naccache. *On the Hardness of the Mersenne Low Hamming Ratio Assumption*. Cryptology ePrint Archive, Report 2017/522. https://eprint.iacr.org/2017/522. 2017.

[Bih94a] Eli Biham. "New Types of Cryptanalytic Attacks Using Related Keys." In: *Journal of Cryptology* 7.4 (Dec. 1994), pp. 229–246. DOI: 10.1007/BF00203965.

[Bih94b] Eli Biham. "New Types of Cryptanalytic Attacks Using related Keys (Extended Abstract)." In: *EUROCRYPT'93*. Ed. by Tor Helleseth. Vol. 765. LNCS.

Springer, Heidelberg, May 1994, pp. 398–409. DOI: 10.1007/3-540-48285-7_34.

[BS97] Eli Biham and Adi Shamir. "Differential Fault Analysis of Secret Key Cryptosystems." In: *CRYPTO'97*. Ed. by Burton S. Kaliski Jr. Vol. 1294. LNCS. Springer, Heidelberg, Aug. 1997, pp. 513–525. DOI: 10.1007/BFb0052259.

[BK09] Alex Biryukov and Dmitry Khovratovich. "Related-Key Cryptanalysis of the Full AES-192 and AES-256." In: *ASIACRYPT 2009*. Ed. by Mitsuru Matsui. Vol. 5912. LNCS. Springer, Heidelberg, Dec. 2009, pp. 1–18. DOI: 10.1007/978-3-642-10366-7_1.

[BKN09] Alex Biryukov, Dmitry Khovratovich, and Ivica Nikolic. "Distinguisher and Related-Key Attack on the Full AES-256." In: *CRYPTO 2009*. Ed. by Shai Halevi. Vol. 5677. LNCS. Springer, Heidelberg, Aug. 2009, pp. 231–249. DOI: 10.1007/978-3-642-03356-8_14.

[BW99] Alex Biryukov and David Wagner. "Slide Attacks." In: *FSE'99*. Ed. by Lars R. Knudsen. Vol. 1636. LNCS. Springer, Heidelberg, Mar. 1999, pp. 245–259. DOI: 10.1007/3-540-48519-8_18.

[Bla06] John Black. "The Ideal-Cipher Model, Revisited: An Uninstantiable Blockcipher-Based Hash Function." In: *FSE 2006*. Ed. by Matthew J. B. Robshaw. Vol. 4047. LNCS. Springer, Heidelberg, Mar. 2006, pp. 328–340. DOI: 10.1007/11799313_21.

[BRS02] John Black, Phillip Rogaway, and Thomas Shrimpton. "Black-Box Analysis of the Block-Cipher-Based Hash-Function Constructions from PGV." In: *CRYPTO 2002*. Ed. by Moti Yung. Vol. 2442. LNCS. Springer, Heidelberg, Aug. 2002, pp. 320–335. DOI: 10.1007/3-540-45708-9_21.

[BRS03] John Black, Phillip Rogaway, and Thomas Shrimpton. "Encryption-Scheme Security in the Presence of Key-Dependent Messages." In: *SAC 2002*. Ed. by Kaisa Nyberg and Howard M. Heys. Vol. 2595. LNCS. Springer, Heidelberg, Aug. 2003, pp. 62–75. DOI: 10.1007/3-540-36492-7_6.

[BDJW18] Koen de Boer, Léo Ducas, Stacey Jeffery, and Ronald de Wolf. "Attacks on the AJPS Mersenne-Based Cryptosystem." In: *Post-Quantum Cryptography - 9th International Conference, PQCrypto 2018*. Ed. by Tanja Lange and Rainer Steinwandt. Springer, Heidelberg,

2018, pp. 101–120. DOI: 10.1007/978-3-319-79063-3_5.

[BDJW17] Koen de Boer, Léo Ducas, Stacey Jeffery, and Ronald de Wolf. *Attacks on the AJPS Mersenne-based cryptosystem*. Cryptology ePrint Archive, Report 2017/1171. https://eprint.iacr.org/2017/1171. 2017.

[BDH14] Florian Böhl, Gareth T. Davies, and Dennis Hofheinz. "Encryption Schemes Secure under Related-Key and Key-Dependent Message Attacks." In: *PKC 2014*. Ed. by Hugo Krawczyk. Vol. 8383. LNCS. Springer, Heidelberg, Mar. 2014, pp. 483–500. DOI: 10.1007/978-3-642-54631-0_28.

[BDL97] Dan Boneh, Richard A. DeMillo, and Richard J. Lipton. "On the Importance of Checking Cryptographic Protocols for Faults (Extended Abstract)." In: *EUROCRYPT'97*. Ed. by Walter Fumy. Vol. 1233. LNCS. Springer, Heidelberg, May 1997, pp. 37–51. DOI: 10.1007/3-540-69053-0_4.

[BHHO08] Dan Boneh, Shai Halevi, Michael Hamburg, and Rafail Ostrovsky. "Circular-Secure Encryption from Decision Diffie-Hellman." In: *CRYPTO 2008*. Ed. by David Wagner. Vol. 5157. LNCS. Springer, Heidelberg, Aug. 2008, pp. 108–125. DOI: 10.1007/978-3-540-85174-5_7.

[BV98] Dan Boneh and Ramarathnam Venkatesan. "Breaking RSA May Not Be Equivalent to Factoring." In: *EUROCRYPT'98*. Ed. by Kaisa Nyberg. Vol. 1403. LNCS. Springer, Heidelberg, 1998, pp. 59–71. DOI: 10.1007/BFb0054117.

[BRC60] Raj Chandra Bose and Dwijendra K. Ray-Chaudhuri. "On a class of error correcting binary group codes." In: *Information and control* 3.1 (1960), pp. 68–79.

[BDMN16] Lilian Bossuet, Nilanjan Datta, Cuauhtemoc Mancillas-López, and Mridul Nandi. "ELmD: A Pipelineable Authenticated Encryption and Its Hardware Implementation." In: *IEEE Trans. Computers* 65.11 (2016), pp. 3318–3331. DOI: 10.1109/TC.2016.2529618. URL: https://doi.org/10.1109/TC.2016.2529618.

[BPV98] Victor Boyko, Marcus Peinado, and Ramarathnam Venkatesan. "Speeding up Discrete Log and Factoring Based Schemes via Precomputations." In: *EUROCRYPT'98*. Ed. by Kaisa Nyberg. Vol. 1403. LNCS. Springer, Heidelberg, 1998, pp. 221–235. DOI: 10.1007/BFb0054129.

[BGMW93] Ernest F. Brickell, Daniel M. Gordon, Kevin S. McCurley, and David Bruce Wilson. "Fast Exponentiation with Precomputation (Extended Abstract)." In: *EUROCRYPT'92*. Ed. by Rainer A. Rueppel. Vol. 658. LNCS. Springer, Heidelberg, May 1993, pp. 200–207. DOI: 10.1007/3-540-47555-9_18.

[Bro16] Daniel R. L. Brown. "Breaking RSA May Be As Difficult As Factoring." In: *Journal of Cryptology* 29.1 (Jan. 2016), pp. 220–241. DOI: 10.1007/s00145-014-9192-y.

[Buh98] Joe Buhler, ed. *Algorithmic Number Theory, Third International Symposium, ANTS-III, Portland, Oregon, USA, June 21-25, 1998, Proceedings*. Vol. 1423. Lecture Notes in Computer Science. Springer, 1998. ISBN: 3-540-64657-4. DOI: 10.1007/BFb0054849. URL: https://doi.org/10.1007/BFb0054849.

[CCS09] Jan Camenisch, Nishanth Chandran, and Victor Shoup. "A Public Key Encryption Scheme Secure against Key Dependent Chosen Plaintext and Adaptive Chosen Ciphertext Attacks." In: *EUROCRYPT 2009*. Ed. by Antoine Joux. Vol. 5479. LNCS. Springer, Heidelberg, Apr. 2009, pp. 351–368. DOI: 10.1007/978-3-642-01001-9_20.

[CL01] Jan Camenisch and Anna Lysyanskaya. "An Identity Escrow Scheme with Appointed Verifiers." In: *CRYPTO 2001*. Ed. by Joe Kilian. Vol. 2139. LNCS. Springer, Heidelberg, Aug. 2001, pp. 388–407. DOI: 10.1007/3-540-44647-8_23.

[CGH98] Ran Canetti, Oded Goldreich, and Shai Halevi. "The Random Oracle Methodology, Revisited (Preliminary Version)." In: *30th ACM STOC*. ACM Press, May 1998, pp. 209–218. DOI: 10.1145/276698.276741.

[CK08] Jung Hee Cheon and HongTae Kim. "Analysis of Low Hamming Weight Products." In: *Discrete Applied Mathematics* 156.12 (2008), pp. 2264–2269.

[Cli96] William J. Clinton. "Administration of Export Controls on Encryption Products." In: *Executive Order 13026* (1996).

[CPS14] Benoit Cogliati, Jacques Patarin, and Yannick Seurin. "Security Amplification for the Composition of Block Ciphers: Simpler Proofs and New Results." In: *SAC 2014*. Ed. by Antoine Joux and Amr M. Youssef. Vol. 8781. LNCS. Springer, Heidelberg, Aug. 2014, pp. 129–146. DOI: 10.1007/978-3-319-13051-4_8.

[CS15] Benoit Cogliati and Yannick Seurin. "On the Provable Security of the Iterated Even-Mansour Cipher Against Related-Key and Chosen-Key Attacks." In: *EUROCRYPT 2015, Part I*. Ed. by Elisabeth Oswald and Marc Fischlin. Vol. 9056. LNCS. Springer, Heidelberg, Apr. 2015, pp. 584–613. DOI: 10.1007/978-3-662-46800-5_23.

[U.S. Const.] Constitution of the United States. 1787.

[CS97] Don Coppersmith and Adi Shamir. "Lattice Attacks on NTRU." In: *Advances in Cryptology - EUROCRYPT '97, International Conference on the Theory and Application of Cryptographic Techniques, Konstanz, Germany, May 11-15, 1997, Proceeding*. Ed. by Walter Fumy. Vol. 1233. Lecture Notes in Computer Science. Springer, 1997, pp. 52–61. ISBN: 3-540-62975-0. DOI: 10.1007/3-540-69053-0_5. URL: https://doi.org/10.1007/3-540-69053-0_5.

[CMT01] Jean-Sébastien Coron, David M'Raïhi, and Christophe Tymen. "Fast Generation of Pairs (k, [k]P) for Koblitz Elliptic Curves." In: *Selected Areas in Cryptography, 8th Annual International Workshop, SAC 2001 Toronto, Ontario, Canada, August 16-17, 2001, Revised Papers*. Ed. by Serge Vaudenay and Amr M. Youssef. Vol. 2259. Lecture Notes in Computer Science. Springer, 2001, pp. 151–164. ISBN: 3-540-43066-0.

[CPS08] Jean-Sébastien Coron, Jacques Patarin, and Yannick Seurin. "The Random Oracle Model and the Ideal Cipher Model Are Equivalent." In: *CRYPTO 2008*. Ed. by David Wagner. Vol. 5157. LNCS. Springer, Heidelberg, Aug. 2008, pp. 1–20. DOI: 10.1007/978-3-540-85174-5_1.

[DSST17] Yuanxi Dai, Yannick Seurin, John P. Steinberger, and Aishwarya Thiruvengadam. "Indifferentiability of Iterated Even-Mansour Ciphers with Non-idealized Key-Schedules: Five Rounds Are Necessary and Sufficient." In: *CRYPTO 2017, Part III*. Ed. by Jonathan Katz and Hovav Shacham. Vol. 10403. LNCS. Springer, Heidelberg, Aug. 2017, pp. 524–555. DOI: 10.1007/978-3-319-63697-9_18.

[Den02] Alexander W. Dent. "Adapting the Weaknesses of the Random Oracle Model to the Generic Group Model." In: *ASIACRYPT 2002*. Ed. by Yuliang Zheng. Vol. 2501. LNCS. Springer, Heidelberg, Dec. 2002, pp. 100–109. DOI: 10.1007/3-540-36178-2_6.

[Des00] Anand Desai. "The Security of All-or-Nothing Encryption: Protecting against Exhaustive Key Search." In: *CRYPTO 2000*. Ed. by Mihir Bellare. Vol. 1880. LNCS. Springer, Heidelberg, Aug. 2000, pp. 359–375. DOI: 10.1007/3-540-44598-6_23.

[DH76] Whitfield Diffie and Martin E. Hellman. "New Directions in Cryptography." In: *IEEE Transactions on Information Theory* 22.6 (1976), pp. 644–654.

[ElG84] Taher ElGamal. "A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms." In: *CRYPTO'84*. Ed. by G. R. Blakley and David Chaum. Vol. 196. LNCS. Springer, Heidelberg, Aug. 1984, pp. 10–18.

[ElG86] Taher ElGamal. "On Computing Logarithms Over Finite Fields." In: *CRYPTO'85*. Ed. by Hugh C. Williams. Vol. 218. LNCS. Springer, Heidelberg, Aug. 1986, pp. 396–402. DOI: 10.1007/3-540-39799-X_28.

[EM93] Shimon Even and Yishay Mansour. "A Construction of a Cipher From a Single Pseudorandom Permutation." In: *ASIACRYPT'91*. Ed. by Hideki Imai, Ronald L. Rivest, and Tsutomu Matsumoto. Vol. 739. LNCS. Springer, Heidelberg, Nov. 1993, pp. 210–224. DOI: 10.1007/3-540-57332-1_17.

[FKV17] Pooya Farshim, Louiza Khati, and Damien Vergnaud. "Security of Even–Mansour Ciphers under Key-Dependent Messages." In: *IACR Trans. Symm. Cryptol.* 2017.2 (2017), pp. 84–104. ISSN: 2519-173X. DOI: 10.13154/tosc.v2017.i2.84-104.

[FP15] Pooya Farshim and Gordon Procter. "The Related-Key Security of Iterated Even-Mansour Ciphers." In: *FSE 2015*. Ed. by Gregor Leander. Vol. 9054. LNCS. Springer, Heidelberg, Mar. 2015, pp. 342–363. DOI: 10.1007/978-3-662-48116-5_17.

[FN17a] Houda Ferradi and David Naccache. *Integer Reconstruction Public-Key Encryption*. Cryptology ePrint Archive, Report 2017/1231. https://eprint.iacr.org/2017/1231. 2017.

[FN17b] Houda Ferradi and David Naccache. *Integer Reconstruction Public-Key Encryption*. Cryptology ePrint Archive, Report 2017/1231. https://eprint.iacr.org/2017/1231. 2017.

[FS87] Amos Fiat and Adi Shamir. "How to Prove Yourself: Practical Solutions to Identification and Signature Problems." In: *CRYPTO'86*. Ed. by Andrew M. Odlyzko. Vol. 263. LNCS. Springer, Heidelberg, Aug. 1987, pp. 186–194. DOI: 10.1007/3-540-47721-7_12.

[Gal12] Steven D. Galbraith. *Mathematics of Public Key Cryptography*. Cambridge University Press, 2012. ISBN: 9781107013926. URL: https://www.math.auckland.ac.nz/~sgal018/crypto-book/crypto-book.html.

[GM84] Shafi Goldwasser and Silvio Micali. "Probabilistic Encryption." In: *Journal of Computer and System Sciences* 28.2 (1984), pp. 270–299.

[GOR11] Vipul Goyal, Adam O'Neill, and Vanishree Rao. "Correlated-Input Secure Hash Functions." In: *TCC 2011*. Ed. by Yuval Ishai. Vol. 6597. LNCS. Springer, Heidelberg, Mar. 2011, pp. 182–200. DOI: 10.1007/978-3-642-19571-6_12.

[Gre13] Glenn Greenwald. "NSA collecting phone records of millions of Verizon customers daily." In: *The Guardian* 6.06 (2013).

[HRC16] United Nations HRC. *33/2. The safety of journalists*. Resolution adopted by the 33rd Session of the United Nations Human Rights Council. Oct. 2016. URL: http://ap.ohchr.org/documents/dpage_e.aspx?si=A/HRC/RES/33/2.

[HK07] Shai Halevi and Hugo Krawczyk. "Security under key-dependent inputs." In: *ACM CCS 2007*. Ed. by Peng Ning, Sabrina De Capitani di Vimercati, and Paul F. Syverson. ACM Press, Oct. 2007, pp. 466–475. DOI: 10.1145/1315245.1315303.

[Hoc59] Alexis Hocquenghem. "Codes correcteurs d'erreurs." In: *Chiffres* 2.2 (1959), pp. 147–56.

[HPS98] Jeffrey Hoffstein, Jill Pipher, and Joseph H. Silverman. "NTRU: A Ring-Based Public Key Cryptosystem." In: *Algorithmic Number Theory, Third International Symposium, ANTS-III, Portland, Oregon, USA, June 21-25, 1998, Proceedings*. Ed. by Joe Buhler. Vol. 1423. Lecture Notes in Computer Science. Springer, 1998, pp. 267–288. ISBN: 3-540-64657-4. DOI: 10.1007/BFb0054868. URL: https://doi.org/10.1007/BFb0054868.

[HS03] Jeffrey Hoffstein and Joseph H. Silverman. "Random small Hamming weight products with applications to cryptography." In: *Discrete Applied Mathematics* 130.1 (2003), pp. 37–49.

[HL05] Susan Hohenberger and Anna Lysyanskaya. "How to Securely Outsource Cryptographic Computations." In: *TCC 2005*. Ed. by Joe Kilian. Vol. 3378. LNCS. Springer, Heidelberg, Feb. 2005, pp. 264–282. DOI: 10.1007/978-3-540-30576-7_15.

[How07] Nick Howgrave-Graham. "A Hybrid Lattice-Reduction and Meet-in-the-Middle Attack Against NTRU." In: *Advances in Cryptology - CRYPTO 2007, 27th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2007, Proceedings*. Ed. by Alfred Menezes. Vol. 4622. Lecture Notes in Computer Science. Springer, 2007, pp. 150–169. ISBN: 978-3-540-74142-8. DOI: 10.1007/978-3-540-74143-5_9. URL: https://doi.org/10.1007/978-3-540-74143-5_9.

[IKNP03] Yuval Ishai, Joe Kilian, Kobbi Nissim, and Erez Petrank. "Extending Oblivious Transfers Efficiently." In: *CRYPTO 2003*. Ed. by Dan Boneh. Vol. 2729. LNCS. Springer, Heidelberg, Aug. 2003, pp. 145–161. DOI: 10.1007/978-3-540-45146-4_9.

[IK04] Tetsu Iwata and Tadayoshi Kohno. "New Security Proofs for the 3GPP Confidentiality and Integrity Algorithms." In: *FSE 2004*. Ed. by Bimal K. Roy and Willi Meier. Vol. 3017. LNCS. Springer, Heidelberg, Feb. 2004, pp. 427–445. DOI: 10.1007/978-3-540-25937-4_27.

[Jou09] Antoine Joux. *Algorithmic cryptanalysis*. CRC Press, 2009.

[JLNT09] Antoine Joux, Reynald Lercier, David Naccache, and Emmanuel Thomé. "Oracle-Assisted Static Diffie-Hellman Is Easier Than Discrete Logarithms." In: *12th IMA International Conference on Cryptography and Coding*. Ed. by Matthew G. Parker. Vol. 5921. LNCS. Springer, Heidelberg, Dec. 2009, pp. 351–367.

[KL07] Jonathan Katz and Yehuda Lindell. *Introduction to Modern Cryptography*. Chapman and Hall/CRC Press, 2007. ISBN: 978-1-58488-551-1.

[KR01] Joe Kilian and Phillip Rogaway. "How to Protect DES Against Exhaustive Key Search (an Analysis of DESX)." In: *Journal of Cryptology* 14.1 (Jan. 2001), pp. 17–35. DOI: 10.1007/s001450010015.

[KU16] Mehmet Sabir Kiraz and Osmanbey Uzunkol. "Efficient and verifiable algorithms for secure outsourcing of cryptographic computations." In: *Int. J. Inf. Sec.* 15.5 (2016), pp. 519–537. DOI: 10.1007/s10207-

015-0308-7. URL: http://dx.doi.org/10.1007/s10207-015-0308-7.

[KF17] Paul Kirchner and Pierre-Alain Fouque. "Revisiting Lattice Attacks on Overstretched NTRU Parameters." In: *Advances in Cryptology - EUROCRYPT 2017 - 36th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Paris, France, April 30 - May 4, 2017, Proceedings, Part I*. Ed. by Jean-Sébastien Coron and Jesper Buus Nielsen. Vol. 10210. Lecture Notes in Computer Science. 2017, pp. 3–26. ISBN: 978-3-319-56619-1. DOI: 10.1007/978-3-319-56620-7_1. URL: https://doi.org/10.1007/978-3-319-56620-7_1.

[Knu93] Lars R. Knudsen. "Cryptanalysis of LOKI91." In: *AUSCRYPT'92*. Ed. by Jennifer Seberry and Yuliang Zheng. Vol. 718. LNCS. Springer, Heidelberg, Dec. 1993, pp. 196–208. DOI: 10.1007/3-540-57220-1_62.

[KM15] Neal Koblitz and Alfred J. Menezes. "The random oracle model: a twenty-year retrospective." In: *Des. Codes Cryptography* 77.2-3 (2015), pp. 587–610.

[LS13] Rodolphe Lampe and Yannick Seurin. "How to Construct an Ideal Cipher from a Small Set of Public Permutations." In: *ASIACRYPT 2013, Part I*. Ed. by Kazue Sako and Palash Sarkar. Vol. 8269. LNCS. Springer, Heidelberg, Dec. 2013, pp. 444–463. DOI: 10.1007/978-3-642-42033-7_23.

[LLL82] Arjen Klaas Lenstra, Hendrik Willem Lenstra, and László Lovász. "Factoring polynomials with rational coefficients." In: *Mathematische Annalen* 261.4 (1982), pp. 515–534.

[LL94] Chae Hoon Lim and Pil Joong Lee. "More Flexible Exponentiation with Precomputation." In: *CRYPTO'94*. Ed. by Yvo Desmedt. Vol. 839. LNCS. Springer, Heidelberg, Aug. 1994, pp. 95–107. DOI: 10.1007/3-540-48658-5_11.

[MN96] David M'Raïhi and David Naccache. "Batch Exponentiation: A Fast DLP-Based Signature Generation Strategy." In: *CCS '96, Proceedings of the 3rd ACM Conference on Computer and Communications Security, New Delhi, India, March 14-16, 1996*. Ed. by Li Gong and Jacques Stearn. ACM, 1996, pp. 58–61.

[Mau94] Ueli M. Maurer. "Towards the Equivalence of Breaking the Diffie-Hellman Protocol and Computing Discrete Algorithms." In: *CRYPTO'94*. Ed. by Yvo

Desmedt. Vol. 839. LNCS. Springer, Heidelberg, Aug. 1994, pp. 271–281. DOI: 10.1007/3-540-48658-5_26.

[MP04] Ueli M. Maurer and Krzysztof Pietrzak. "Composition of Random Systems: When Two Weak Make One Strong." In: *TCC 2004*. Ed. by Moni Naor. Vol. 2951. LNCS. Springer, Heidelberg, Feb. 2004, pp. 410–427. DOI: 10.1007/978-3-540-24638-1_23.

[NSS01] Phong Q. Nguyen, Igor E. Shparlinski, and Jacques Stern. "Distribution of modular sums and the security of the server aided exponentiation." In: *Cryptography and Computational Number Theory*. Springer, 2001, pp. 331–342.

[NS99] Phong Q. Nguyen and Jacques Stern. "The Hardness of the Hidden Subset Sum Problem and Its Cryptographic Implications." In: *CRYPTO'99*. Ed. by Michael J. Wiener. Vol. 1666. LNCS. Springer, Heidelberg, Aug. 1999, pp. 31–46. DOI: 10.1007/3-540-48405-1_3.

[NS01] Phong Q. Nguyen and Jacques Stern. "The Two Faces of Lattices in Cryptology." In: *Cryptography and Lattices, International Conference, CaLC 2001, Providence, RI, USA, March 29-30, 2001, Revised Papers*. Ed. by Joseph H. Silverman. Vol. 2146. Lecture Notes in Computer Science. Springer, 2001, pp. 146–180. ISBN: 3-540-42488-1. DOI: 10.1007/3-540-44670-2_12. URL: https://doi.org/10.1007/3-540-44670-2_12.

[PUB77] NIST FIPS PUB. "46-3. Data Encryption Standard." In: *Federal Information Processing Standards, National Bureau of Standards, US Department of Commerce* (1977).

[PH78] Stephen C. Pohlig and Martin E. Hellman. "An improved algorithm for computing logarithms over $GF(p)$ and its cryptographic significance." In: *IEEE Trans. Information Theory* 24.1 (1978), pp. 106–110. DOI: 10.1109/TIT.1978.1055817. URL: http://dx.doi.org/10.1109/TIT.1978.1055817.

[PS96] David Pointcheval and Jacques Stern. "Security Proofs for Signature Schemes." In: *EUROCRYPT'96*. Ed. by Ueli M. Maurer. Vol. 1070. LNCS. Springer, Heidelberg, May 1996, pp. 387–398. DOI: 10.1007/3-540-68339-9_33.

[PS00] David Pointcheval and Jacques Stern. "Security Arguments for Digital Signatures and Blind Signatures." In: *Journal of Cryptology* 13.3 (June 2000), pp. 361–396. DOI: 10.1007/s001450010003.

[RS60] Irving S. Reed and Gustave Solomon. "Polynomial codes over certain finite fields." In: *Journal of the society for industrial and applied mathematics* 8.2 (1960), pp. 300–304.

[Reg16] General Data Protection Regulation. "Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46." In: *Official Journal of the European Union (OJ)* 59.1-88 (2016), p. 294.

[RSA78] Ronald L. Rivest, Adi Shamir, and Leonard M. Adleman. "A Method for Obtaining Digital Signature and Public-Key Cryptosystems." In: *Communications of the Association for Computing Machinery* 21.2 (1978), pp. 120–126.

[Sch90] Claus-Peter Schnorr. "Efficient Identification and Signatures for Smart Cards." In: *CRYPTO'89*. Ed. by Gilles Brassard. Vol. 435. LNCS. Springer, Heidelberg, Aug. 1990, pp. 239–252. DOI: 10.1007/0-387-34805-0_22.

[SOOS95] Richard Schroeppel, Hilarie K. Orman, Sean W. O'Malley, and Oliver Spatscheck. "Fast Key Exchange with Elliptic Curve Systems." In: *CRYPTO'95*. Ed. by Don Coppersmith. Vol. 963. LNCS. Springer, Heidelberg, Aug. 1995, pp. 43–56. DOI: 10.1007/3-540-44750-4_4.

[Sha71] Daniel Shanks. "Class number, a theory of factorization, and genera." In: *Proc. of Symp. Math. Soc., 1971*. Vol. 20. 1971, pp. 41–440.

[Sho97] Victor Shoup. "Lower Bounds for Discrete Logarithms and Related Problems." In: *EUROCRYPT'97*. Ed. by Walter Fumy. Vol. 1233. LNCS. Springer, Heidelberg, May 1997, pp. 256–266. DOI: 10.1007/3-540-69053-0_18.

[Uni07] European Union. *Consolidated version of the Treaty on the Functioning of the European Union*. Dec. 2007. URL: http://eur-lex.europa.eu/resource.html?uri=cellar:c382f65d-618a-4c72-9135-1e68087499fa.0006.02/DOC_3&format=PDF.

[Uni12] European Union. *Charter of Fundamental Rights of the European Union*. Oct. 2012. URL: http://www.europarl.europa.eu/charter/pdf/text_en.pdf.

[de 95] Peter de Rooij. "Efficient Exponentiation using Precomputation and Vector Addition Chains." In: *EUROCRYPT'94*. Ed. by Alfredo De Santis. Vol. 950. LNCS. Springer, Heidelberg, May 1995, pp. 389–399. DOI: 10.1007/BFb0053453.

[de 97] Peter de Rooij. "On Schnorr's Preprocessing for Digital Signature Schemes." In: *Journal of Cryptology* 10.1 (Dec. 1997), pp. 1–16. DOI: 10.1007/s001459900016.

[den90] Bert den Boer. "Diffie-Hellman is as Strong as Discrete Log for Certain Primes (Rump Session)." In: *CRYPTO'88*. Ed. by Shafi Goldwasser. Vol. 403. LNCS. Springer, Heidelberg, Aug. 1990, pp. 530–539. DOI: 10.1007/0-387-34799-2_38.

PERSONAL PUBLICATIONS

[ABCGMRR18] Arash Atashpendar, Marc Beunardeau, Aisling Connolly, Remi Geraud, David Mestel, A. W. Roscoe, and Peter Y. A. Ryan. "From Clustering Supersequences to Entropy Minimizing Subsequences for Single and Double Deletions." In: *CoRR* abs/1802.00703 (2018). arXiv: 1802.00703. URL: http://arxiv.org/abs/1802.00703.

[BCGN16a] Marc Beunardeau, Aisling Connolly, Remi Geraud, and David Naccache. "Cdoe Obofsucaitn: Securing Software from Within." In: *IEEE Security & Privacy* 14.3 (2016), pp. 78–81.

[BCGN16b] Marc Beunardeau, Aisling Connolly, Remi Geraud, and David Naccache. "Fully Homomorphic Encryption: Computations with a Blindfold." In: *IEEE Security & Privacy* 14.1 (2016), pp. 63–67.

[BCGN16c] Marc Beunardeau, Aisling Connolly, Remi Geraud, and David Naccache. "White-Box Cryptography: Security in an Insecure Environment." In: *IEEE Security & Privacy* 14.5 (2016), pp. 88–92.

[BCGN17a] Marc Beunardeau, Aisling Connolly, Rémi Géraud, and David Naccache. "On the Hardness of the Mersenne Low Hamming Ratio Assumption." In: *Progress in Cryptology - LATINCRYPT 2017 - 5th International Conference on Cryptology and Information Security in Latin America, Havana, Cuba, September 20-22, 2017, Revised Selected Papers*. 2017, pp. 166–174. DOI: 10.1007/978-3-030-25283-0_9. URL: https://doi.org/10.1007/978-3-030-25283-0_9.

[BCGN17b] Marc Beunardeau, Aisling Connolly, Remi Geraud, and David Naccache. "The Case for System Command Encryption." In: *ACM Asia Conference on Computer and Communications Security (ASIACCS) 2017, Abu Dhabi, UAE*. 2017.

[BCGNV17] Marc Beunardeau, Aisling Connolly, Remi Geraud, David Naccache, and Damien Vergnaud. "Reusing nonces in Schnorr signatures." In: *Proceedings of the 22nd European Symposium on Research in Computer Security, ESORICS 2017, Oslo, Norway, September 11–15*. 2017.

[Con18] Aisling Connolly. "Freedom of Encryption." In: *IEEE Security & Privacy* 16.1 (2018), pp. 102–103. DOI: 10.1109/MSP.2018.1331023. URL: https://doi.org/10.1109/MSP.2018.1331023.

[CFF19] Aisling Connolly, Pooya Farshim, and Georg Fuchsbauer. "Key-Correlated Security for Symmetric Primitives." In: (2019). To appear.

RÉSUMÉ

Cette thèse présente des résultats nouveaux portant sur trois domaines fondamentaux de la cryptographie : les propriétés de sécurité, les hypothèses cryptographiques, et l'efficacité algorithmique.
La première partie s'intéresse à la sécurité des primitives symétriques. Nous introduisons une nouvelle propriété de sécurité correspondant à la plus forte sécurité pour les primitives symétriques prouvées sûres dans le modèle de l'oracle aléatoire. Les attaques par clé corrélées capturent les scénarios dans lesquels toutes les entrées (clés, messages, et éventuellement nonces et en-têtes) sont corrélées avec avec la clé secrète. Sous des hypothèses relativement faibles nous prouvons la sécurité contre les attaques par clé corrélées pour les algorithmes de chiffrement par bloc, et montrons que trois tours d'Even-Mansour sont nécessaires pour cela. Nous étendons ensuite les attaques par clés corrélées au chiffrement authentifié basé sur les nonces, et fournissons une transformation en boîte noire qui, partant d'un chiffrement authentifié à utilisateurs multiples, donne un chiffrement authentifié démontré résistant aux attaques par clés corrélés dans le modèle de l'oracle aléatoire. Nous établissons les relations et séparations avec les notions déjà existantes (sécurité contre les attaques par clés apparentées et par message dépendant de la clé) pour montrer que la sécurité contre les attaques par clé corrélés est strictement plus forte, et implique les autres.
La partie suivante porte sur la cryptographie à clé publique, et analyse les hypothèses sous-jacentes au nouveau cryptosystème introduit dans AJPS17. La cryptanalyse de cette hypothèse, reposant sur l'arithmétique modulo un premier de Mersenne, nous permet de reconstruire la clé secrète à partir de la clé publique uniquement. Nous proposons alors une variante de ce sytème à clé publique, fondée sur une modification de l'hypothèse précédente, résistant aux attaques connues (classiques et quantiques).
La dernière partie aborde l'efficacité algorithmique du schéma de signature de Schnorr. En mettant à profit la structure de groupe nous pouvons tirer parti du nonce pour produire un lot de signatures. Combinant ceci avec des méthodes de précalcul nous permet de rendre plus efficace l'algorithme de signature de Schnorr . La sécurité est préservée sous une hypothèse nouvelle, dont on montre qu'elle est vraie dans le modèle du groupe générique.

MOTS CLÉS

Cryptographie Symétrique, Cryptographie Asymétrique, Signatures Numériques, Preuve de sécurité, Cryptanalyse.

ABSTRACT

This thesis presents new results addressing three fundamental areas of cryptography: security notions, assumptions, and efficiency.
The first part encompasses the security of symmetric primitives. We give a new security notion that provides the strongest security for symmetric primitives proven in the random oracle model (ROM). Key-correlated attacks (KCA) model the scenario where all inputs (keys, messages, and possibly nonces and headers) are correlated with the secret key. Under mild assumptions, we prove KCA security of blockciphers, and show that 3-rounds of Even-Mansour are necessary to achieve this. Then, we define a KCA-security notion for nonce-based authenticated encryption (AE), and provide a black-box transformation that turns a multiuser-secure AE into an AE scheme that is provably KCA secure in the ROM. We show relations and separations with older notions (related-key and key-dependent message security) to show that security under KCA is strictly stronger, and implies the others.
The next part turns to public-key cryptography, and analyses the assumptions underlying the new public-key cryptosystem of AJPS17. Cryptanalysis of their assumption, based on arithmetic modulo a Mersenne prime, allowed us to reconstruct the secret key, given only the public key. Based on a modified version of the assumption, we propose a variant post-quantum secure public-key cryptosystem.
The last part turns to efficiency, and studies the Schnorr Signature scheme. Exploiting the group structure we can generate multiple nonces (instead of just one) with which we can then generate a batch of signatures. This, together with some preprocessing tricks, allow us to increase efficiency of Schnorr signature generation. Security is maintained under a new assumption shown intractable in the Generic Group Model.

KEYWORDS

Symmetric Cryptography, Public-key Cryptography, Digital Signatures, Provable Security, Cryptanalysis.