THE JOY OF CRYPTOGRAPHY

Mike Rosulek ⟨mike@joyofcryptography.com⟩
School of Electrical Engineering & Computer Science
Oregon State University, Corvallis, Oregon, USA

Draft of March 21, 2019
Preface

The Joy of Cryptography is an undergraduate textbook in cryptography. This book grew out of lecture notes I developed for the cs427 course at Oregon State University (and before that, cs473 at the University of Montana).

I am well aware that the title is ridiculous, but all of the serious titles were already taken. At this point I’m committed to the gag, for better or worse. Anyway, actual joy not guaranteed.

Information for Students

What Will You Learn In This Book?

I’m not going to lie. This book has a theoretical flavor, that reflects my personal bias as a theorician.

I understand that theory-for-theory’s-sake doesn’t motivate everyone in the same way that it motivates me. If I can’t get everyone to fall in love with the theory, my instructional goal is to ensure that everyone can at least appreciate it. In the book I try to keep the real-world implications of the theory in view.

The book does cover:

► How it is possible to formally define security properties and reason about them mathematically.

► How the most common cryptographic constructions work: what makes them secure, while similar constructions are insecure?

► The difference between different kinds of cryptographic primitives (PRFs, block ciphers, encryption, MACs, hash functions, etc). This includes differences in their interfaces, differences in their security properties, and most importantly, how to think about which primitive is best suited for a particular security goal.

The book does not cover:

► How to use encryption/privacy software like PGP, TrueCrypt, Signal, etc.

► Cryptocurrencies like Bitcoin.

► How to safely implement production-ready cryptographic algorithms. At times the book hints at some implementation issues, mostly to show how incredibly difficult it is to get things right.
What goes inside low-level primitives like block ciphers and hash functions. I think readers of this book are much more likely to build systems out of these primitives, rather than design their own primitives. Thus, the focus is on understanding what these different primitives provide, and how to combine them in sound ways.

Background Knowledge

You will get the most out of this book if you have a solid foundation in standard undergraduate computer science material:

- Discrete mathematics (of the kind you typically find in year 2 or 3 of an undergraduate CS program) is **required background.** The book assumes familiarity with basic modular arithmetic, discrete probabilities, simple combinatorics, and especially proof techniques.

- Algorithms & data structures background is **highly recommended,** and theory of computation (automata, formal languages & computability) is also **recommended.** We deal with computations and algorithms at a high level of abstraction, and with mathematical rigor. This can be a significant challenge if you haven’t had prior experience from these courses.

A quick look at Chapter 0 will give you more specifics about what kind of knowledge is assumed in this book.

Information for Teachers

Disclaimers & Apologies

Although I’ve used this book as a primary course reference for several years now, I still consider it to be a draft. Of course I make every effort to ensure the accuracy of the content, but there are sure to be plenty of bugs, ranging in their severity. *Caveat emptor!*

I teach a 10-week cryptography course, since my institution is on the quarter system. I manage to cover essentially all of this book in those 10 weeks. Because of this, it is always easier for me to focus on polishing the existing material rather than adding entirely new chapters. Someday I will add those new chapters (see the roadmap below), but currently there are some quite shameful omissions.

There is no solutions manual, and I currently have no plans to make one.

I welcome feedback of all kinds — not just on errors and typos but also on the selection, organization, and presentation of the material.

Code-based games

The security definitions and proofs in these notes are presented in a style that is known to the research community as **code-based games.** I’ve chosen this style because I think it offers significant pedagogical benefits:

- Every security definition can be expressed in the same style, as the indistinguishability of two games. In my terminology, the games are **libraries** with a common
interface/API but different internal implementations. An adversary is any calling program on that interface. These libraries use a concrete pseudocode that reduces ambiguity about an adversary’s capabilities. For instance, the adversary controls arguments to subroutines that it calls and sees only the return value. The adversary cannot see any variables that are privately scoped to the library.

A consistent framework for definitions leads to a consistent process for proving and breaking security — the two fundamental activities in cryptography.

In these notes, breaking a construction always corresponds to writing a program that expects a particular interface and behaves as differently as possible in the presence of two particular implementations of the interface.

Proving security nearly always refers to showing a sequence of libraries (called hybrids), each of which is indistinguishable from the previous one. Each of these hybrids is written in concrete pseudocode. By identifying what security property we wish to prove, we identify what the endpoints of this sequence must be. The steps that connect adjacent hybrids are stated in terms of syntactic rewriting rules for pseudocode, including down-to-earth steps like factoring out and inlining subroutines, changing the value of unused variables, and so on.

Cryptography is full of conditional statements of security: “if A is a secure thingamajig, then B is a secure doohickey.” A conventional proof of such a statement would address the contrapositive: “given an adversary that attacks the doohickey-security of B, I can construct an attack on the thingamajig-security of A.”

In my experience, students struggle to find the right way to transform an abstract, hypothetical B-attacking adversary into a successful A-attacking adversary. By defining security in terms of games/libraries, we can avoid this abstract challenge, and indeed avoid the context switch into the contrapositive altogether. In these notes, the thingamajig-security of A gives the student a new rewriting rule that can be placed in his/her toolbox and used to bridge hybrids when proving the doohickey-security of B.

Code-based games were first proposed by Shoup¹ and later expanded by Bellare & Rogaway.² These notes adopt a simplified and unified style of games, since the goal is not to encompass every possible security definition but only the fundamental ones. The most significant difference in style is that the games in these notes have no explicit INITIAlIZE or FINALIZE step. As a result, all security definitions are expressed as indistinguishability of two games/libraries, even security definitions that are fundamentally about unforgeability. Yet, we can still reason about unforgeability properties within this framework. For instance, to say that no adversary can forge a MAC, it suffices to say that no adversary can distinguish a MAC-verification subroutine from a subroutine that always returns FALSE.

An index of security definitions has been provided at the end of the book.

²Mihir Bellare & Philip Rogaway: Code-Based Game-Playing Proofs and the Security of Triple Encryption. ia.cr/2004/331
and easy to interpret (and perhaps I achieved neither in the end). See Chapter 11 for my best attempt.

Supplementary Material

Security proofs in this book follow a standard pattern: We start from one “library” and perform a sequence of small, cumulative modifications. Each modification results in a separate hybrid library that is indistinguishable from the previous one.

I have prepared PDF slide decks to supplement the security proofs contained in the book. They are available from the course website. The slides allow the reader to step forward and backward through the proof’s sequence of logical steps, seeing only the current hybrid library at any time (with changes highlighted and annotated).

Other Boring Stuff

Copyright

This work is copyright by Mike Rosulek and made available under the Creative Commons BY-NC-SA 4.0 license. Under this license, you are free to:

Share: copy and redistribute the material in any medium or format.

Adapt: remix, transform, and build upon the material.

The licensor cannot revoke these freedoms as long as you follow the following license terms:

Attribution: You must give appropriate credit, provide a link to the license, and indicate if changes were made. You may do so in any reasonable manner, but not in any way that suggests the licensor endorses you or your use.

NonCommercial: You may not use the material for commercial purposes.

ShareAlike: If you remix, transform, or build upon the material, you must distribute your contributions under the same license as the original.

You may not apply legal terms or technological measures that legally restrict others from doing anything the license permits.

About the cover

The cover design consists of assorted shell illustrations from Bibliothèque conchyliologique, published in 1846. The images are no longer under copyright, and were obtained from the Biodiversity Heritage Library (http://biodiversitylibrary.org/bibliography/11590). Like a properly deployed cryptographic primitive, a properly deployed shell is the most robust line of defense for a mollusk. To an uniformed observer, a shell is just a shell, and crypto is just crypto. However, there are a wide variety of cryptographic primitives, each of which provides protection against a different kind of attack. Just as for a seasoned conchologist,
the joy is in appreciating the unique beauty of each form and understanding the subtle
differences among them.

Acknowledgements
Some financial support for writing this book has been kindly provided by the National
Science Foundation (awards #1149647, #1617197) and the Oregon State University Open
Textbook Initiative.

Thanks to Brent Carmer & Leo Reyzin for many thoughtful suggestions and comments
about the material. I am also grateful for the many students in CS427 who have reported
countless bugs.

Changelog
2019-03-21: Chapter 11 (hash functions) significant revisions: no more impenetrable se-
curity definition for collision-resistance; explicit treatment of salts; better examples for
Merkle-Damgård and length-extension. New draft Chapter 15 on AEAD (after next revi-
sion will be inserted after Chapter 11).

2019-01-07: Extensive revisions; only the major ones listed here. Lots of homework
problems added/updated throughout. I tried to revise the entire book in time for my Win-
ter 2019 offering, but ran out of time.

► Added a changelog!
► Chapter 1: Kerckhoffs’ Principle now discussed here (previously only mentioned for
the first time in Ch 2).
► Chapter 2: Now the concepts are introduced in context of specific one-time security
definition, not in the abstract. More examples of interchangeable libraries.
► Chapter 3: Polynomial interpolation now shown explicitly with LaGrange polyno-
mials (rather than Vandermonde matrices). Full interpolation example worked out.
► Chapter 4: Better organization. Real-world contextual examples of extreme (large
& small) $2^n$ values. Full proof of bad-event lemma. Generalized avoidance-sampling
libraries.
► Chapter 5: Motivate PRGs via pseudo-OTP idea. Better illustration of PRG function,
and conceptual pitfalls. How NOT to build a PRG. New section on stream cipher &
symmetric ratchet.
► Chapter 6: Combined PRF & PRP chapters. Motivate PRFs via $m \mapsto (r, F(k, r) \oplus m)$
construction. Better discussion of eager vs. lazy sampling of exponentially large
table. How NOT to build a PRF. New section on constructing PRG from PRF, and
more clarity on security proofs with variable number of hybrids. Better illustrations
& formal pseudocode for Feistel constructions.
► Chapter 7: Other ways to avoid insecurity of deterministic encryption (stateful &
nonce-based). Ridiculous Socratic dialog on the security of the PRF-based encryp-
tion scheme.
Chapter 8: Compare & contrast CTR & CBC modes.

Road Map

The following topics are shamefully missing from the book, but are planned or being considered:

1. signature schemes, authenticated key agreement, secure messaging / ratcheting (high priority)
2. random oracle & ideal cipher models (medium priority)
3. elliptic curves, post-quantum crypto (but I need to learn them first)
4. DH-based socialist millionaires, PSI, PAKE, simple PIR, basic MPC concepts (low priority)
# Contents

0 **Review of Concepts & Notation**  
0.1 Modular Arithmetic ............................................. 1  
0.2 Strings ................................................................. 2  
0.3 Functions ............................................................... 2  
0.4 Probability ............................................................. 3  
0.5 Asymptotics (Big-O) ................................................ 4  

1 **One-Time Pad & Kerckhoffs' Principle** 6  
1.2 Specifics of One-Time Pad ...................................... 9  

2 **The Basics of Provable Security** 16  
2.1 Generalizing and Abstracting One-Time Pad .......... 16  
2.2 Towards an Abstract Security Definition ................. 18  
2.3 Provable Security Fundamentals ............................. 22  
2.4 How to Prove Security with The Hybrid Technique .... 28  
2.5 How to Demonstrate Insecurity with Attacks .......... 32  

3 **Secret Sharing** 39  
3.1 Definitions ............................................................ 39  
3.2 A Simple 2-out-of-2 Scheme ................................... 43  
3.3 Polynomial Interpolation ....................................... 46  
3.4 Shamir Secret Sharing ........................................... 50  
* 3.5 Visual Secret Sharing ........................................... 54  

4 **Basing Cryptography on Intractable Computations** 59  
4.1 What Qualifies as a “Computationally Infeasible” Attack? 59  
4.2 What Qualifies as a “Negligible” Success Probability? 62  
4.3 Indistinguishability ................................................. 64  
4.4 Birthday Probabilities & Sampling With/out Replacement 68  

5 **Pseudorandom Generators** 77  
5.1 Definitions ............................................................ 77  
5.2 Pseudorandom Generators in Practice ..................... 79  
5.3 Application: Shorter Keys in One-Time-Secret Encryption 82  
* 5.4 Contrapositive Point of View on Security Proofs ........ 84  
5.5 Extending the Stretch of a PRG .............................. 86  
* 5.6 Applications: Stream Cipher & Symmetric Ratchet .... 88  

6 **Pseudorandom Functions & Block Ciphers** 96  
6.1 Definition .............................................................. 97  
6.2 PRFs vs PRGs; Variable-Hybrid Proofs ..................... 100  
6.3 Block Ciphers (Pseudorandom Permutations) .............. 110  
6.4 Relating PRFs and Block Ciphers ............................ 111  
6.5 PRFs and Block Ciphers in Practice ....................... 114  

© Copyright Mike Rosulek. Creative Commons BY-NC-SA 4.0. Latest version at joyofcryptography.com.
6.6 Strong Pseudorandom Permutations ........................................ 115

7 Security Against Chosen Plaintext Attacks ................................. 120
  7.1 Limits of Deterministic Encryption ....................................... 120
  7.2 Pseudorandom Ciphertexts .................................................. 123
  7.3 CPA-Secure Encryption Based On PRFs ................................. 125

8 Block Cipher Modes of Operation ............................................. 134
  8.1 A Tour of Common Modes ................................................... 134
  8.2 CPA Security and Variable-Length Plaintexts ......................... 137
  8.3 Security of OFB Mode ...................................................... 139
  8.4 Padding & Ciphertext Stealing ............................................ 142

9 Chosen Ciphertext Attacks ..................................................... 152
  9.1 Padding Oracle Attacks .................................................... 152
  9.2 What Went Wrong? .......................................................... 155
  9.3 Defining CCA Security ...................................................... 158
  9.4 A Simple CCA-Secure Scheme ............................................ 161

10 Message Authentication Codes ............................................... 172
  10.1 Definition ................................................................. 172
  10.2 A PRF is a MAC ............................................................ 176
  10.3 MACs for Long Messages ................................................ 181
  10.4 Encrypt-Then-MAC ........................................................ 184

11 Hash Functions .................................................................. 191
  11.1 Security Properties for Hash Functions ................................ 191
  11.2 Merkle-Damgård Construction ............................................ 195
  11.3 Hash Functions vs. MACs: Length-Extension Attacks ............ 198

12 The RSA Function ............................................................... 204
  12.1 Modular Arithmetic & Number Theory .................................. 204
  12.2 The RSA Function .......................................................... 207
  12.3 Chinese Remainder Theorem ............................................. 208
  12.4 The Hardness of Factoring \( N \) .......................................... 210
  12.5 Malleability of RSA, and Applications ................................. 213

13 Diffie-Hellman Key Agreement ............................................... 220
  13.1 Cyclic Groups ............................................................... 220
  13.2 Diffie-Hellman Key Agreement .......................................... 221
  13.3 Decisional Diffie-Hellman Problem .................................... 222

14 Public-Key Encryption .......................................................... 226
  14.1 Security Definitions ........................................................ 226
  14.2 One-Time Security Implies Many-Time Security ................... 227
  14.3 ElGamal Encryption ......................................................... 230
  14.4 Hybrid Encryption ........................................................ 233

ix
15 Authenticated Encryption & AEAD

15.1 Definitions .......................................................... 238
15.2 Achieving AE/AEAD .................................................. 240
15.3 Carter-Wegman MACs ............................................... 241
15.4 Galois Counter Mode for AEAD ................................. 248

Index of Security Definitions ........................................... 249