

Computer Security and the Internet: Tools and Jewels from Malware to Bitcoin

(Second Edition)

Paul C. van Oorschot

The official version of this book is available at
<https://www.springer.com/gp/book/9783030834104>

ISBN: 978-3-030-83410-4 (hardcopy), 978-3-030-83411-1 (eBook)

Copyright ©2020-2022 Paul C. van Oorschot. Under publishing license to Springer.

For personal use only.

This author-created, self-archived copy is from the author's web page.

Reposting, or any form of redistribution without permission, is strictly prohibited.

To the knowledge and best efforts of the author at the time of writing, the content herein is accurate and technically correct. It is distributed on an “As Is” basis, without warranty of any kind, expressed or implied. Neither the author nor the publisher shall have any liability, under any circumstances, with respect to any direct or indirect loss or damage caused by, or alleged to be related to, use of any content in this book.

version: 29 June 2021

Pages ii, iii, iv to be provided by Springer

To J, J and E

for bringing joy to every day

and for understanding that universal decryption keys do not exist

Contents in Brief

Table of Contents	ix
Preface to Second Edition	xv
Foreword	xix
Preface to First Edition	xxiii
Typesetting Conventions	xxix
Chapter 1: Security Concepts and Principles	1
Chapter 2: Cryptographic Building Blocks	29
Chapter 3: User Authentication—Passwords, Biometrics and Alternatives	55
Chapter 4: Authentication Protocols and Key Establishment	91
Chapter 5: Operating System Security and Access Control	125
Chapter 6: Software Security—Exploits and Privilege Escalation	155
Chapter 7: Malicious Software	183
Chapter 8: Public-Key Certificate Management and Use Cases	213
Chapter 9: Web and Browser Security	245
Chapter 10: Firewalls and Tunnels	281
Chapter 11: Intrusion Detection and Network-Based Attacks	309
Chapter 12: Wireless LAN Security: 802.11 and Wi-Fi	339
Chapter 13: Bitcoin, Blockchains and Ethereum	375
Epilogue	411
Index	417

Table of Contents

Preface to Second Edition	xv
Foreword	xix
Preface to First Edition	xxiii
Typesetting Conventions	xxix
Chapter 1: Security Concepts and Principles ...	1
1.1 Fundamental goals of computer security	2
1.2 Computer security policies and attacks	4
1.3 Risk, risk assessment, and modeling expected losses	6
1.4 Adversary modeling and security analysis	9
1.5 Threat modeling: diagrams, trees, lists and STRIDE	11
1.6 Model-reality gaps and real-world outcomes	16
1.7 ‡Design principles for computer security	20
1.8 ‡Why computer security is hard	25
1.9 ‡End notes and further reading	27
References	28
Chapter 2: Cryptographic Building Blocks ...	29
2.1 Encryption and decryption (generic concepts)	30
2.2 Symmetric-key encryption and decryption	32
2.3 Public-key encryption and decryption	37
2.4 Digital signatures and verification using public keys	39
2.5 Cryptographic hash functions	41
2.6 Message authentication (data origin authentication)	45
2.7 ‡Authenticated encryption and further modes of operation	47
2.8 ‡Certificates, elliptic curves, and equivalent keylengths	49
2.9 ‡End notes and further reading	51
References	52
Chapter 3: User Authentication—Passwords, Biometrics and Alternatives ...	55
3.1 Password authentication	56
3.2 Password-guessing strategies and defenses	59
3.3 Account recovery and secret questions	65
3.4 One-time password generators and hardware tokens	67
3.5 Biometric authentication	71
3.6 ‡Password managers and graphical passwords	76
3.7 ‡CAPTCHAs (humans-in-the-loop) vs. automated attacks	79
3.8 ‡Entropy, passwords, and partial-guessing metrics	81
3.9 ‡End notes and further reading	86
References	88

Chapter 4: Authentication Protocols and Key Establishment ...	91
4.1 Entity authentication and key establishment (context)	92
4.2 Authentication protocols: concepts and mistakes	97
4.3 Establishing shared keys by public agreement (DH)	100
4.4 Key authentication properties and goals	104
4.5 Password-authenticated key exchange: EKE and SPEKE	105
4.6 ‡Weak secrets and forward search in authentication	111
4.7 ‡Single sign-on (SSO) and federated identity systems	113
4.8 ‡Cyclic groups and subgroup attacks on Diffie-Hellman	115
4.9 ‡End notes and further reading	120
References	122
Chapter 5: Operating System Security and Access Control ...	125
5.1 Memory protection, supervisor mode, and accountability	127
5.2 The reference monitor, access matrix, and security kernel	130
5.3 Object permissions and file-based access control	133
5.4 Setuid bit and effective userid (eUID)	137
5.5 Directory permissions and inode-based example	138
5.6 Symbolic links, hard links and deleting files	142
5.7 Role-based (RBAC) and mandatory access control	144
5.8 ‡Protection rings: isolation meets finer-grained sharing	146
5.9 ‡Relating subjects, processes, and protection domains	149
5.10 ‡End notes and further reading	151
References	153
Chapter 6: Software Security—Exploits and Privilege Escalation ...	155
6.1 Race conditions and resolving filenames to resources	157
6.2 Integer-based vulnerabilities and C-language issues	159
6.3 Stack-based buffer overflows	166
6.4 Heap-based buffer overflows and heap spraying	168
6.5 ‡Return-to-libc exploits	171
6.6 Buffer overflow exploit defenses and adoption barriers	172
6.7 Privilege escalation and the bigger picture	174
6.8 ‡Background: process creation, syscalls, shells, shellcode	176
6.9 ‡End notes and further reading	178
References	180

Chapter 7: Malicious Software ...	183
7.1 Defining malware	184
7.2 Viruses and worms	186
7.3 Virus anti-detection and worm-spreading techniques	191
7.4 Stealth: Trojan horses, backdoors, keyloggers, rootkits	194
7.5 Rootkit detail: installation, object modification, hijacking	197
7.6 ¶Drive-by downloads and droppers	200
7.7 Ransomware, botnets and other beasts	202
7.8 Social engineering and categorizing malware	205
7.9 ¶End notes and further reading	207
References	209
Chapter 8: Public-Key Certificate Management and Use Cases ...	213
8.1 Certificates, certification authorities and PKI	214
8.2 Certificate chain validation and certificate extensions	217
8.3 ¶Certificate revocation	221
8.4 CA/PKI architectures and certificate trust models	224
8.5 TLS web site certificates and CA/browser trust model	229
8.6 Secure email overview and public-key distribution	235
8.7 ¶Secure email: specific technologies	238
8.8 ¶End notes and further reading	241
References	242
Chapter 9: Web and Browser Security ...	245
9.1 Web review: domains, URLs, HTML, HTTP, scripts	246
9.2 TLS and HTTPS (HTTP over TLS)	252
9.3 HTTP cookies and DOM objects	255
9.4 Same-origin policy (DOM SOP)	257
9.5 Authentication cookies, malicious scripts and CSRF	260
9.6 More malicious scripts: cross-site scripting (XSS)	262
9.7 SQL injection	266
9.8 ¶Usable security, phishing and web security indicators	269
9.9 ¶End notes and further reading	274
References	276

Chapter 10: Firewalls and Tunnels ...	281
10.1 Packet-filter firewalls	282
10.2 Proxy firewalls and firewall architectures	288
10.3 SSH: Secure Shell	292
10.4 VPNs and encrypted tunnels (general concepts)	297
10.5 \ddagger IPsec: IP security suite (details)	300
10.6 \ddagger Background: networking and TCP/IP	303
10.7 \ddagger End notes and further reading	306
References	307
Chapter 11: Intrusion Detection and Network-Based Attacks ...	309
11.1 Intrusion detection: introduction	310
11.2 Intrusion detection: methodological approaches	313
11.3 Sniffers, reconnaissance scanners, vulnerability scanners	316
11.4 Denial of service attacks	320
11.5 Address resolution attacks (DNS, ARP)	325
11.6 \ddagger TCP session hijacking	329
11.7 \ddagger End notes and further reading	332
References	335
Chapter 12: Wireless LAN Security: 802.11 and Wi-Fi ...	339
12.1 Background: 802.11 WLAN architecture and overview	340
12.2 WLAN threats and mitigations	343
12.3 Security architecture: access control, EAP and RADIUS	347
12.4 RC4 stream cipher and its use in WEP	351
12.5 WEP attacks: authentication, integrity, keystream reuse	353
12.6 WEP security summary and full key recovery	357
12.7 \ddagger AES-CCMP frame encryption and key hierarchy	361
12.8 Robust authentication, key establishment and WPA3	364
12.9 \ddagger End notes and further reading	369
References	371

Chapter 13: Bitcoin, Blockchains and Ethereum ...	375
13.1 Bitcoin overview	376
13.2 Transaction types and fields	379
13.3 ‡Bitcoin script execution (signature validation)	382
13.4 Block structure, Merkle trees and the blockchain	384
13.5 Mining of blocks, block preparation and hashing targets	386
13.6 Building the blockchain, validation, and full nodes	391
13.7 ‡Simple payment verification, user wallets, private keys	395
13.8 ‡Ethereum and smart contracts	399
13.9 ‡End notes and further reading	405
References	407
Epilogue	411
Index	417

‡This symbol denotes sections that readers or instructors may elect to omit on first reading, or if time-constrained. A few of these are background sections, which students may read on their own as review. The end notes completing each section suggest both further elementary readings, and entry points to the research literature.

Preface to Second Edition

Perfection is finally attained not when there is no longer anything to add, but when there is no longer anything to take away.

—Antoine de Saint-Exupéry, 1900–1944

What is new in the Second Edition

The Second Edition adds two new chapters (discussed next). Minor editorial revisions were also made to improve clarity (without altering pagination), for example in statements of the Chapter 1 security design principles.

Chapter 12 (new) gives an overview of wireless local area network (WLAN) security, then drills down to examine its common implementation using IEEE standard 802.11, recognizable by widespread *Wi-Fi* branding of laptops and mobile phones. This provides lessons on the challenges of designing and deploying complete systems in practice, shows that design errors and new attacks are more common than expected, and enables an understanding of current WLAN security architectures and the evolution of WPA3 (*Wi-Fi Protected Access* version 3). Many concepts from earlier chapters are pulled together, from threat models to practical difficulties of key management to subtle design choice errors allowing the defeat of strong cryptographic algorithms. Readers learn what happens when security technology meets real world products and motivated attackers.

Chapter 13 (new) explains how blockchains work, with a main focus on Bitcoin’s virtual cryptography-based currency system, often called a *cryptocurrency*. We leverage this to also introduce Ethereum, which broadens the underlying blockchain ideas to a decentralized computing model supporting programs called *smart contracts*. Independent of current hype, blockchain methods appear destined to have broad and long-term impact, through their ability to deliver publically verifiable, append-only (immutable) records, whose integrity relies neither on a centralized (trusted) party nor on the confidentiality of a master secret key. This makes blockchains and understanding their properties important for everyone interested in information technology. Many nations are also exploring the potential of cryptocurrencies as government-backed (i.e., *fiat*) currencies. We note also that while cryptography plays a crucial role in blockchains, the cryptographic background required for Chapter 13 is surprisingly little and localized—the main concepts relied on are (1) the basic collision-resistance property of cryptographic hash functions, and (2) the high-level (input-output) functionality of digital signatures. Thus, a fear of cryptography should not stop instructors or students from learning about blockchain systems.

Suggestions for including the expanded content into shorter courses

The Second Edition now contains sufficient content for a two-term undergraduate course. The optional exercises (which are largely written as introductions to further material) and curated references in chapter end notes may be used as the basis for short projects, or as extra content for advanced students.

This raises the question of how best to use the book for a course taught over a single 13-week term or quarter—as few instructors are likely to have time to cover all chapters in one term. Instructors of a one-term course now enjoy a rich set of options from which to build a course matching their personal tastes; we outline a few of these in what follows.

As noted in the original Preface, the first eleven chapters suit a tightly run single-term course in computer and network security, with the expectation that some sections marked as optional (or instructor-selected substitutes) will be omitted. For one-term courses, a simple option is thus to ignore the new chapters and fall back on our convention of double-daggers (§) on section and paragraph headings suggesting material to consider omitting. However, we expect that most instructors will prefer to teach one or both of the new chapters, as they are of high interest to students, were added based on real-world relevance, and offer compelling examples of how complete systems implement security in practice. To this end, Fig. 1 gives two examples of chapter sequences for one-term courses.

Fig. 1(a) suggests a single-term course focused on authentication and uses of cryptography in Internet security. This path uses Chapters 2–4 as its basis. Instructors here might opt to: reduce coverage of Chapters 5–7, but retain §7.7 (to highlight how ransomware exploits public-key cryptography); emphasize, in Chapter 8 (certificate management), both the browser trust model and secure email (§8.5–8.7); omit parts of Chapter 9 (web security) but retain §9.2 (TLS) and §9.8 (usability of certificates); within Chapter 10 (firewalls and tunnels), focus on §10.3–10.5 (SSH, VPNs, IPsec); and from Chapter 11 (intrusion detection and network attacks), focus on §11.5 (current DNS weaknesses call attention to the need for cryptographic protection) and §11.6 (showing the importance of session encryption to stop TCP session hijacking). Both new chapters are included in this path.

Fig. 1(b) suggests a single-term course that emphasizes network and systems security with lighter coverage of cryptography. Here, as the shaded boxes suggest, other courses may provide cryptographic background (e.g., Chapters 2, 4, 8), and perhaps also the basics

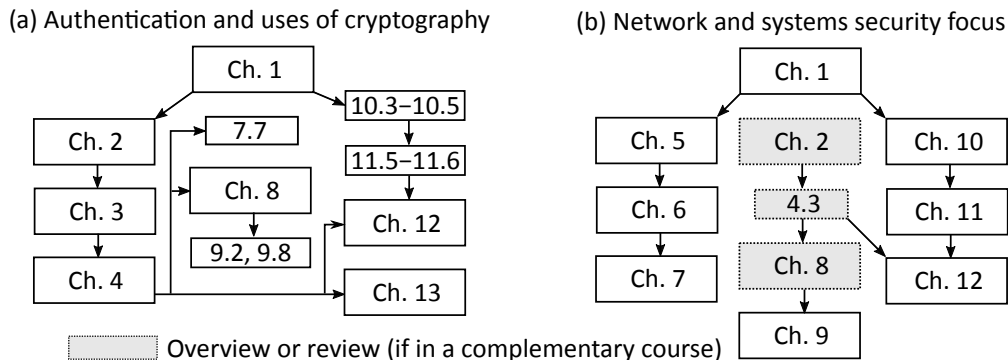


Figure 1: Example roadmaps from which to build single-term courses.

of passwords and user authentication (Chapter 3). Students will likely still benefit from a review of Chapter 2, §4.3 (Diffie-Hellman key exchange), and Chapter 8 (certificate-based public-key infrastructure). The core elements of systems security in this path are operating systems, software security and malicious software (Chapters 5–7); those of network security are in Chapters 10–12, while web security (Chapter 9) spans systems and network security. The primary content may again be enriched by using material from exercises and end notes; topics such as network monitoring and vulnerability scanning tools (§11.3) are natural candidates among many other examples. Path (b) is recommended for students with access to a complementary course that focuses on applied cryptography.

Who this book aims to serve

The primary audience is students in a one-term or two-term, third- or fourth-year undergraduate course in a computer science, engineering or information technology program, or an introductory graduate course. Graduate students beginning a research program may use this book as a framework for learning the subject, and are a natural audience for the optional inline exercises, and literature references in chapter end notes. Instructors and established researchers may find this to be a useful handbook for organizing knowledge of computer security, software and web security, authentication, and the security of networked systems. A secondary audience is those open to self-study in order to gain a general understanding of these areas as background knowledge, including developers and information technology professionals, or indeed anyone else interested and having a basic understanding of computers or software (entry-level familiarity with operating systems and networking concepts will ease this journey, but can be picked up along the way). Finally, the new Chapter 13 offers a concise, yet broad, systematic introduction to Bitcoin and Ethereum for individuals specifically interested in these systems, but without the time to read entire books on these single subjects. Many such entire books themselves assume background beyond that of interested readers; in contrast, we are able to efficiently explain how these systems work, by relying on a few key concepts introduced in earlier, self-contained chapters.

Acknowledgements (Second Edition)

I am grateful to Wayne Wheeler and Ronan Nugent at Springer for ongoing support and encouragement. Beyond those in the original Preface, I sincerely thank all colleagues who have kindly provided constructive feedback to improve this book, its overall evolution, and most recently Chapters 12 (*) and 13 (*), especially:

Chris Bellman**	Jeremy Clark*	Yusef Karim*	Udaya Parampalli*
Rainer Böhme*	Will Enck*	M. Mannan**	Petr Svenda*
Joseph Bonneau*	Gus Gutoski*	Václav (Vashek) Matyáš**	Mathy Vanhoef*
Xavier Carnavalet*	Syed Rafiul Hussain*	Alfred Menezes*	Amr Youssef*

Any errors that remain are my own.

Paul C. van Oorschot
Ottawa, Canada
29 June 2021

About the Author

Paul C. van Oorschot is a Professor of Computer Science at Carleton University (Ottawa), where he is Canada Research Chair in Authentication and Computer Security. He is an ACM Fellow, an IEEE Fellow, and a Fellow of the Royal Society of Canada. He was Program Chair of the New Security Paradigms Workshop 2014-2015 (NSPW), USENIX Security Symposium 2008, and the Internet Society's Network and Distributed System Security Symposium 2001-2002 (NDSS). Prior to writing *Computer Security and the Internet: Tools and Jewels* (2020), he was co-author of the *Handbook of Applied Cryptography* (1996), and *An Introduction to Error Correcting Codes with Applications* (1989). He has served on the editorial boards of IEEE TDSC (*Transactions on Dependable and Secure Computing*), IEEE TIFS (*Transactions on Information Forensics and Security*), and ACM TISSEC/TOPS (*Transactions on Information and System Security*, now retitled *Transactions on Privacy and Security*). His research interests include authentication and identity management, computer security, Internet security, security and usability, software security, and applied cryptography. He earned a Ph.D. in Computer Science from the University of Waterloo in 1988. In 2017 he was appointed Honorary Professor at the University of Melbourne. His academic career was preceded by 14 years of industrial research and development in telecommunications and software security.

Foreword

There's an old adage that many people espouse: "Keep It Simple, Stupid". Unfortunately, when it comes to the nitty-gritty of securing computer systems, networks, and the Internet: Everything is Complicated. A highly relevant quote comes from Albert Einstein: "Everything should be made as simple as possible, *but no simpler.*" It is often making things too simple that leads to missing requirements, design flaws, implementation errors, and system failures.

At the end of Chapter 1, Paul Van Oorschot lists two dozen fundamental principles that should underlie the design and implementation of systems, and particularly systems with stringent requirements for trustworthiness (e.g., security, reliability, robustness, resilience, and human safety). These principles all can contribute in many ways to better systems, and indeed they are highlighted throughout each following chapter as they apply to particular situations.

One particularly desirable principled approach toward dealing with complexity involves the pervasive use of design abstraction with encapsulation, which requires carefully defined modular interfaces. If applied properly, this approach can give the appearance of simplicity, while at the same time actually hiding internal state information and other functional complexities behind the interface.

Paul's book identifies ten relatively self-contained structural areas of widespread concern, each of which is probed with detail sufficient to establish relatively accessible groundwork for the primary concepts. The book makes considerable headway into underlying realities that otherwise tend to make things difficult to understand, to design, and to implement correctly. It provides collected wisdom on how to overcome complexity in many critical areas, and forms a sound basis for many of the necessary fundamental components and concepts. Also, much of what is described here is well chosen, because it has survived the test of time.

Paul has a remarkably diverse background, which is reflected in the content of this book. He is one of the special people who has made major contributions to the literature in multiple areas: computer/network security (including certification-based system architectures, authentication, alternatives to passwords, Internet infrastructure, protocols, and misuse detection), applied cryptography (e.g., public-key infrastructure, enhanced authentication), software, and system usability. Much of his work crosses over all four of these areas, which are mirrored in the topics in this book. Paul has extensive background in both academia and the software industry. His earlier *Handbook of Applied Cryptography*

is highly regarded and widely used, and also reflects the cross-disciplinary thinking that went into writing *Tools and Jewels*.

The balance of Paul's academic and practical experiences is demonstrated by the somewhat unusual organization of this book. It takes a fresh view of each chapter's content, and focuses primarily on precisely what he believes should be taught in a first course on the subject.

To avoid readers expecting something else, I summarize what the book intentionally does *not* attempt to do. It does *not* seek to be a cookbook on how to build or integrate systems and networks that are significantly more secure than what is common today, partly because there is no one-size-fits-all knowledge; there are just too many alternative design and implementation choices. It also does *not* deal with computer architecture in the large—beginning with total-system requirements for hardware, software, and applications. However, it *is* oriented toward practical application rather than specific research results; nevertheless, it cites many important items in the literature. Understanding everything in this book is an essential precursor to achieving meaningfully trustworthy systems.

Altogether, Paul's realistic approach and structural organization in this book are likely to provide a very useful early step—particularly for students and emerging practitioners (e.g., toward being a knowledgeable system designer/developer/administrator), but also for computer users interested in a better understanding of what attaining security might entail. The book may also become an excellent starting point for anyone who subsequently wishes to understand the next stages of dealing with complexity, including layered and compositional architectures for achieving total-system trustworthiness in hardware, software, networks, and applications—as well as coping with the pitfalls inherent in system development, or in more wisely using the Internet.

Designing, developing, and using systems with serious requirements for trustworthiness has inherent complexities. Multics is a historical example of a clean-slate system—with new hardware, a new operating system, and a compiler that facilitated taking advantage of the new hardware features—that is worth studying as a major step forward into secure multi-access computing. Paul begins to invoke some of its architecture (e.g., in Chapter 5). But the real lessons from Multics may be its highly principled design and development process, which carefully exposed new problems and then resolved them with a long view of the future (e.g., completely avoiding stack buffer overflows and the Y2K problem in 1965!). Paul noted to me that if we don't teach anything about strong features of our old systems, how can we make progress?

Chapter 6 provides an example of the book's pragmatic focus, discussing software security and related vulnerabilities—headlined still, sadly, by buffer overruns (mentioned above). It provides background on a selection of well-known exploits currently in use, exposing the importance of software security. A main focus there, and throughout the book, is on helping readers understand what goes wrong, and how. This puts them in a position to appreciate the need for solutions, and motivates pursuit of further details in additional sources, e.g., using references in the chapter End Notes and occasional inline Exercises (and perhaps follow-up courses). This style allows instructors and readers to pick and choose elements to drill deeper, according to personal interests and time constraints.

Such an approach, and the book’s pervasive focus on principles, match my own interests quite well, e.g., including my ongoing involvement in developing the CHERI clean-slate capability-based hardware-software system, a highly principled effort taking advantage of the past history of computer systems. Among other foci, the CHERI hardware, software, and extended hardware-aware LLVM (low-level virtual machine) compiler specifically help remediate most of the software security issues that are the main focus of Chapter 6—including spatial and temporal memory safety issues, and protection against buffer overflows in C. Paul cites a CHERI reference in his far-reaching epilogue. (CHERI is a joint effort of SRI and the University of Cambridge.)

Each of the other chapters that I have not mentioned specifically here is a valuable contribution by itself. All in all, I believe Paul’s timely book will be extremely useful to a wide audience.

Peter G. Neumann
Chief Scientist, SRI International Computer Science Lab,
and moderator of the ACM Risks Forum
July 2019

Preface to First Edition

Do not write so that you can be understood, write so that you cannot be misunderstood.

—Epictetus, 55-135 AD

Why this book, approach and target audience

This book provides a concise yet comprehensive overview of computer and Internet security, suitable for a one-term introductory course for junior/senior undergrad or first-year graduate students. It is also suitable for self-study by anyone seeking a solid footing in security—including software developers and computing professionals, technical managers and government staff. An overriding focus is on brevity, without sacrificing breadth of core topics or technical detail within them. The aim is to enable a broad understanding in roughly 300 pages. Further prioritization is supported by designating as optional selected content within this. Fundamental academic concepts are reinforced by specifics and examples, and related to applied problems and real-world incidents.

The first chapter provides a gentle overview and 20 design principles for security. The ten chapters that follow aim to provide a framework for understanding computer and Internet security. They regularly refer back to the principles, with supporting examples. These principles are the conceptual counterparts of security-related error patterns that have been recurring in software and system designs for over 50 years.

The book is “elementary” in that it assumes no background in security, but unlike “soft” high-level texts, does not avoid low-level details; instead it selectively dives into fine points for exemplary topics, to concretely illustrate concepts and principles. The book is rigorous in the sense of being technically sound, but avoids both mathematical proofs and lengthy source-code examples that typically make books inaccessible to general audiences. Knowledge of elementary operating system and networking concepts is helpful, but review sections summarize the essential background. For graduate students, inline exercises and supplemental references provided in per-chapter end notes provide a bridge to further topics and a springboard to the research literature; for those in industry and government, pointers are provided to helpful surveys and relevant standards, e.g., documents from the Internet Engineering Task Force (IETF), and the U.S. National Institute of Standards and Technology.

Selection of topics

For a one-term course in computer and network security, what topics should you cover, in what order—and should breadth or technical depth be favored? We provide a roadmap.

A common complaint is the lack of a concise introductory book that provides a broad overview without being superficial. While no one book will meet the needs of all readers, existing books fall short in several ways. Detailed treatments on the latest advances in specialty areas will not be introductory-level. Books that dwell on recent trends rapidly become dated. Others are too long to be useful introductions—instructors who are not subject-area experts, and readers new to the subject, require guidance on what core material to cover, and what to leave for follow-up or special-topic courses. Some books fail at the presentation level (lacking the technical elements required for engineering and computer science students to develop an understanding), while others that provide detailed code-level examples often lack context and background.

Our aim is to address these deficiencies in a balanced way. Our choices of what to include and exclude, and when to provide low-level details vs. high-level overviews, are informed by guidance from peers, personal experience in industrial and academic research, and from teaching computer security and cryptography for 30 years. The presentation style—which some readers may find atypical—reflects the way in which I organize my own thoughts—metaphorically, putting things into boxes with labels. Thus the material is delivered in “modular” paragraphs, most given short titles indicating their main focus. Those familiar with the *Handbook of Applied Cryptography* (1996) will notice similarities. The topics selected also reflect a personal preference of the core content that I would expect of software developers starting out in industry, or junior security researchers. The material herein also corresponds to what I wish had been available in a book to learn from when I myself began in computer and network security many years ago. The soundness of these choices will be revealed in the course of time.

Framework and systematization

My hope is that this book may serve as a framework from which others may build their own tailored courses. Instructors who prefer to teach from their own notes and slide decks may find it helpful to point students to this text as a coherent baseline reference, augmenting its material with their own specialized content. Indeed, individual instructors with special expertise often wish to teach certain topics in greater detail or add extra topics and examples, while other experts will have analogous desires—but expanding different topics. While a single book clearly cannot capture all such material, the present book may serve as a common foundation and framework for such courses—providing instructors a unified overview and basis for further study in security, rather than leave students without a designated textbook.

Why use a book at all, if essentially all of the information can be found online, in pieces? Piecemeal sources leave students with inconsistent terminology, material of widely varying clarity and correctness, and a lack of supporting background (or redundancy thereof) due to sources targeting readers with different backgrounds. This makes learning inefficient for students. In contrast, services provided by a solid introductory text include: context with well-organized background, consistent terminology, content selection and prioritization, appropriate level of detail, and clarity.

A goal consistent with providing a framework is to help systematize knowledge by

carefully chosen and arranged content. Unlike “tidy” subareas such as cryptography, computer and Internet security as a broad discipline is not particularly orderly, and is less well structured—it often more closely resembles an ad hoc collection of vaguely related items and lessons. We collect these over time, and try to organize them into what we call knowledge. Books like this aim to arrive at a more unified understanding of a broad area, and how its subareas are related. Organization of material may help us recognize common techniques, methods and defensive approaches, as well as recurring attack approaches (e.g., middle-person attacks, social engineering). Note that where security lacks absolute rules (such as laws of physics), we fall back on principles. This acknowledges that we do not always have precise, well-defined solutions that can be universally applied.

Length, prioritization and optional sections

A major idea underlying a shorter book, consciously limited in total page count, is to avoid overwhelming novices. Many introductory security textbooks span 600 to 1000 pages or more. These have different objectives, and are typically a poor fit for a one-term course. While offering a wealth of possible topics to choose from, they are more useful as handbooks or encyclopaedias than introductory texts. They leave readers at a loss as to what to skip over without losing continuity, and the delivery of core topics is often split across several chapters.

In contrast, our approach is to organize discussion of major topics in single locations—thereby also avoiding repetition—and to make informed (hard) choices about which topics to cover. We believe that careful organization and informed selection of core material allows us to maintain equal breadth at half the length of comparable books. To accommodate the fact that some instructors will not be able to cover even our page-reduced content, our material is further prioritized by marking (with a double-dagger prefix, “‡”) the headings of sections that can be omitted without loss of continuity. Within undaggered sections, this same symbol denotes paragraphs and exercises that we suggest may be omitted on first reading or by time-constrained readers.

Counterintuitively, a longer book is easier to write in that it requires fewer choices by the author on what to omit. We recall the apology of Blaise Pascal (1623–1662) for the length of a letter, indicating that he did not have time to write a shorter one. (“*Je n’ai fait celle-ci plus longue que parce que je n’ai pas eu le loisir de la faire plus courte.*”)

Order of chapters, and relationships between them

The order of chapters was finalized after trying several alternatives. While each individual chapter has been written to be largely independent of others (including its own set of references), subsequences of chapters that work well are: (5, 6, 7), (8, 9) and (10, 11). The introduction (Chapter 1) is followed by cryptography (Chapter 2). User authentication (Chapter 3) then provides an easy entry point via widely familiar passwords. Chapter 4 is the most challenging for students afraid of math and cryptography; if this chapter is omitted on first pass, Diffie-Hellman key agreement is useful to pick up for later chapters.

Our prioritization of topics is partially reflected by the order of chapters, and sections within them. For example, Chapter 11 includes intrusion detection, and network-based attacks such as session hijacking. Both of these, while important, might be left out by

time-constrained instructors who may instead give priority to, e.g., web security (Chapter 9). These same instructors might choose to include, from Chapter 11, if not denial of service (DoS) in general, perhaps distributed DoS (DDoS) and pharming. This explains in part why we located this material in a later chapter; another reason is that it builds on many earlier concepts, as does Chapter 10. More fundamental concepts appear in earlier chapters, e.g., user authentication (Chapter 3), operating system security (Chapter 5), and malware (Chapter 7). Readers interested in end-user aspects may find the discussion of email and HTTPS applications in Chapter 8 appealing, as well as discussion of usable security and the web (Section 9.8); Information Technology staff may find, e.g., the details of IPsec (Chapter 10) more appealing. Software developers may be especially interested in Chapters 4 (using passwords as authentication keys), 6 (software security including buffer overruns and integer vulnerabilities), and 8 (public-key infrastructure).

Cryptography vs. security course

Our book is intended as a text for a course in computer security, rather than for one split between cryptography and computer security. Chapter 2 provides cryptographic background and details as needed for an introduction to security, assuming that readers do not have prior familiarity with cryptography. Some readers may choose to initially skip Chapter 2, referring back to parts of it selectively as needed, in a “just-in-time” strategy. A few cryptographic concepts are deferred to later chapters to allow in-context introduction (e.g., Lamport hash chains are co-located with one-time passwords in Chapter 3; Diffie-Hellman key agreement is co-located with middle-person attacks and key management in Chapter 4, along with small-subgroup attacks). Chapter 4 gives additional background on crypto protocols, and Chapter 8 also covers trusts models and public-key certificates. The goal of this crypto background is to make the book self-contained, including important concepts for developers who may, e.g., need to use crypto toolkits or make use of crypto-based mechanisms in web page design. In our own institution, we use this book for an undergrad course in computer and network security, while applied cryptography is taught in a separate undergrad course (using a different book).

Helpful background

Security is a tricky subject area to learn and to teach, as the necessary background spans a wide variety of areas from operating systems, networks, web architecture and programming languages to cryptography, human factors, and hardware architecture. The present book addresses this by providing mini-reviews of essential background where needed, supplemented by occasional extra sections often placed towards the end of the relevant chapter (so as not to disrupt continuity). As suggested earlier, it is helpful if readers have background comparable to a student midway through a computer science or computer engineering program, ideally with exposure to basic programming and standard topics in operating systems and network protocols (data communications). For example, readers will benefit from prior exposure to Unix (Chapter 5), memory layout for a run-time stack and the C programming language (Chapter 6), web technologies such as HTML, HTTP and JavaScript (Chapter 9), and basic familiarity with TCP/IP protocols (for Chapters 10 and 11). A summary of relevant prerequisite material is nonetheless given herein.

Trendy topics vs. foundational concepts

For a course that intends to convey the latest information on fast-moving areas, a web site is better suited than content delivery by a textbook. We have designed this book to avoid quickly becoming obsolete, by providing a solid foundation with focus on long-standing concepts, principles, and recurring error patterns. Perhaps surprisingly, a great many security-related errors are invariant over time, independent of evolving platforms. For this same reason, we find it instructive to teach about old flaws that are now fixed (temporarily, in specific products and protocols), as lessons to help us avoid repeating these errors on each new platform. (Mistakes are inevitably repeated, but this way we at least have names for them and can look up the best repair ideas from earlier.)

When teaching, we also rely on a complementary set of hands-on assignments and software tools; these may change as operating systems and technologies evolve. Many excellent, independent online resources are available for programming and tool-based security assignments; we do not attempt to tie hands-on assignments directly to this book itself. The “Exercises” interspersed throughout our chapters herein are not programming-based, but rather are simple review-type thought experiments or more frequently, pointers into the literature for additional material suggested as interesting next-step topics (albeit non-essential to a basic introduction). Along with references provided in the chapter end notes, these are suitable for strong undergrads or more commonly, beginning graduate students who seek exposure one level deeper or broader.

Acknowledgements

This book has benefited immensely from the thoughtful feedback of many colleagues and students. Among many others too numerous to mention, I wish to thank in particular:

AbdelRahman Abdou	Aleks Essex	M. Mannan	Reza Samanfar
Nicholas Akinyokun	Hemant Gupta	Ashraf Matrawy	R. Sekar
Furkan Alaca	Urs Hengartner	Václav (Vashek) Matyáš	Anil Somayaji
David Barrera	Cormac Herley	Lee McCallum	Elizabeth Stobert
Chris Bellman	Jason Hinek	Fabian Monrose	Tao Wan
Sonia Chiasson	James Huang	Peter G. Neumann	Dave L. Whyte
Peter Choynowski	Trent Jaeger	Bryan Parno	Glenn Wurster
Wayne Du	Poe Kgengwenyane	Joel Reardon	Tatu Ylönen
Will Enck	Hassan Khan	Vladimir Robotka	Lianying (Viau) Zhao

I would also like to thank Ronan Nugent at Springer for his utmost professionalism, and for making the publication process a true pleasure. Errors that remain are of course my own. I would appreciate any feedback (with details to allow corrections), and welcome all suggestions for improvement.

Paul C. van Oorschot
Ottawa, Canada
September 2019

Typesetting Conventions

Conventions used in this book for text fonts, coloring, and styles include the following.

	Examples
paragraph labels	INLINE HOOKING.
headings for examples	Example (<i>WannaCrypt 2017</i>).
headings for exercises	Exercise (Free-lunch attack tree).
emphasis (often regular words or phrases)	<i>This computer is secure</i>
technical terms	<i>passcode generators</i>
security principle (label for easy cross-reference)	LEAST-PRIVILEGE (P6)
software systems, tools or programs	<i>Linux, Firefox</i>
security incidents or malware name	<i>Morris worm, Code Red II</i>
filesystem pathnames and filenames	<i>/usr/bin/passwd, chmod.exe</i>
system calls and standard library functions	<i>exec(), execve(), malloc</i>
command-line utilities (as user commands)	passwd
OS data structures, flag bits, account names	<i>inode, setuid, root</i>
computer input, output, code or URL	<i>ls -al, http://domain.com</i>

