Secure Communicating Systems

More and more working computer professionals are actively confronted with the use, maintenance, or customization of cryptographic components and program certification mechanisms for local or remote (mobile) code. This text, meant for advanced undergraduate and beginning graduate students, tells what every computer scientist ought to know about cryptographic systems, security protocols, and secure information flow in programs. In addition to the standard material on public-key cryptosystems, stream and block ciphers, and certain secure communication protocols, the author presents several important topics not treated in most other texts:

- a detailed description of the new advanced encryption standard (AES) of NIST, the cipher Rijndael, announced as winner of the AES design competition on October 2, 2000;
- a complete description of an optimal public-key encryption using RSA that turns "textbook RSA" into a practical implementation whose semantic security is supported by a theoretical analysis conducted in the random oracle model;
- a current, formal discussion of standard security models for information flow in computer programs or human organizations;
- a presentation of a formal method for specifying and debugging security protocols; and
- a current discussion of the moral, legal, and political ramifications of cryptology and an overview of recent legislative efforts.

In addition, the text has WWW support and contains numerous implementation projects, a rigorous analysis of the Miller–Rabin algorithm, and a proof of the existence of primitive roots for prime powers.

Michael Huth is a Senior Lecturer in the Department of Computing at the Imperial College of Science, Technology and Medicine (London). He has also held positions at Kansas State University (Manhattan), the Technical University of Darmstadt, and the University of Birmingham. He has given numerous invited lectures and seminars and is the author of more than twenty papers on computer science and mathematics in international journals and conference proceedings. Together with Mark Ryan he wrote the textbook *Logic in Computer Science: Reasoning and Modelling about Systems,* recently published by Cambridge University Press.

Secure Communicating Systems

Design, Analysis, and Implementation

Michael R A Huth

Imperial College of Science, Technology and Medicine

CAMBRIDGE
UNIVERSITY PRESS

PUBLISHED BY THE PRESS SYNDICATE OF THE UNIVERSITY OF CAMBRIDGE
The Pitt Building, Trumpington Street, Cambridge, United Kingdom

CAMBRIDGE UNIVERSITY PRESS
The Edinburgh Building, Cambridge CB2 2RU, UK
40 West 20th Street, New York, NY 10011-4211, USA
10 Stamford Road, Oakleigh, VIC 3166, Australia
Ruiz de Alarcón 13, 28014 Madrid, Spain
Dock House, The Waterfront, Cape Town 8001, South Africa

http://www.cambridge.org

First published 2001

Printed in the United States of America

Typeface Times 10.5/13 pt. *System* AMS-TEX [FH]

A catalog record for this book is available from the British Library.

Library of Congress Cataloging in Publication Data
Huth, Michael, 1962–
Secure communicating systems : design, analysis, and implementation / Michael R A Huth.
p. cm.
Includes bibliographical references and index.
ISBN 0-521-80731-X
1. Telecommunication – Security measures.

TK5102.85.H88 2001
005.8 – dc21 2001025484

ISBN 0 521 80731 X hardback

Contents

Preface

In the past ten years, the dramatic growth of the Internet has had a profound and lasting impact on the way in which organizations and individuals communicate and conduct their public and private affairs. Tax forms are available online, students may submit their exams electronically to a (possibly remote) campus network, and companies may use the Internet as a public channel for linking up internal computing facilities or processes. For example, an employee may dial into a company's intranet from a hotel room or her home via a public Internet service provider. Since the Internet protocol does not provide sufficient mechanisms for ensuring the privacy, authenticity, integrity, and (if desired) anonymity of data that are processed through a usually dynamically determined chain of computers, there is a need for tools that guarantee the confidentiality and authenticity of data and of their communication sources and targets. Cautious consumers of mobile or foreign code prefer to verify that downloaded programs (e.g., Java applets) abide by a formal set of safety rules, possibly defined by the individual consumer. These needs appear to be even more pressing in the recent evolution of *electronic commerce,* where the act of selecting and purchasing a product occurs online. Although online companies are still waiting to reap their first real profits, it is evident that companies in general need to offer this mode of business in order to survive in a new economy that is global and at the same time strengthens regional identity.

The design and analysis of cryptographic systems, security protocols, and programs that process secret or confidential information – together with the safety analysis of (possibly foreign) code – are important tools for establishing a sufficient level of security and confidentiality between human agents, social groups, and machines that communicate over a public, and therefore untrusted, medium. Alas, current computer science and information technology degree programs typically only touch upon these topics in a course on operation systems or telecommunication systems within the larger context of "computer security". As more and more working computer professionals are actively confronted with the use, maintenance, or customization of cryptographic components and program certification mechanisms, I see a pressing need for a textbook, aimed at the advanced undergraduate and beginning graduate level, that teaches "what every computer scientist ought to know about cryptographic systems, security protocols, and secure information flow in programs". This book presents public-key cryptosystems, stream and block ciphers, certain secure communication protocols, and so forth that are usually covered in similar texts. However, this text distinguishes itself, and goes beyond most existing books, in several important ways.

1. It contains several topics that are quite novel and mostly absent from current textbooks:
   - a detailed description of the new advanced encryption standard (AES) of NIST, the cipher Rijndael, announced as the winner of the AES design competition on 2 October 2000;
   - a complete description of an optimal public-key encryption system using RSA that turns "textbook RSA" into a practical implementation whose semantic security is supported by a theoretical analysis conducted in the random oracle model;
   - a current and formal discussion of standard security models for information flow in computer programs or human organizations;
   - the presentation of a formal method for specifying and debugging security protocols;
   - log-in protocols based on zero-knowledge proofs;
   - the basics of elliptic curve public-key and signature systems;
   - the subtleties in meaning of terms used in the informal or formal specification of security protocols, exemplified by the term "authentication"; and
   - a discussion of the moral, legal, and political ramifications of cryptology and an overview of recent legislative efforts.
2. It provides a *cohesive text* with a *vast number of carefully designed and stated exercises,* many of which explore variations or extensions of material covered in this text at multiple levels of difficulty.
3. It features small *programming projects* that help clarify the nature and potential complexity of the number-theoretic concepts used in this text (e.g., how decryption and encryption work for RSA).
4. It animates each topic with substantial *implementation exercises* that are ideally assigned to *teams* of students.
5. It proves *in full detail* the correctness of the Miller–Rabin algorithm for primality testing, thereby making an important educational contribution to the analysis and design of (probabilistic) algorithms.
6. It includes a mathematically rigorous appendix on primitive roots, which allows for additional reading and course work by mathematics majors and makes this book appropriate and useful for a mathematics course in applied number theory.
7. It is supported by a website that contains ancillary material, such as Java source code for some of the programs featured in this text. This website features links to all the sites mentioned in the book as well as links to online papers and tutorials that complement or deepen the presented topics.

The cipher Rijndael will certainly become a global standard for symmetric encryption software and hardware, and it will be found in a full range of computational objects – from smartcards to mainframes. At the time of publication, this text is likely among the first to include a full exposition of this cipher.

The inclusion of an optimal public-key cryptosystem using RSA transforms RSA from its textbook version to a practical implementation that is rigorous and secure. To my knowledge, the discussion of such an important practical realization of RSA is absent from other textbooks on this subject.[1] This practical discussion is complemented by a proof of exact security results in the random oracle model.

[1] I acknowledge an anonymous reviewer who brought this to my attention and suggested that I include this material.

Another principal contribution that sets this text apart from existing ones is its elementary description and well-motivated design of tools built for formally reasoning about security protocols. As their analysis component, most texts consider mathematical and often sophisticated techniques for assessing the strength of, say, a particular block cipher encryption algorithm. Although these techniques are important, they are meant for the specialist whose task it is to design new – and attack existing – cryptographic algorithms. This text therefore delegates such specialized topics to the references, but it emphasizes the analysis of specified security protocols as a major task in avoiding the corruption of secrecy, integrity, and anonymity in a communication network. I base this choice on the fact that inherent design flaws in protocols are, next to implementation flaws and compromises, the most likely cause for a cryptographic system to be broken. Moreover, the detection of such design errors is typically as difficult as the discovery of bugs in ordinary synchronous or asynchronous concurrent systems. For the latter, automated (e.g., the model checker SMV) and semi-automated (e.g., the theorem prover PVS) tools and specification frameworks have been developed and are already being embraced by research and development labs. The tool I feature is a model checker combined with a natural deduction engine modeling an attacker, due to W. Marrero, E. Clarke, and S. Jha.

As another applied component, I discuss D. Denning's (1976, 1977) classical work on program certification for secure information flow but present it in a contemporary and rigorous framework of a type inference system. This treatment allows for a formal proof that this analysis of secure information flow in programs satisfies a noninterference property that can be used to guarantee secrecy or integrity of information flow. I then present a semantic approach to secure information flow in programs, due to R. Joshi and K. R. M. Leino, that uses weakest predicate transformers and partial correctness proofs for its refutation and validation of program security. This material, as well as the analysis part of the optimal RSA encryption, constitutes the more advanced part of this text and is likely to be covered in a graduate course or presented by talented undergraduate students in class.

Formal methods for the analysis of cryptographic systems and the secure flow of information in programs, or their secure execution, are currently a vibrant research area, and their fruitful development should be a vital step toward the establishment of sound methodologies for "cryptographic engineering", just as such working standards have already emerged in conventional software engineering. The education of future security engineers in such tools may also help to address the next set of challenges in security engineering on the Internet. For example: How can one establish and reason about a dynamically evolving "network of trusted nodes"? What are sound methodologies for the verification of complex specifications within multiparty protocols (electronic cash flow between consumers, merchants, and banks; broadcasting and multicasting communication sessions; etc.)? How can we realize efficient but reliable platforms for the definition, verification, and certification of safety policies for mobile code?

Cryptography and the certification of (mobile) code are certainly only two requirements for the establishment and maintenance of a reliably functioning digital society. Yet, considering that an alarming percentage of the current cryptographic products make poor or even unprofessional design decisions (choice of algorithm, key length, protocol, etc.), it seems evident that students ought to know the "dos and don'ts" of this area. Although this text is not meant to become a standard monograph or a standard reference text, I believe that it can well become the preferred choice of instructors who – while

not necessarily being experts in this field themselves – mean to effectively teach students whose backgrounds necessitate a delicate and careful presentation and development of nontrivial mathematical concepts and who need to see these concepts applied in a concrete context they can relate to; this I hope to accomplish through the inclusion of small programming exercises and larger implementation projects. Although competing texts present more cryptographic topics and at a more advanced level, instructors may decide to use this text because it reasons also about the secure behavior of programs, noting that a framework for trusted (mobile) code cannot be implemented with cryptographic techniques alone: We can use cryptography to authenticate the origin of mobile code or to ensure that this code has not been tampered with in transit; but even establishing all of that tells us nothing about the actual behavior of the program when it is executed locally.

This text contains more material than one could cover in a 12–15-week course. Beyond a common backbone of fundamentally important sections, instructors should feel free to omit or emphasize certain topics as they see fit for their individual course objectives. I took great care in presenting almost all the key issues, even though some may be condensed or confined to the exercises. At the same time, I strove for the creation of a relatively compact text that is highly interconnected and reasonably self-contained. The provided links to online research papers, tutorials, and cited references should enable instructors and students alike to extend appropriately the breadth and depth of the material presented here.

I have taken care to write this text without creating deep dependencies between any of its chapters. It is possible to read any of these chapters in isolation, as long as one has a "black-box understanding" of the concepts discussed in each chapter. Some dependencies, however, are inescapable. In particular, most topics discussed in Chapter 4 rely on material from the first three chapters.

So far, I have taught two interdisciplinary courses based on a draft of this text in three phases. The first phase was conducted in a "traditional" lecture style, where I made heavy use of this text in discussing the basics of symmetric and public cryptosystems and security protocols. During that time, I assigned additional reading and exercises from drafts of this book. In the second phase, I let student teams "implement" various standards (e.g., SHS, DSS, and triple DES) in a programming language of their choice. In the third phase, students made use of the more advanced part of this text or consulted online resources in order to identify papers and/or tools they chose to present in class. Feedback regarding these three phases, their mode, and their contents was extremely positive. Generally, students felt that the implementation work helped them solidify the mathematical underpinnings of the utilized techniques.

The supplementary material of this text is collected on the website

www.doc.ic.ac.uk/~mrh/scs

and includes the Java source code of some of the featured programs. Also included are links to research papers, repositories, tutorials, public and private standards, articles, and companies that promote their security products. The site features a current list of errata for this book; readers are kindly asked to report errors not found in that list to m.huth@doc.ic.ac.uk.

Acknowledgments

Many people have, directly or indirectly, assisted in writing and certainly improving this book. K. Rustan M. Leino made several critical suggestions on how to improve Section 6.3. Jason Lamm, Corina Păsăreanu, Guillaume Ravanas, and Matthew Zimmer pointed out several embarrassing typographical and conceptual errors. Wendy Bohnenkamp kept me informed on the current popular pulse in cryptography. Mark Ryan has provided substantial LaTeX support through consulting and the writing of style files. I made use of Paul Taylor's LaTeX style file for proof trees. The search engine www.google.com has been an effective tool that facilitated the writing of this text. I held illuminating conversations with David Schmidt on abstraction and weakest precondition semantics. I thank the numerous anonymous reviewers of various drafts of this text for their constructive and most helpful criticism; in fact, one of them encouraged me to write the chapter on optimal public-key encryption with RSA. My editor Lauren Cowles helped shape the vision of this text. I am also grateful for the enthusiasm and support of students at Kansas State University who made it challenging and rewarding to teach this material. Notwithstanding all this kind support, I am expressly and solely responsible for all errors of fact or presentation that this text may well include.