

Information Security and Cryptography

Series Editors

David Basin
Kenny Paterson

Advisory Board

Michael Backes
Gilles Barthe
Ronald Cramer
Ivan Damgård
Andrew D. Gordon
Joshua D. Guttman
Christopher Kruegel
Ueli Maurer
Tatsuaki Okamoto
Adrian Perrig
Bart Preneel

More information about this series at <http://www.springer.com/series/4752>

Mário S. Alvim • Konstantinos Chatzikokolakis
Annabelle McIver • Carroll Morgan
Catuscia Palamidessi • Geoffrey Smith

The Science of Quantitative Information Flow

 Springer

Mário S. Alvim
Computer Science Department
Universidade Federal de Minas Gerais
Belo Horizonte, Brazil

Annabelle McIver
Department of Computing
Macquarie University
Sydney, NSW, Australia

Catuscia Palamidessi
Inria Saclay and LIX
École Polytechnique
Institut Polytechnique de Paris
Palaiseau, France

Konstantinos Chatzikokolakis
Department of Informatics
and Telecommunications
University of Athens
Athens, Greece

Carroll Morgan
School of Computer Science
& Engineering
University of New South Wales
Trustworthy Systems, Data61
CSIRO
Sydney, NSW, Australia

Geoffrey Smith
School of Computing
& Information Sciences
Florida International University
Miami, FL, USA

ISSN 1619-7100 ISSN 2197-845X (electronic)
Information Security and Cryptography
ISBN 978-3-319-96129-3 ISBN 978-3-319-96131-6 (eBook)
<https://doi.org/10.1007/978-3-319-96131-6>

© Springer Nature Switzerland AG 2020

This work is subject to copyright. All rights are reserved by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed.

The use of general descriptive names, registered names, trademarks, service marks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.

The publisher, the authors, and the editors are safe to assume that the advice and information in this book are believed to be true and accurate at the date of publication. Neither the publisher nor the authors or the editors give a warranty, expressed or implied, with respect to the material contained herein or for any errors or omissions that may have been made. The publisher remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

This Springer imprint is published by the registered company Springer Nature Switzerland AG
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

The authors dedicate this book as follows:

Mário S. Alvim to his mother, Maria Angélica, his stepfather, Mario, his brothers, Marco Antônio and Marcus Vinícius, and his husband, Trevor.

Kostas Chatzikokolakis to his father, Thymios.

Annabelle McIver to her daughter, Eleanor, and her parents, Anne and Ted.

Carroll Morgan to the policy of diversity and tolerance deliberately instituted and actively sustained at Data61's Trustworthy Systems Group.

Catuscia Palamidessi to her husband, Dale Miller, and their children, Alexis and Nadia Miller.

Geoffrey Smith to his parents, Marilyn and Seward, his wife, Elena, his sons, Daniel and David, and his cockatiel, Yoshi.



Cockatiel *Yoshi* as a probabilistic channel C that maps a top-secret document X to a (randomly generated) pile of shredded paper Y

Preface

Information Flow is the transfer of information from a source (who knows the information) to a target (who does not yet know it). In history, that topic has sometimes been studied in order to *impede* flow (e.g. Caesar’s CIPHER from millennia ago), and sometimes to *facilitate* it (e.g. Shannon’s work in the 1940’s). Usually, however, the aims are a careful mixture of the two: to let information flow to those who need to know it, but to keep it from those who must not have it. That is the focus of our contemporary perspective –facilitate some flows, impede others– and our main (but not exclusive) concern here is computer systems.

But *first*: what is so special about now? Information-flow security is a critical problem today because of recent technological developments and their –largely uncontrolled– spread to many hands: all the way from everyday home users to super-skilled hackers, and all over the earth. Data is being collected more than ever before (smart phones, surveillance cameras, “loyalty” cards); networks then enable its transmission to unknown (or unintended) destinations; and powerful corporate and governmental agents gain financial and/or political benefits by collecting and analyzing that data. And, of course, there are the criminals.

Because so much is flowing, and so many have access to it, and we know so little specifically about who they are, we can no longer protect our information by relying on the people through whose hands it passes. Thus the standard technologies like *access control* and *encryption* are insufficient, because there we require the entities granted access to our data to handle it appropriately, and that implied trust might well be misplaced: a smartphone app could legitimately need access to our location, for example, but then leak that information to some other party, perhaps maliciously — but also perhaps just by accident.

Thus instead we must try to generate, process, and transfer our data with systems that protect *themselves*, that are safe no matter who accesses them or how they might abuse that access. It demands a fundamental, rigorous approach; and that fundamental rigor is exactly the *science* that we are striving for.

Thus, *second*: how can it be done? Early rigorous work in information-flow security (since the 1970’s) suggested ways in which programs could be analyzed to see whether the program variables an adversary “could see” might depend on variables that were not supposed to be seen: our secrets. If there was no dependence, then the program was secure; but if there was *any dependence at all*, then the program was deemed insecure. That “depends or not” criterion was later realized to be too coarse, however: even a password-checking program, no matter how carefully constructed, would be deemed insecure, because **Access Denied** still unavoidably exhibits a dependence — on what the password *is not*.

Quantitative information flow solves the “depends or doesn’t”, the “black or white” problem by relativizing information leaks, recognizing that it’s not really that clear-cut — some leaks are more important than others, and thus some are tolerable (e.g. leaking what a password isn’t, provided it’s only infrequently). A typical quantitative approach is to use Shannon’s information theory to measure the “entropy” of a secret (roughly, how hard it is to guess) before a system is run, and then to determine what the entropy would become after the program is run (by analyzing the source code, which we assume is available to our adversary). The difference between the two entropies, before minus after, is then how many bits have flowed from the system (escaped, if that flow is not desirable) and —again roughly— if it’s a small proportion of the bits that should remain secret, then the actual impact might be considered to be quite limited. Further, because the flow is quantified, the impact can actually be reasoned about rather than merely regretted. That technique realizes a powerful insight, and it works well in many situations: quantifying secrecy in the Shannon style (via entropy) provides the needed nuance to escape the earlier “all or nothing” judgments. For example, if the amount of entropy leaked by a failed login is indeed very small, it is exactly there that quantitative reasoning allows us to calculate with “very small” and “how often” and compare the result to “tolerable”.

But much more recently still, it was suggested that Shannon’s approach could be generalized, taken further, because in some situations also *it* turned out to be too inflexible: were the numbers it produced, how many bits escaped, really the numbers we needed to know? The generalization was to allow a *selection* of entropies —many more than just Shannon’s alone— whose characteristics were derived empirically from a study of the possible adversaries’ motivations and capabilities. Which secrets do they really want, and which ones would they not bother to steal? What exactly can they do with their knowledge about the secret? That last step —the generalized entropies— completes the conceptual trajectory from “Does information flow at all?” (simple dependence) through “How many bits of information flow?” (Shannon leakage) to finally (at least for the moment) “What is the *value to the adversary* of the information that flows?” or, dually, “What damage to us is caused by that flow, and how much would we spend (or should we have spent) to prevent it?” Generalized entropies (of which Shannon entropy is a special case) are captured by what we call “loss functions”; dually, we also consider generalized “vulnerabilities”, captured by “gain functions”. Furthermore, loss- and gain functions enable a connection with the science of program *development*, where specification programs are “refined” into implementation programs that satisfy those specifications both in terms of functionality and security. (Shannon-entropy leakage is not usually a compositional criterion; and yet compositionality is essential for reliable program construction. The use of generalized entropies, however, *is* compositional.)

For all of those reasons, our study of *the science of quantitative information flow* aims to understand fundamentally how sensitive information “flows” as it is processed by an authorized entity (e.g. our computer program), and to ensure that those flows are acceptable to us in terms of the quantified damage they might cause. And here —as we will emphasize— it is important to understand “flows” in a very broad sense: indeed flow occurs whenever sensitive information is *correlated* with observable outputs, allowing an adversary to make *inferences* about the sensitive information. Such correlations can be blatant, as when a sensitive file is copied to some publicly observable place, but they can also be subtle, as when a medical database outputs a patient’s country as “United States” if the patient has diabetes and as “USA” if not: in that case the patient’s diabetes status “flows” to the country output in a way that probably was not intended.

Extant studies of information flow encompass a variety of domains –such as non-interference, anonymity, unlinkability, secure multi-party computation, differential privacy, statistical databases, side channels, voting, and anonymous communication and publishing– and we have tried to do the same. Something that makes those studies challenging, and our study as well, is that perfection is often unachievable, because *some* undesirable flows cannot be helped. Publishing statistics about a database of medical records necessarily involves revealing some information about the individual records: keeping those records completely private is not an option in that case. Indeed there are many practical reasons for accepting flows that –in a perfect world– we would prefer not to have:

- Sometimes a flow is *intentional*: we *want* to learn something from our statistical database.
- Sometimes a flow is due to *side channels* that are hard or impossible to control fully.
- Sometimes a flow is in exchange for a *service*, one which for example might need our location.
- Sometimes a flow is in exchange for *efficiency*, as when a weaker but more efficient anonymous communication system is used instead of a stronger but less efficient protocol.

All of those support our belief that we must not (only) ask *whether* there is an information flow, and not even (only) *how many* bits of Shannon entropy might flow. We try to study instead *how much damage* an information flow would cause; and because of the generality of that approach, the earlier two are special cases.

The six authors of this book come from a number of distinct research domains, including process calculi, privacy, type systems for secure information flow, and programming-language semantics and refinement. As we all came to understand information flow better, we recognized that our efforts shared deep commonalities; and so, merging our earlier specialties, we have been working intensively as a group together since about 2010. This book is our comprehensive treatment of *quantitative information flow (QIF)* as we currently understand it — and we hope that it will lead to further and wider collaboration with those who might read it.

Much of what we present here is based on material already published, but by no means all of it — it is not at all merely “a collection of papers”. Instead we have tried hard to write a unified and self-contained text, hoping as we did that to find better terminology and notation than we might have used before, and then in some cases even rewriting whole presentations from scratch to take advantage of it. As well, in many cases we have also replaced earlier mathematical proofs with new ones that are clearer and more self-contained.

Finally, while this book is mainly focused on the systematic development of the theory of quantitative information flow, we also demonstrate the theory’s practical utility by including (in Part V) case studies showing how quantitative–information-flow analysis can be applied to a number of interesting realistic scenarios.

Intended readership

Our intended reader is anyone interested in the mathematical foundations of computer security. As far as the required technical background is concerned, we have tried to make the main story understandable to anyone with just a basic knowledge of discrete probability, though sometimes deeper concepts are used. But, in those cases, we have tried to minimize the need for prior familiarity by presenting the necessary material within our text.

It is worth clarifying however that this book is not aimed at readers interested in the legal, ethical, or sociological aspects of information flow. While it is clear that some information flows are beneficial and others are harmful, we make no effort to address the question of which are which.

And finally, we recognize that information flow is in fact a *general phenomenon* with relevance beyond security. So while the theory developed here has largely been motivated by the question of how to limit the leakage of sensitive information, that same theory can no doubt be applied fruitfully in diverse contexts such as machine learning, recommendation systems, and robotics. (Interestingly, in those contexts information flow would typically be seen as a *good* thing.) For this reason, readers outside the field of security may also profit from reading this book.

Organization and structure

We now briefly describe the overall structure of the book.

In Part I, we motivate the study of quantitative information flow, and we give an informal overview of some of its important concepts by discussing information leakage in a very simple context.

In Part II, we begin our detailed development by explaining what a *secret* X actually is, or at least what we consider it to be: a *probability distribution* π that specifies the adversary's knowledge about the likelihood of X 's possible values. We also consider how π can be used in quantifying either X 's *vulnerability* or (complementarily) the adversary's *uncertainty* about X , observing that there are *many* reasonable ways to do that, depending on the operational scenario, and showing that a single framework, based on “gain functions” (or dually “loss functions”), can encompass them all.

In Part III, we move from secrets to *systems*, modeled as information-theoretic channels that process secret information and possibly leak some of it to their public outputs. We develop a rich family of gain-function–leakage measures to quantify the damage a channel's leakage might cause, carefully considering the *operational significance* of such measures and developing theory that supports *robust* judgments about leakage.

In Part IV, we consider a more detailed model of systems as *programs* written in a simple probabilistic imperative programming language, enabling *compositional reasoning* about information leakage. Here, with assignment statements to program variables we can treat secrets that *change over time*. For that we introduce a mathematical technique that generalizes both channels (which leak secrets) and assignments (which update them). The technique is based on *Hidden Markov Models*.

Finally, in Part V we present a number of *case studies* showing how one can apply quantitative–information-flow analysis to many interesting realistic scenarios — including anonymity protocols, side-channel attacks on cryptography, voting protocols, and even differential privacy in statistical databases. Those chapters are intended to be somewhat self-contained, and readers interested in applications might wish to browse through them early.

Details of presentation

We sometimes format a definition, theorem, or paragraph in a box to give it greater visual prominence, as we have done in this paragraph. Our intent in doing that is to express our judgments, necessarily subjective, about which things are particularly significant or interesting.

The main text has been kept essentially free of literature citations and historical remarks — instead they are collected in a final section “Chapter Notes” for each chapter. The bibliography is, similarly, organized chapter by chapter.

Cited authors can be found alphabetically in the index, where they appear within square brackets, for example “[Claude E. Shannon]”. A glossary appears just before the index, and its entries are in order of first occurrence in the main text. The entry usually reads “*see something*”, without a page number, in the hope that the something on its own will be enough to jog the memory. If it isn’t, the index entry for “something” itself should be consulted to get a page reference.

Possible usage as a textbook

We have used draft chapters from Parts I, II, and III in a master’s-level course on the foundations of cybersecurity that also included extensive coverage of cryptography.

For a full-semester course, we envisage that a course based on Parts I, II, and III and selected chapters from Part V could be taught at both the advanced undergraduate and master’s levels. Part IV is more advanced mathematically, and is probably more suitable for doctoral students.

To facilitate the use of the book as a course textbook, we have included a section of *Exercises* at the end of most chapters. Solutions to these exercises are available to qualified instructors.

Language issues

Turning finally to questions of language: we come from six different countries (Brazil, Greece, the United Kingdom, Australia, Italy, and the United States) — which had the advantage that the sun never set on this book’s preparation: at all times at least one of us could be found hard at work on it. But such diversity also raises issues of spelling and usage. For the sake of consistency we have made an essentially arbitrary choice to follow American conventions throughout.

Also, with respect to the thorny question of personal pronouns, we have chosen to refer to the *defender* (i.e. the person or entity trying to protect sensitive information) as “he” or “him”, to the *adversary* as “she” or “her”, and to the *authors* and *readers* of this book as “we” or “us”. When there are several points of view, for example in multi-party protocols, we will occasionally use the neuter “it”. While assigning genders to the defender and adversary is of course arbitrary (and some readers might indeed prefer the opposite assignment), it has the advantages of avoiding the syntactic awkwardness of “he or she” and, more importantly, of enabling us to write with greater clarity and precision.

Acknowledgments

Our many collaborators have made profound contributions to our understanding of quantitative information flow — and we are particularly grateful to Arthur Américo, Miguel Andrés, Nicolás Bordenabe, Chris Chen, Michael R. Clarkson, Pierpaolo Degano, Kai Engelhardt, Barbara Espinoza, Natasha Fernandes, Jeremy Gibbons, Michael Hicks, Yusuke Kawamoto, Boris Köpf, Piotr Mardziel, Larissa Meinicke, Ziyuan Meng, Tahiry Rabehaja, Andre Scedrov, Fred B. Schneider, Tom Schrijvers, David M. Smith, Marco Stronati, and Roland Wen.

The authors are grateful for support from Digiteo and the Inria équipe associée Princess. Also, Mário S. Alvim was supported by the Computer Science Department at Universidade Federal de Minas Gerais (DCC/UFMG), by the National Council for Scientific and Technological Development (CNPq), by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES), and by the Fundação de Amparo à Pesquisa de Minas Gerais (FAPEMIG). Konstantinos Chatzikokolakis was supported by the Centre national de la recherche scientifique (CNRS), by the Institut national de recherche en sciences et technologies du numérique (Inria), and by the Department of Informatics and Telecommunications of the National and Kapodistrian University of Athens. Annabelle McIver was supported by the Department of Computing at Macquarie University and the Optus Macquarie Cyber Security Hub, Carroll Morgan by the Trustworthy Systems Group of CSIRO's Data61 and the School of Engineering and Computer Science at the University of New South Wales, and both of them by the Australian Research Council and the Information Security Group at ETH Zürich. Catuscia Palamidessi was supported by the Institut national de recherche en sciences et technologies du numérique (Inria), by her ERC grant HYPATIA and by the ANR project REPAS. Geoffrey Smith was supported by the School of Computing and Information Sciences at Florida International University and by the National Science Foundation under grant CNS-1116318.

Belo Horizonte
Athens
Sydney
Sydney
Paris
Miami
April 2020

Mário S. Alvim
Konstantinos Chatzikokolakis
Annabelle McIver
Carroll Morgan
Catuscia Palamidessi
Geoffrey Smith

Contents

Preface	vii
I Motivation	1
1 Introduction	3
1.1 A first discussion of information leakage	5
1.1.1 Secrets	5
1.1.2 Bayes vulnerability	5
1.1.3 Deterministic channels	6
1.1.4 Posterior distributions and hyper-distributions	7
1.1.5 Posterior Bayes vulnerability	8
1.1.6 Quantifying leakage	9
1.2 Looking ahead	10
1.3 Exercises	11
1.4 Chapter notes	12
II Secrets and How to Measure Them	15
2 Modeling secrets	17
2.1 Secrets and probability distributions	17
2.2 Shannon entropy	18
2.3 Bayes vulnerability	20
2.4 A more general view	21
2.5 Exercises	22
2.6 Chapter notes	22
3 On g-vulnerability	25
3.1 Basic definitions	25
3.1.1 Graphing g -vulnerability	27
3.2 A catalog of gain functions	29
3.2.1 The identity gain function	29
3.2.2 Gain functions induced from distance functions	30
3.2.3 Binary gain functions	31
3.2.4 Gain functions for a password database	33

3.2.5	A gain function that penalizes wrong guesses	34
3.2.6	A gain function for a medical diagnosis scenario	35
3.2.7	A loss function that gives guessing entropy	35
3.2.8	A loss function that gives Shannon entropy	37
3.3	Classes of gain functions	39
3.3.1	Finite-valued, non-negative vulnerabilities: the class $\mathbb{G}\mathcal{X}$	39
3.3.2	Finitely many actions: $\mathbb{G}^{\text{fin}}\mathcal{X}$	40
3.3.3	Non-negative gain functions: $\mathbb{G}^+\mathcal{X}$	40
3.3.4	One-bounded gain functions: $\mathbb{G}^\dagger\mathcal{X}$	41
3.4	Mathematical properties	41
3.4.1	Gain function algebra	42
3.5	On “absolute” versus “relative” security	43
3.6	Exercises	44
3.7	Chapter notes	44
III	Channels and Information Leakage	47
4	Channels	49
4.1	Channel matrices	49
4.2	The effect of a channel on the adversary’s knowledge	51
4.3	From joint distributions to hyper-distributions	54
4.4	Abstract channels	57
4.5	More on abstract channels	61
4.6	A first look at channel compositions	63
4.6.1	Convex combinations of channels	63
4.6.2	Cascading and the Data-Processing Inequality	64
4.7	Exercises	65
4.8	Chapter notes	66
5	Posterior vulnerability and leakage	71
5.1	Posterior g -vulnerability and its basic properties	71
5.2	Multiplicative and additive g -leakage	80
5.3	A closer look at posterior Bayes vulnerability and Bayes leakage	82
5.4	Measuring leakage with Shannon entropy	84
5.5	More properties of posterior g -vulnerability and g -leakage	86
5.5.1	A matrix-based formulation of posterior g -vulnerability	86
5.5.2	A trace-based formulation of posterior g -vulnerability	87
5.5.3	A linear-programming formulation	90
5.6	Example channels and their leakage	91
5.7	Max-case posterior g -vulnerability	93
5.8	Exercises	94
5.9	Chapter notes	97
6	Robustness	101
6.1	The need for robustness	101
6.2	Approaches to robustness	103
6.3	Exercises	103
6.4	Chapter notes	103

7	Capacity	107
7.1	Multiplicative Bayes capacity	107
7.2	Additive Bayes capacity	111
7.3	General capacities	116
7.4	Multiplicative capacities	117
7.4.1	Fixed g , maximize over π	117
7.4.2	Fixed π , maximize over g	118
7.4.3	Maximize over both g and π	119
7.5	Additive capacities	119
7.5.1	Fixed g , maximize over π	119
7.5.2	Fixed π , maximize over g	120
7.5.3	Maximize over both g and π	123
7.6	Obtaining bounds on leakage	124
7.6.1	The additive miracle theorem	124
7.6.2	Improved miracle bounds	124
7.6.3	Examples	125
7.7	Exercises	127
7.8	Chapter notes	127
8	Composition of channels	131
8.1	Compositions of (concrete) channel matrices	131
8.1.1	Parallel composition	132
8.1.2	External fixed-probability choice	133
8.1.3	External conditional choice	134
8.1.4	External (general) probabilistic choice	135
8.1.5	Internal fixed-probability choice	136
8.1.6	Internal conditional choice	137
8.1.7	Internal (general) probabilistic choice	137
8.1.8	Cascading	137
8.2	Compositions of abstract channels	138
8.2.1	The issue of compositionality	138
8.2.2	Parallel composition	139
8.2.3	External fixed-probability choice	139
8.2.4	External conditional choice	140
8.2.5	External (general) probabilistic choice	140
8.2.6	The internal choices, and cascading	140
8.3	Exercises	142
8.4	Chapter notes	143
9	Refinement	147
9.1	Refinement: for the <i>customer</i> ; for the <i>developer</i>	147
9.2	Structural refinement: the developer's point of view	148
9.2.1	Structural refinement for deterministic channels	148
9.2.2	Structural refinement for probabilistic channels	150
9.3	Testing refinement: the customer's point of view	152
9.4	Soundness of structural refinement	153
9.5	Completeness of structural refinement: the Coriaceous theorem	154
9.6	The structure of abstract channels under refinement	157
9.7	Refinement and monotonicity	159
9.7.1	Compositionality for contexts	159
9.7.2	Monotonicity with respect to refinement	160

9.8	Why does refinement (\sqsubseteq) have to be so complicated?	160
9.8.1	Who gets to define refinement, anyway?	160
9.8.2	A subjective argument: keeping the customer satisfied	162
9.8.3	An objective argument: compositional closure	164
9.9	Capacity is unsuitable as a criterion for refinement	166
9.10	Exercises	167
9.11	Chapter notes	167
10	The Dalenius perspective	171
10.1	Dalenius scenarios	172
10.2	Compositional closure for Dalenius contexts	175
10.2.1	Safety and necessity with respect to Dalenius contexts	175
10.2.2	Justifying refinement: an example	176
10.3	Bounding Dalenius leakage	177
10.4	Chapter notes	179
11	Axiomatics	183
11.1	An axiomatic view of vulnerability	183
11.2	Axiomatization of prior vulnerabilities	185
11.2.1	Soundness and completeness of V_g with respect to continuous, convex functions	186
11.3	Axiomatization of posterior vulnerabilities	188
11.3.1	Possible definitions of posterior vulnerabilities	189
11.4	Applications of axiomatization to understanding leakage measures	197
11.5	Chapter notes	199
12	The geometry of hypers, gains and losses	205
12.1	Barycentric representation of gain/loss functions	208
12.2	Barycentric representation of hypers and their refinement	210
12.3	Primitive hyper-distributions and their refinements	213
12.4	Hyper-distributions are not a lattice under refinement	216
12.5	A geometric proof of antisymmetry of refinement	218
12.6	Exercises	220
12.7	Chapter notes	220
IV	Information Leakage in Sequential Programs	223
13	Quantitative information flow in sequential computer programs	225
13.1	Markovs don't leak; and channels don't update	226
13.2	Specifications and implementations: a review	228
13.2.1	When is one <i>program</i> better than another, and why?	228
13.2.2	When is one <i>channel</i> better than another, and why?	229
13.2.3	Programs and channels together: what is "better" for both?	230
13.3	Aligning functional refinement with information-flow refinement	230
13.3.1	Generalizing Hoare logic for probability	230
13.3.2	Using loss functions	231
13.3.3	Refinement in general	232
13.3.4	Initial-final correlations, and Dalenius	233
13.4	Larger information-flow-aware programs	235
13.4.1	Sequential composition	235
13.4.2	On the terms <i>prior</i> , <i>posterior</i> , <i>initial</i> and <i>final</i>	240

13.4.3	Conditionals	241
13.4.4	The power of the adversary: <i>gedanken</i> experiments	242
13.4.5	Iteration	243
13.5	Syntax for probabilistic choice	243
13.6	Summary	246
13.7	Exercises	246
13.8	Chapter notes	247
14	Hidden-Markov modeling of <i>QIF</i> in sequential programs	255
14.1	<i>Concrete</i> Hidden Markov Models	255
14.1.1	<i>A priori</i> versus <i>a posteriori</i> reasoning — in more detail	257
14.2	Operations on and specializations of concrete <i>HMM</i> 's	258
14.2.1	Pure-channel and pure-markov <i>HMM</i> 's	258
14.2.2	Sequential composition of concrete <i>HMM</i> 's	258
14.2.3	General (concrete) <i>HMM</i> 's	260
14.3	<i>Abstract</i> Hidden Markov Models	260
14.3.1	Sequential (Kleisli) composition of abstract <i>HMM</i> 's	261
14.4	Syntax and abstract- <i>HMM</i> semantics of <i>QIF</i> -programs	264
14.4.1	Probabilistic assignment	264
14.4.2	Information flow via channels: leaking with PRINT	265
14.4.3	External probabilistic choice	266
14.4.4	(Internal probabilistic choice)	267
14.4.5	Sequential composition	268
14.4.6	Conditional	268
14.4.7	Iteration	268
14.4.8	Local variables	268
14.5	Leaks caused by conditionals and by external choice	270
14.6	Examples of small <i>QIF</i> programs	272
14.6.1	First example: Bertrand's Boxes	272
14.6.2	Second example: Goldfish or piraña?	274
14.6.3	Third example: Repeated independent runs	275
14.7	Underlying and unifying structures: a summary	275
14.8	Exercises	278
14.9	Chapter notes	279
15	Program algebra for <i>QIF</i>	283
15.1	Semantics, logic, and program algebra	283
15.2	Static visibility declarations; multiple variables	284
15.3	Simple examples of program derivations in <i>QIF</i>	286
15.3.1	The Encryption Lemma	286
15.3.2	From qualitative proofs to quantitative proofs	287
15.3.3	The One-Time Pad	287
15.4	Algebraic rules for reordering statements	290
15.5	Larger example 1: Oblivious Transfer	291
15.6	Larger example 2: Two-party conjunction, or <i>The Lovers' protocol</i>	296
15.7	Sub-protocols and declassification	298
15.8	Refinement and quantitative analyses	298
15.9	Exercises	301
15.10	Chapter notes	304

16 Iteration and nontermination	307
16.1 Why iteration is “different”	307
16.2 Classical nontermination	307
16.3 Nontermination for markovs and channels	308
16.3.1 Nontermination for markovs	308
16.3.2 Nontermination for channels	310
16.3.3 Applying abstract channels and markovs to sub-hypers	310
16.3.4 The semantic model for nontermination	311
16.4 The algebra of nontermination in <i>QIF</i>	311
16.5 A refinement order on <i>sub</i> -hyper-distributions	313
16.6 From nontermination to termination	316
16.7 Example of (certain) termination: how to design a password checker	317
16.8 A taxonomy of refinement orders	319
16.9 Exercises	321
16.10 Chapter notes	321
17 A demonic lattice of information	325
17.1 A <i>deterministic</i> lattice of information — the original	325
17.1.1 Historical introduction, intuition and abstraction	325
17.1.2 Structural definition of refinement for deterministic channels	328
17.1.3 Testing, soundness and completeness: deterministic	329
17.2 Our probabilistic partial order	330
17.3 Basic structure of the demonic lattice	331
17.4 Examples of demonically nondeterministic channels	334
17.5 Testing, soundness and completeness: demonic	336
17.6 A reformulation of demonic testing	337
17.7 Reduced demonic channels	339
17.8 Compositional closure	339
17.9 “Weakest pre-tests” and source-level reasoning	342
17.10 Exercises	345
17.11 Chapter notes	346
V Applications	351
18 The Crowds protocol	353
18.1 Introduction to Crowds, and its purpose	353
18.2 Modeling the Crowds protocol	354
18.3 Bayes vulnerability and Bayes leakage	357
18.4 Explanation of the paradox	358
18.4.1 Modified Crowds	358
18.4.2 Vulnerability of the original protocol	359
18.5 Why φ matters, even for uniform priors	360
18.5.1 Probable innocence as no lion leakage	361
18.6 Refinement: increasing φ is always safe	361
18.7 Multiple paths	363
18.7.1 Paths recreated by the initiator	363
18.7.2 Paths repaired by the last working node	364
18.7.3 Multiple detections and deviating from the protocol	365
18.8 Exercises	365
18.9 Chapter notes	366

19	Timing attacks on blinded and bucketed cryptography	369
19.1	Cryptographic background	369
19.2	A first leakage bound	370
19.3	A better leakage bound	372
19.4	Analytic results about $cap_b(n)$	374
19.5	Analytic proofs	378
19.6	Another proof of Theorem 19.5	384
19.7	Chapter notes	385
20	Defense against side channels	389
20.1	Evaluating a defense against side channels	389
20.2	<i>QIF</i> exploration of the fast-exponentiation algorithm	391
20.2.1	Cost/benefit analysis	394
20.3	Chapter notes	395
21	Multi-party computation: The Three Judges protocol	399
21.1	Introduction to The Three Judges	400
21.2	Developing an implementation of the Three Judges	401
21.2.1	First attempt	401
21.2.2	Second development attempt (sketch)	402
21.2.3	Successful development	403
21.2.4	Two-party exclusive-or	405
21.2.5	Summary	406
21.3	Exercises	409
21.4	Chapter notes	409
22	Voting systems	413
22.1	Elections and privacy risks	413
22.2	An illustrative and simplified <i>QIF</i> model for elections	414
22.2.1	The tallying	414
22.2.2	The casting	415
22.2.3	The Dalenius perspective: casting then tallying	416
22.3	Election by simple majority: first past the post	417
22.3.1	<i>QIF</i> channels for simple-majority elections: two examples	417
22.4	Election by preferences: instant run-off	418
22.4.1	<i>QIF</i> channels for instant-run-off elections: two examples	419
22.5	Gain functions for privacy of elections: a first example	419
22.6	The effect of small electorates in general	421
22.7	Case studies of small-electorate impact	422
22.7.1	First past the post, in small electorates	422
22.7.2	Instant run-off in small electorates	426
22.8	Chapter notes	429
23	Differential privacy	433
23.1	Notation and definition	434
23.2	Mechanisms as information-theoretic channels	435
23.3	The relation between differential privacy and multiplicative g -leakage	436
23.3.1	Bounds on leakage do not imply differential privacy	438
23.4	Exercises	439
23.5	Chapter notes	441
	Glossary and Index	445

List of definitions, theorems, examples, *etc.*

Theorem 1.1	8	Example 4.17	62
Corollary 1.2	9	Definition 4.18	64
Definition 2.1	17	Example 5.1	71
Conjecture 2.2	20	Definition 5.2	72
Definition 2.3	20	Example 5.3	73
Definition 3.1	25	Example 5.4	73
Definition 3.2	26	Example 5.5	75
Example 3.3	26	Theorem 5.6	77
Definition 3.4	27	Theorem 5.7	78
Definition 3.5	30	Theorem 5.8	78
Theorem 3.6	30	Theorem 5.9	79
Definition 3.7	30	Theorem 5.10	79
Definition 3.8	31	Definition 5.11	80
Definition 3.9	39	Theorem 5.12	80
Definition 3.10	40	Theorem 5.13	81
Definition 3.11	40	Example 5.14	81
Definition 3.12	41	Theorem 5.15	83
Theorem 3.13	41	Example 5.16	83
Theorem 3.14	42	Theorem 5.17	84
Definition 4.1	50	Theorem 5.18	86
Example 4.2	52	Example 5.19	87
Theorem 4.3	53	Theorem 5.20	87
Example 4.4	56	Definition 5.21	87
Definition 4.5	56	Lemma 5.22	88
Definition 4.6	57	Theorem 5.23	88
Definition 4.7	57	Theorem 5.24	89
Corollary 4.8	58	Example 5.25	89
Definition 4.9	59	Example 5.26	89
Theorem 4.10	59	Algorithm 5.27	90
Corollary 4.11	59	Algorithm 5.28	91
Example 4.12	59	Definition 5.29	93
Definition 4.13	60	Theorem 5.30	94
Definition 4.14	61	Theorem 5.31	94
Example 4.15	61	Definition 7.1	107
Theorem 4.16	62	Theorem 7.2	108

Corollary 7.3	108	Example 9.15	158
Corollary 7.4	108	Definition 9.16	160
Theorem 7.5	109	Definition 9.17	162
Example 7.6	109	Example 9.18	163
Theorem 7.7	110	Definition 9.19	165
Theorem 7.8	110	Example 9.20	166
Definition 7.9	111	Definition 10.1	173
Example 7.10	111	Theorem 10.2	173
Theorem 7.11	112	Example 10.3	173
Theorem 7.12	113	Theorem 10.4	175
Definition 7.13	116	Definition 10.5	177
Theorem 7.14	118	Theorem 10.6	178
Example 7.15	118	Theorem 10.7	178
Definition 7.16	121	Theorem 10.8	179
Definition 7.17	121	Definition 11.1	185
Lemma 7.18	121	Definition 11.2	185
Lemma 7.19	122	Definition 11.3	186
Theorem 7.20	122	Theorem 11.4	187
Theorem 7.21	122	Theorem 11.5	187
Example 7.22	123	Definition 11.6	188
Theorem 7.23	124	Definition 11.7	188
Theorem 7.24	125	Definition 11.8	189
Definition 8.1	132	Definition 11.9	190
Definition 8.2	133	Theorem 11.10	190
Definition 8.3	134	Theorem 11.11	191
Definition 8.4	135	Theorem 11.12	191
Definition 8.5	136	Lemma 11.13	191
Definition 8.6	138	Theorem 11.14	192
Definition 8.7	138	Definition 11.15	193
Definition 8.8	139	Theorem 11.16	193
Definition 8.9	139	Example 11.17	194
Definition 8.10	140	Theorem 11.18	194
Definition 8.11	140	Theorem 11.19	194
Definition 8.12	141	Corollary 11.20	195
Lemma 8.13	141	Definition 11.21	195
Definition 8.14	141	Theorem 11.22	196
Lemma 8.15	141	Example 11.23	196
Definition 9.1	149	Definition 12.1	210
Theorem 9.2	149	Lemma 12.2	213
Theorem 9.3	149	Definition 12.3	216
Theorem 9.4	150	Lemma 12.4	218
Definition 9.5	150	Corollary 12.5	218
Definition 9.6	151	Example 12.6	220
Theorem 9.7	151	Definition 13.1	232
Example 9.8	151	Definition 14.1	258
Corollary 9.9	152	Definition 14.2	260
Definition 9.10	152	Definition 14.3	262
Theorem 9.11	153	Theorem 14.4	263
Theorem 9.12	155	Definition 16.1	309
Theorem 9.13	156	Definition 16.2	313
Theorem 9.14	157	Definition 16.3	314

Definition 16.4	314	Theorem 18.3	363
Lemma 16.5	315	Theorem 18.4	363
Corollary 16.6	315	Theorem 19.1	371
Theorem 16.7	315	Definition 19.2	372
Theorem 16.8	316	Theorem 19.3	373
Definition 17.1	331	Theorem 19.4	375
Definition 17.2	332	Theorem 19.5	376
Definition 17.3	332	Theorem 19.6	377
Definition 17.4	333	Theorem 19.7	377
Definition 17.5	333	Theorem 19.8	377
Lemma 17.6	333	Definition 22.1	419
Lemma 17.7	333	Definition 22.2	420
Lemma 17.8	333	Definition 22.3	420
Lemma 17.9	334	Definition 22.4	420
Theorem 17.10	337	Definition 22.5	421
Definition 17.11	337	Definition 22.6	422
Lemma 17.12	338	Theorem 22.7	423
Lemma 17.13	339	Theorem 22.8	423
Definition 17.14	339	Definition 23.1	435
Lemma 17.15	339	Theorem 23.2	437
Definition 17.16	340	Corollary 23.3	437
Definition 17.17	340	Theorem 23.4	437
Theorem 17.18	341	Example 23.5	438
Definition 18.1	354	Example 23.6	439
Theorem 18.2	361		

List of figures

Figure 3.1	28	Figure 17.2	327
Figure 3.2	29	Figure 17.3	327
Figure 3.3	35	Figure 17.4	329
Figure 3.4	36	Figure 17.5	329
Figure 3.5	37	Figure 17.6	330
Figure 4.1	53	Figure 17.7	343
Figure 5.1	75	Figure 18.1	354
Figure 5.2	76	Figure 18.2	357
Figure 5.3	77	Figure 18.3	361
Figure 5.4	82	Figure 19.1	373
Figure 9.1	148	Figure 19.2	374
Figure 9.2	156	Figure 19.3	375
Figure 9.3	162	Figure 20.1	390
Figure 11.1	190	Figure 20.2	391
Figure 11.2	193	Figure 20.3	392
Figure 11.3	198	Figure 20.4	393
Figure 12.1	206	Figure 21.1	406
Figure 12.2	207	Figure 21.2	407
Figure 12.3	211	Figure 21.3	408
Figure 12.4	211	Figure 22.1	418
Figure 12.5	212	Figure 22.2	424
Figure 12.6	212	Figure 22.3	424
Figure 12.7	214	Figure 22.4	425
Figure 12.8	215	Figure 22.5	426
Figure 12.9	217	Figure 22.6	427
Figure 12.10	217	Figure 22.7	428
Figure 12.11	219	Figure 22.8	428
Figure 14.1	256	Figure 22.9	429
Figure 14.2	259	Figure 23.1	435
Figure 14.3	263	Figure 23.2	436
Figure 15.1	295	Figure 23.3	438
Figure 15.2	299	Figure 23.4	439
Figure 17.1	326		

List of exercises

Exercise 1.1	11	Exercise 9.4	167
Exercise 1.2	11	Exercise 9.5	167
Exercise 2.1	22	Exercise 9.6	167
Exercise 3.1	44	Exercise 9.7	167
Exercise 3.2	44	Exercise 9.8	167
Exercise 4.1	65	Exercise 9.9	167
Exercise 4.2	66	Exercise 9.10	167
Exercise 4.3	66	Exercise 12.1	220
Exercise 4.4	66	Exercise 13.1	246
Exercise 5.1	94	Exercise 13.2	246
Exercise 5.2	94	Exercise 13.3	247
Exercise 5.3	94	Exercise 13.4	247
Exercise 5.4	95	Exercise 14.1	278
Exercise 5.5	96	Exercise 14.2	278
Exercise 5.6	96	Exercise 14.3	279
Exercise 5.7	96	Exercise 14.4	279
Exercise 6.1	103	Exercise 14.5	279
Exercise 7.1	127	Exercise 15.1	301
Exercise 7.2	127	Exercise 15.2	301
Exercise 7.3	127	Exercise 15.3	302
Exercise 7.4	127	Exercise 15.4	302
Exercise 7.5	127	Exercise 15.5	302
Exercise 8.1	142	Exercise 15.6	303
Exercise 8.2	142	Exercise 15.7	303
Exercise 8.3	142	Exercise 15.8	303
Exercise 8.4	142	Exercise 15.9	303
Exercise 8.5	143	Exercise 15.10	303
Exercise 8.6	143	Exercise 15.11	303
Exercise 8.7	143	Exercise 15.12	303
Exercise 8.8	143	Exercise 15.13	303
Exercise 8.9	143	Exercise 16.1	321
Exercise 8.10	143	Exercise 16.2	321
Exercise 8.11	143	Exercise 16.3	321
Exercise 8.12	143	Exercise 16.4	321
Exercise 9.1	167	Exercise 16.5	321
Exercise 9.2	167	Exercise 17.1	345
Exercise 9.3	167	Exercise 17.2	345

List of exercises

Exercise 17.3	345	Exercise 18.1	365
Exercise 17.4	345	Exercise 18.2	365
Exercise 17.5	345	Exercise 21.1	409
Exercise 17.6	345	Exercise 21.2	409
Exercise 17.7	345	Exercise 21.3	409
Exercise 17.8	345	Exercise 21.4	409
Exercise 17.9	346	Exercise 23.1	439
Exercise 17.10	346	Exercise 23.2	440
Exercise 17.11	346	Exercise 23.3	440
Exercise 17.12	346	Exercise 23.4	440
Exercise 17.13	346		