Automated Software Diversity

Synthesis Lectures on Information Security, Privacy, & Trust

Editors
Elisa Bertino, ***Purdue University***
Ravi Sandhu, ***University of Texas at Austin***

The Synthesis Lectures Series on Information Security, Privacy, and Trust publishes 50- to 100-page publications on topics pertaining to all aspects of the theory and practice of Information Security, Privacy, and Trust. The scope largely follows the purview of premier computer security research journals such as ACM Transactions on Information and System Security, IEEE Transactions on Dependable and Secure Computing and Journal of Cryptology, and premier research conferences, such as ACM CCS, ACM SACMAT, ACM AsiaCCS, ACM CODASPY, IEEE Security and Privacy, IEEE Computer Security Foundations, ACSAC, ESORICS, Crypto, EuroCrypt and AsiaCrypt. In addition to the research topics typically covered in such journals and conferences, the series also solicits lectures on legal, policy, social, business, and economic issues addressed to a technical audience of scientists and engineers. Lectures on significant industry developments by leading practitioners are also solicited.

Automated Software Diversity
Per Larsen, Stefan Brunthaler, Lucas Davi, Ahmad-Reza Sadeghi, and Michael Franz
2015

Trust in Social Media
Jiliang Tang and Huan Liu
2015

Physically Unclonable Functions (PUFs): Applications, Models, and Future Directions
Christian Wachsmann and Ahmad-Reza Sadeghi
2014

Usable Security: History, Themes, and Challenges
Simson Garfinkel and Heather Richter Lipford
2014

Reversible Digital Watermarking: Theory and Practices
Ruchira Naskar and Rajat Subhra Chakraborty
2014

Mobile Platform Security
N. Asokan, Lucas Davi, Alexandra Dmitrienko, Stephan Heuser, Kari Kostiainen, Elena Reshetova, and Ahmad-Reza Sadeghi
2013

Security and Trust in Online Social Networks
Barbara Carminati, Elena Ferrari, and Marco Viviani
2013

RFID Security and Privacy
Yingjiu Li, Robert H. Deng, and Elisa Bertino
2013

Hardware Malware
Christian Krieg, Adrian Dabrowski, Heidelinde Hobel, Katharina Krombholz, and Edgar Weippl
2013

Private Information Retrieval
Xun Yi, Russell Paulet, and Elisa Bertino
2013

Privacy for Location-based Services
Gabriel Ghinita
2013

Enhancing Information Security and Privacy by Combining Biometrics with Cryptography
Sanjay G. Kanade, Dijana Petrovska-Delacrétaz, and Bernadette Dorizzi
2012

Analysis Techniques for Information Security
Anupam Datta, Somesh Jha, Ninghui Li, David Melski, and Thomas Reps
2010

Operating System Security
Trent Jaeger
2008

© Springer Nature Switzerland AG 2022
Reprint of original edition © Morgan & Claypool 2016

All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted in any form or by any means—electronic, mechanical, photocopy, recording, or any other except for brief quotations in printed reviews, without the prior permission of the publisher.

Automated Software Diversity

Per Larsen, Stefan Brunthaler, Lucas Davi, Ahmad-Reza Sadeghi, and Michael Franz

ISBN: 978-3-031-01218-1 paperback
ISBN: 978-3-031-02346-0 ebook

DOI 10.1007/978-3-031-02346-0

A Publication in the Springer series
SYNTHESIS LECTURES ON INFORMATION SECURITY, PRIVACY, & TRUST

Lecture #14
Series Editors: Elisa Bertino, *Purdue University*
Ravi Sandhu, *University of Texas at Austin*
Series ISSN
Print 1945-9742 Electronic 1945-9750

Automated Software Diversity

Per Larsen
Immunant, Inc., USA

Stefan Brunthaler
SBA Research, Vienna, Austria

Lucas Davi
Technische Universität Darmstadt, Germany

Ahmad-Reza Sadeghi
Technische Universität Darmstadt, Germany

Michael Franz
University of California, Irvine, USA

SYNTHESIS LECTURES ON INFORMATION SECURITY, PRIVACY, & TRUST #14

ABSTRACT

Whereas user-facing applications are often written in modern languages, the firmware, operating system, support libraries, and virtual machines that underpin just about any modern computer system are still written in low-level languages that value flexibility and performance over convenience and safety. Programming errors in low-level code are often exploitable and can, in the worst case, give adversaries unfettered access to the compromised host system.

This book provides an introduction to and overview of automatic software diversity techniques that, in one way or another, use randomization to greatly increase the difficulty of exploiting the vast amounts of low-level code in existence. Diversity-based defenses are motivated by the observation that a single attack will fail against multiple targets with unique attack surfaces. We introduce the many, often complementary, ways that one can diversify attack surfaces and provide an accessible guide to more than two decades worth of research on the topic. We also discuss techniques used in conjunction with diversity to prevent accidental disclosure of randomized program aspects and present an in-depth case study of one of our own diversification solutions.

KEYWORDS

software diversity, code randomization, data randomization, information leaks, leakage resilience, code reuse, exploitation

Contents

Preface ix

Acknowledgments xi

1 Introduction 1

1.1 A Brief History of Program Randomization 2

1.2 Book Overview 3

2 Attacking and Defending 5

2.1 Taxonomy of Attacks 5

2.1.1 Memory Corruption Attacks 6

2.1.2 Information Leaks 7

2.1.3 Code Injection 8

2.1.4 Code Reuse 8

2.1.5 JIT Attacks 9

2.1.6 Program Tampering 10

2.1.7 Reverse Engineering 10

2.2 Taxonomy of Defenses 10

2.2.1 Enforcement-based Defenses 10

2.2.2 Program Integrity Monitors 11

2.2.3 Diversity-based Defenses 12

2.2.4 Program Obfuscation 12

3 What to Diversify 13

3.1 Instruction Level 13

3.2 Basic Block Level 14

3.3 Loop Level 16

3.4 Function Level 16

3.5 Program Level 17

3.6 System Level 19

4 When to Diversify . . . 25
4.1 The Software Life Cycle . . . 25
4.2 Quantifying the Impact of Diversity . . . 34
4.2.1 Security Impact . . . 34
4.2.2 Performance Impact . . . 35

5 Case Study: Compile-time Diversification . . . 39
5.1 System Description . . . 39
5.1.1 Inserting NOP Instructions . . . 39
5.1.2 Equivalent Instruction Substitution . . . 40
5.1.3 Instruction Scheduling . . . 41
5.2 Scalability of Compile-time Diversification . . . 41
5.2.1 Cloud-based Compilation . . . 41
5.2.2 Scalability is Practical . . . 43
5.3 Evaluating Diversification . . . 45
5.3.1 Assessing Diversification Efficiency . . . 45
5.3.2 Implementing Survivor . . . 46
5.4 Evaluating Security . . . 48
5.4.1 Frequently Surviving Gadgets . . . 48
5.4.2 Determining Optimal Compiler Parameters . . . 49

6 Information Leakage Resilience . . . 53

7 Advanced Topics . . . 59
7.1 Hybrid Approaches . . . 59
7.2 Error Reports and Patches . . . 60

Bibliography . . . 61

Authors' Biographies . . . 75

Preface

To err is human or so the Latin proverb goes. In spite of all technological progress since classical antiquity, human error has not been eliminated. While computer programmers have numerous tools at their disposal to help catch and correct mistakes, modern software development is so complex that residual errors inevitably remain as programs are released to end users. This is a book about the security implications of some such errors and a family of techniques—collectively known as software diversity—that prevent reliable exploitation of software flaws by making the target environment less predictable for an attacker.

Per Larsen, Stefan Brunthaler, Lucas Davi, Ahmad-Reza Sadeghi, and Michael Franz
December 2015

Acknowledgments

The authors thank Andrei Homescu, Stephen Crane, Todd Jackson, Steven Neisius, and Alen Stojanov for their contributions to the work presented in this book. Hamed Okhravi and Mathias Payer provided lots of constructive suggestions and feedback on draft manuscripts; thank you!

This material is based upon work partially supported by the Defense Advanced Research Projects Agency (DARPA) under contracts D11PC20024, N660001-1-2-4014, FA8750-15-C-0124, FA8750-15-C-0085, and N66001-13-C-4057 as well as gifts from Mozilla, Oracle, and Qualcomm. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the Defense Advanced Research Projects Agency (DARPA), its Contracting Agents, or any other agency of the U.S. Government.

Portions of this work are from Larsen et al. [76], copyright © 2014, IEEE.

Portions of this work are from Crane et al. [39], copyright © 2015, IEEE.

Portions of this work are from Homescu et al. [61], copyright © 2015, IEEE.

Per Larsen, Stefan Brunthaler, Lucas Davi, Ahmad-Reza Sadeghi, and Michael Franz
December 2015