Formal Methods

LNCS 11800

Maurice H. ter Beek
Annabelle McIver
José N. Oliveira (Eds.)

Formal Methods – The Next 30 Years

Third World Congress, FM 2019
Porto, Portugal, October 7–11, 2019
Proceedings

Springer

Lecture Notes in Computer Science 11800

Formal Methods

Subline of Lectures Notes in Computer Science

More information about this series at http://www.springer.com/series/7408

Maurice H. ter Beek · Annabelle McIver ·
José N. Oliveira (Eds.)

Formal Methods – The Next 30 Years

Third World Congress, FM 2019
Porto, Portugal, October 7–11, 2019
Proceedings

Springer

Editors
Maurice H. ter Beek
Consiglio Nazionale delle Ricerche
Pisa, Italy

Annabelle McIver
Macquarie University
Sydney, NSW, Australia

José N. Oliveira
University of Minho
Braga, Portugal

ISSN 0302-9743 ISSN 1611-3349 (electronic)
Lecture Notes in Computer Science
ISBN 978-3-030-30941-1 ISBN 978-3-030-30942-8 (eBook)
https://doi.org/10.1007/978-3-030-30942-8

LNCS Sublibrary: SL2 – Programming and Software Engineering

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

Preface

This volume contains the papers presented at the 23rd Symposium on Formal Methods (FM 2019), held in Porto, Portugal, in the form of the Third World Congress on Formal Methods, during October 7–11, 2019. These proceedings also contain five papers selected by the Program Committee (PC) of the Industry Day (I-Day).

FM 2019 was organized under the auspices of Formal Methods Europe (FME), an independent association whose aim is to stimulate the use of, and research on, formal methods for software development. It has been more than 30 years since the first VDM symposium in 1987 brought together researchers with the common goal of creating methods to produce high-quality software based on rigor and reason. Since then the diversity and complexity of computer technology has changed enormously and the formal methods community has stepped up to the challenges those changes brought by adapting, generalizing, and improving the models and analysis techniques that were the focus of that first symposium. The theme for FM 2019, "The Next 30 Years," was a reflection on how far the community has come and the lessons we can learn for understanding and developing the best software for future technologies.

To reflect the fact that it has been 20 years since FM 1999 in Toulouse and 10 years since FM 2009 in Eindhoven, FM 2019 was organized as a World Congress, and we composed a PC of renowned scientists from 42 different countries spread across all continents except for Antarctica. We originally received a stunning total of 185 abstract submissions, which unfortunately resulted in 'only' 129 paper submissions from 36 different countries. Each submission went through a rigorous review process in which 95% of the papers were reviewed by four PC members. Following an in-depth discussion phase lasting two weeks, we selected 37 full papers and 2 short tool papers, an acceptance rate of 30%, for presentation during the symposium and inclusion in these proceedings. The symposium featured keynotes by Shriram Krishnamurthi (Brown University, USA), Erik Poll (Radboud University, The Netherlands), and June Andronick (CSIRO-Data61 and UNSW, Australia). We hereby thank these invited speakers for having accepted our invitation. The program also featured a Lucas Award and FME Fellowship Award Ceremony.

We are grateful to all involved in FM 2019. In particular the PC members and subreviewers for their accurate and timely reviewing, all authors for their submissions, and all attendees of the symposium for their participation. We also thank all the other committees (I-Day, Doctoral Symposium, Journal First Track, Workshops, and Tutorials), itemized on the following pages, and particularly the excellent local organization and publicity teams. In addition to FM 2019 they also managed the FM week consisting of another 8 conferences, 17 workshops, and 7 tutorials, as well as 'X', the secret project of a colloquium in honor of Stefania Gnesi based on a Festschrift to celebrate her 65th birthday.

We are very grateful to our platinum sponsors: Amazon Web Services (AWS), Google, and Sony; our gold sponsors: Springer, Semmle, ASML, and PT-FLAD Chair in Smart Cities & Smart Governance; our silver sponsors: Oracle Labs, Runtime Verification Inc., Standard Chartered, GMV, United Technologies Research Center (UTRC), and Efacec; our bronze sponsors i2S, Foundations of Perspicuous Software Systems Collaborative Research Center, and the Mathematical research center of the University of Porto (CMUP); and our basic sponsors: Natixis and Neadvance.

Finally, we thank Springer for publishing these proceedings in their FM subline and we acknowledge the support from EasyChair in assisting us in managing the complete process from submissions to these proceedings to the program.

August 2019

Maurice H. ter Beek
Annabelle McIver
José N. Oliveira

Organization

General Chair

José N. Oliveira	University of Minho and INESC TEC, Portugal

FM Program Chairs

Maurice H. ter Beek	ISTI–CNR, Italy
Annabelle McIver	Macquarie University, Australia

Industry Day Chairs

Joe Kiniry	Galois Inc., USA
Thierry Lecomte	ClearSy, France

Doctoral Symposium Chairs

Alexandra Silva	University College London, UK
Antónia Lopes	University of Lisbon, Portugal

Journal First Track Chair

Augusto Sampaio	Federal University of Pernambuco, Brazil

Workshop and Tutorial Chairs

Emil Sekerinski	McMaster University, Canada
Nelma Moreira	University of Porto, Portugal

FM Program Committee

Bernhard Aichernig	TU Graz, Austria
Elvira Albert	Complutense University of Madrid, Spain
María Alpuente	Polytechnic University of Valencia, Spain
Dalal Alrajeh	Imperial College, UK
Mário S. Alvim	Federal University of Minas Gerais, Brazil
June Andronick	CSIRO-Data61, Australia
Christel Baier	TU Dresden, Germany
Luís Barbosa	University of Minho and UN University, Portugal
Gilles Barthe	IMDEA Software Institute, Spain
Marcello Bersani	Polytechnic University of Milan, Italy
Gustavo Betarte	Tilsor SA and University of the Republic, Uruguay

Nikolaj Bjørner	Microsoft Research, USA
Frank de Boer	CWI, The Netherlands
Sergiy Bogomolov	Australian National University, Australia
Julien Brunel	ONERA, France
Néstor Cataño	Universidad del Norte, Colombia
Ana Cavalcanti	University of York, UK
Antonio Cerone	Nazarbayev University, Kazakhstan
Marsha Chechik	University of Toronto, Canada
David Chemouil	ONERA, France
Alessandro Cimatti	FBK–IRST, Italy
Alcino Cunha	University of Minho and INESC TEC, Portugal
Michael Dierkes	Rockwell Collins, France
Alessandro Fantechi	University of Florence, Italy
Carla Ferreira	New University of Lisbon, Portugal
João Ferreira	Teesside University, UK
José L. Fiadeiro	Royal Holloway University of London, UK
Marcelo Frias	Buenos Aires Institute of Technology, Argentina
Fatemeh Ghassemi	University of Tehran, Iran
Silvia Ghilezan	University of Novi Sad, Serbia
Stefania Gnesi	ISTI–CNR, Italy
Reiner Hähnle	TU Darmstadt, Germany
Osman Hasan	University of Sciences and Technology, Pakistan
Klaus Havelund	NASA Jet Propulsion Laboratory, USA
Anne Haxthausen	TU Denmark, Denmark
Ian Hayes	University of Queensland, Australia
Constance Heitmeyer	Naval Research Laboratory, USA
Jane Hillston	University of Edinburgh, UK
Thai Son Hoang	University of Southampton, UK
Zhenjiang Hu	National Institute of Informatics, Japan
Dang Van Hung	Vietnam National University, Vietnam
Atsushi Igarashi	Kyoto University, Japan
Suman Jana	Columbia University, USA
Ali Jaoua	Qatar University, Qatar
Einar Broch Johnsen	University of Oslo, Norway
Joost-Pieter Katoen	RWTH Aachen University, Germany
Laura Kovács	TU Vienna, Austria
Axel Legay	UCLouvain, Belgium
Gabriele Lenzini	University of Luxembourg, Luxembourg
Yang Liu	Nanyang Technical University, Singapore
Alberto Lluch Lafuente	TU Denmark, Denmark
Malte Lochau	TU Darmstadt, Germany
Michele Loreti	University of Camerino, Italy
Anastasia Mavridou	NASA Ames, USA
Hernán Melgratti	University of Buenos Aires, Argentina
Sun Meng	Peking University, China
Dominique Méry	LORIA and University of Lorraine, France

Rosemary Monahan	Maynooth University, Ireland
Olfa Mosbahi	University of Carthage, Tunisia
Mohammad Mousavi	University of Leicester, UK
César Muñoz	NASA Langley, USA
Tim Nelson	Brown University, USA
Gethin Norman	University of Glasgow, UK
Colin O'Halloran	D-RisQ Software Systems, UK
Federico Olmedo	University of Chile, Chile
Gordon Pace	University of Malta, Malta
Jan Peleska	University of Bremen, Germany
Marielle Petit-Doche	Systerel, France
Alexandre Petrenko	Computer Research Institute of Montréal, Canada
Anna Philippou	University of Cyprus, Cyprus
Jorge Sousa Pinto	University of Minho and INESC TEC, Portugal
André Platzer	Carnegie Mellon University, USA
Jaco van de Pol	Aarhus University, Denmark
Tahiry Rabehaja	Macquarie University, Australia
Steve Reeves	University of Waikato, New Zealand
Matteo Rossi	Polytechnic University of Milan, Italy
Augusto Sampaio	Federal University of Pernambuco, Brazil
Gerardo Schneider	Chalmers University of Gothenburg, Sweden
Daniel Schwartz Narbonne	Amazon Web Services, USA
Natasha Sharygina	University of Lugano, Switzerland
Nikolay Shilov	Innopolis University, Russia
Ana Sokolova	University of Salzburg, Austria
Marielle Stoelinga	University of Twente, The Netherlands
Jun Sun	University of Technology and Design, Singapore
Helen Treharne	University of Surrey, UK
Elena Troubitsyna	Äbo Akademi University, Finland
Tarmo Uustalu	Reykjavik University, Iceland
Andrea Vandin	TU Denmark, Denmark
R. Venkatesh	TCS Research, India
Erik de Vink	TU Eindhoven and CWI, The Netherlands
Willem Visser	Stellenbosch University, South Africa
Farn Wang	National Taiwan University, Taiwan
Bruce Watson	Stellenbosch University, South Africa
Tim Willemse	TU Eindhoven, The Netherlands
Kirsten Winter	University of Queensland, Australia
Jim Woodcock	University of York, UK
Lijun Zhang	Chinese Academy of Sciences, China

Additional Reviewers

Rui Abreu
Arthur Américo
Hugo Araujo
Myla Archer
Sepideh Asadi
Florent Avellaneda
Eduard Baranov
Davide Basile
Cláudio Belo Lourenço
Philipp Berger
František Blahoudek
Martin Blicha
Jean-Paul Bodeveix
Brandon Bohrer
Ioana Boureanu
Laura Bozzelli
Daniel Britten
James Brotherston
Richard Bubel
Doina Bucur
Juan Diego Campo
Laura Carnevali
Gustavo Carvalho
Davide Cavezza
Xiaohong Chen
Yu-Ting Chen
Robert Colvin
Jesús Correas Fernández
Silvano Dal Zilio
Carlos Diego Damasceno
Quoc Huy Do
Sebastian Ehmes
Santiago Escobar
Marco Faella
Paul Fiterau Brostean
Simon Foster
Maria João Frade
Maciej Gazda
Lorenzo Gheri
Eduardo Giménez
Pablo Gordillo
Gloria Gori
Friedrich Gretz
Jerry den Hartog
Raju Halder
Hossein Hojjat
Karel Horak
Zhe Hou
Thomas Hujsa
Andreas Humenberger
Antti Hyvarinen
Peter Häfner
Fabian Immler
Miguel Isabel
Shaista Jabeen
Phillip James
Seema Jehan
Saul Johnson
Violet Ka I Pun
Eduard Kamburjan
Minh-Thang Khuu
Sascha Klüppelholz
Dimitrios Kouzapas
Robbert Krebbers
Shrawan Kumar
Luca Laurenti
Maurice Laveaux
Corey Lewis
Jianlin Li
Yi Li
Yong Li
Ai Liu
Wanwei Liu
Martin Lukac
Carlos Luna
Lars Luthmann
Joshua Moerman
Hendrik Maarand
Kumar Madhukar
Shahar Maoz
Matteo Marescotti
Bojan Marinkovic
Paolo Masci
Mieke Massink
Franco Mazzanti
Larissa Meinicke
Alexandra Mendes
Stephan Merz
Ravindra Metta
Andrea Micheli
Stefan Mitsch
Alvaro Miyazawa
Carroll Morgan
Mariano Moscato
Toby Murray
David Müller
Koji Nakazawa
Pham Ngoc Hung
Omer Nguena-Timo
Hans de Nivelle
Quentin Peyras
Paul Piho
Danny Bøgsted Poulsen
James Power
Tim Quatmann
Jean-Baptiste Raclet
Markus Roggenbach
Guillermo Román-Díez
Jurriaan Rot
Albert Rubio
Enno Ruijters
Sebastian Ruland
David Sanan
Julia Sapiña
Andy Schürr
Ramy Shahin
Neeraj Singh
Andrew Sogokon
B. Srivathsan
Dominic Steinhöfel
Ivan Stojic
Sandro Stucki
Martin Tappler

Laura Titolo
Andrea Turrini
Ben Tyler
Evangelia Vanezi
Alicia Villanueva
Inna Vistbakka
Matthias Volk
Jingyi Wang
Shuling Wang
Markus Weckesser
Stephan Wesemeyer
Pengfei Yang
Haodong Yao

I-Day Program Committee

M. Antony Aiello	AdaCore, USA
Flemming Andersen	Galois Inc., USA
Stylianos Basagianni	United Technologies Research Centre, Ireland
Roderick Chapman	Protean Code Limited, UK
David Cok	GrammaTech, USA
Alessandro Fantechi	University of Florence, Italy
Chris Hawblitzel	Microsoft, USA
Peter Gorm Larsen	Aarhus University, Denmark
Michael Leuschel	University of Düsseldorf, Germany
Yannick Moy	AdaCore, France
Jan Peleska	Verified Systems International GmbH, Germany
Etienne Prun	ClearSy, France
Kenji Taguchi	CAV Technologies Co., Ltd., Japan
Stefano Tonetta	FBK–IRST, Italy
Daniel Zimmerman	Galois Inc., USA

DS Program Committee

Ana Cavalcanti	University of York, UK
André Platzer	Carnegie Mellon University, USA
Alessandro Fantechi	University of Florence, Italy
Carlo A. Furia	USI, Switzerland
Dalal Alrajeh	Imperial College, UK
Einar Broch Johnson	University of Oslo, Norway
Elvira Albert	Complutense University of Madrid, Spain
Jaco van de Pol	Aarhus University, Denmark
Matteo Rossi	Polytechnic University of Milan, Italy
Stefania Gnesi	ISTI-CNR, Italy
Stephan Merz	Inria, France

JFT Program Committee

Cliff Jones	University of Newcastle, UK
Manfred Broy	TU Munich, Germany

Organizing Committee

Luís Soares Barbosa	University of Minho and INESC TEC, Portugal
José Creissac Campos	University of Minho and INESC TEC, Portugal
João Pascoal Faria	University of Porto and INESC TEC, Portugal
Sara Fernandes	University of Minho and INESC TEC, Portugal
Luís Neves	Critical Software, Portugal
Ana Paiva	University of Porto and INESC TEC, Portugal

Local Organizers

Catarina Fernandes	University of Minho and INESC TEC, Portugal
Paula Rodrigues	INESC TEC, Portugal
Ana Rita Costa	INESC TEC, Portugal

Web Team

Francisco Neves	University of Minho and INESC TEC, Portugal
Rogério Pontes	University of Minho and INESC TEC, Portugal
Paula Rodrigues	INESC TEC, Portugal

FME Board

Ana Cavalcanti	University of York, UK
Lars-Henrik Eriksson	Uppsala University, Sweden
Stefania Gnesi	ISTI–CNR, Italy
Einar Broch Johnsen	University of Oslo, Norway
Nico Plat	Thanos, The Netherlands

Formal Methods for Security Functionality and for Secure Functionality (Invited Presentation)

Erik Poll

Digital Security group, Radboud University Nijmegen, The Netherlands
erikpoll@cs.ru.nl

With cyber security becoming a growing concern, it has naturally attracted the attention of researchers in formal methods. One recent success story here is TLS: the development of the new TLS 1.3 specification has gone hand-in-hand with efforts to verify security properties of formal models [5] and the development of a fully verified implementation [3]. Earlier well-known success stories in using formal methods for security are the verifications of operating system kernels or hypervisors, namely seL4 [7] and Microsoft's Hyper-V [10].

These examples – security protocols and OS kernels – are applications whose primary purpose is to provide security. It is natural to apply formal methods to such systems: they are by their very nature security-critical and they provide some security functionality that we can try to specify and verify.

However, we want *all* our systems to be secure, not just these security systems. There is an important difference between *secure* functionality and *security* functionality, or – given that most functionality and most security problems are down to software – between *software security* and *security software* [11]. Many, if not most, security problems arise in systems that have no specific security objective, say PDF viewers or video players, but which can still be hacked to provide attackers with unwanted functionality they can abuse.

Using formal methods to prove security is probably not on the cards of something as complex as a PDF viewer or video player. Just defining what it would mean for such a system to be secure is probably already infeasible. Still, formal methods can be useful, to prove the absence of certain types of security flaws or simply find security flaws. Successes here have been in the use of static analysis in source code analysers, e.g. tools like Fortify SCA that look for flaws in web applications and tools like Coverity that look for memory vulnerabilities in C(++) code. Another successful application of formal methods is the use of symbolic (or concolic) execution to generate test cases for security testing, as in SAGE [6] or, going one step further, not just automatically finding flaws but also automatically generating exploits, as in angr [16].

Downside of these approaches is that they are post-hoc and can only look for flaws in existing code. The *LangSec* paradigm [4, 9], on the other hand, provides ideas on how to prevent many security problems *by construction*. Key insights are that most security flaws occur in input handling and that there are several root causes in play here. Firstly, the input languages involved (e.g. file formats and network protocols) are complex, very expressive, and poorly, informally, specified. Secondly, there are *many* of these input languages, sometimes nested or stacked. Finally, parsers for these languages are typically hand-written, with parsing code scattered throughout the application code in so-called shotgun parsers [12]. With clearer, formal specifications of input languages and generated parser code much security misery could be avoided. (Recent initiatives in tools for parser generation here include Hammer [1] and Nail [2].) Given that formal languages and parser generation are some of the most basic and established formal methods around, it is a bit of an embarrassment to us as formal methods community that sloppy language specifications and hand-coded parsers should cause so many security problems.

Some security flaws in input handling are not so much caused by *buggy* parsing of inputs, but rather by the *unexpected* parsing of input [13]. Classic examples of this are command injection, SQL injection, and Cross-Site Scripting (XSS). Tell-tale sign that unwanted parsing of input may be happening in unexpected places is the heavy use of strings as data types [14].

Information or data flow analysis can be used to detect such flaws; indeed, this is a standard technique used in the source code analysis tools mentioned above. These flaws can also be prevented by construction, namely by using type systems. A recent example of this is the ‘Trusted Types’ browser API [8] by Google, where different types are used to track different kinds of data and different trust level of data to prevent XSS vulnerabilities, esp. the DOM-based XSS vulnerabilities that have proved so difficult to root out.

To conclude, formal methods cannot only be used to *prove* security of security-critical applications and components – i.e. the security software –, but they can be much more widely used to *improve* security by ruling out of the root causes behind security flaws in input handling, and do so by construction, and hence improve software security in general. Moreover, some very basic and lightweight formal methods can be used for this: methods that we teach – or should be teaching – our students in the first years of their Bachelor degree, such as regular expressions, finite state machines, grammars, and types. Indeed, in my own research I have been surprised to see how useful the simple notion of finite state machine for describing input sequences is to discover security flaws [15].

That we have not been able to get these basic techniques into common use does not say much for our success in transferring formal methods to software engineering practice. Still, looking at the bright side, it does suggest opportunities for improvement.

References

1. Anantharaman, P., Millian, M.C., Bratus, S., Patterson, M.L.: Input handling done right: building hardened parsers using language-theoretic security. In: Cybersecurity Development (SecDev), pp. 4–5. IEEE (2017)
2. Bangert, J., Zeldovich, N.: Nail: A practical tool for parsing and generating data formats. In: OSDI 2014, pp. 615–628. Usenix (2014)

3. Bhargavan, K., Blanchet, B., Kobeissi, N.: Verified models and reference implementations for the TLS 1.3 standard candidate. In: Security and Privacy (S&P 2017), pp. 483–502. IEEE (2017)
4. Bratus, S., Locasto, M.E., Patterson, M.L., Sassaman, L., Shubina, A.: Exploit programming: from buffer overflows to weird machines and theory of computation. Login, 13–21 (2011)
5. Cremers, C., Horvat, M., Hoyland, J., Scott, S., van der Merwe, T.: A comprehensive symbolic analysis of TLS 1.3. In: SIGSAC Conference on Computer and Communications Security (CCS 2017), pp. 1773–1788. ACM (2017)
6. Godefroid, P., Levin, M.Y., Molnar, D.: SAGE: Whitebox fuzzing for security testing. Commun. ACM **55**(3), 40–44 (2012)
7. Klein, G., et al.: seL4: Formal verification of an OS kernel. In: ACM SIGOPS, pp. 207–220. ACM (2009)
8. Kotowicz, K.: Trusted types help prevent cross-site scripting (2019). https://developers.google.com/web/updates/2019/02/trusted-types. blog
9. LangSec: Recognition, validation, and compositional correctness for real world security (2013). http://langsec.org/bof-handout.pdf. uSENIX Security BoF hand-out
10. Leinenbach, D., Santen, T.: Verifying the microsoft hyper-V hypervisor with VCC. In: Cavalcanti, A., Dams, D.R. (eds.) FM 2009, LNCS, vol. 5850, pp. 806–809. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-05089-3_51
11. McGraw, G.: Software security. IEEE Secur. Priv. **2**(2), 80–83 (2004)
12. Momot, F., Bratus, S., Hallberg, S.M., Patterson, M.L.: The seven turrets of Babel: a taxonomy of LangSec errors and how to expunge them. In: Cybersecurity Development (SecDev 2016), pp. 45–52. IEEE (2016)
13. Poll, E.: LangSec revisited: input security flaws of the second kind. In: Workshop on Language-Theoretic Security (LangSec 2018). IEEE (2018)
14. Poll, E.: Strings considered harmful. Login, **43**(4), 21–26 (2018)
15. Poll, E., de Ruiter, J., Schubert, A.: Protocol state machines and session languages: specification, implementation, and security flaws. In: Workshop on Language-Theoretic Security (LangSec 2015), pp. 125–133. IEEE (2015)
16. Shoshitaishvili, Y., et al.: SoK:(state of) the art of war: offensive techniques in binary analysis. In: Symposium on Security and Privacy (SP 2016), pp. 138–157. IEEE (2016)

Contents

Concurrency

Model Checking Circus

Model Checking

Analysis Techniques

Modelling Languages

Learning-Based Techniques and Applications

Refactoring and Reprogramming

I-Day Presentations