Journal of Computer Security 25 (2017) 319–321
DOI 10.3233/JCS-0559
IOS Press

Guest Editorial

Special issue on verified information flow security

Toby Murray [a,b,*], Andrei Sabelfeld [c] and Lujo Bauer [d]

[a] *School of Computing and Information Systems, University of Melbourne, Australia*
E-mail: toby.murray@unimelb.edu.au
[b] *Data61, CSIRO, Australia*
[c] *Department of Computer Science and Engineering, Chalmers University of Technology, Sweden*
E-mail: andrei@chalmers.se
[d] *Department of Electrical and Computer Engineering and Institute for Software Research, Carnegie Mellon University, PA, USA*
E-mail: lbauer@cmu.edu

1. Foreword

Information flow security has remained an active topic of research since the seminal work of Denning [4], Goguen and Meseguer [5] and their contemporaries. The past few years have seen the seeds of formal information flow security sown in the preceding three decades bear practical fruit. A number of real-world systems with formally verified guarantees of information flow security now exist. These systems serve as exemplars of verified security that span hardware [1], operating system microkernels [11] and virtualisation platforms [6], programming languages [7], mobile operating systems [9], web browsers [2,8], web applications [3,10] and distributed systems. The time is ripe to mark this success with this special issue of the Journal of Computer Security that presents a collection of papers that synthesise the major results from a range of exemplar projects.

This issue contains three papers that represent the state-of-the-art in distributed systems and programming languages for information flow control. An additional two articles were commissioned for this special issue that do the same for specialist hardware architectures and virtualisation platforms, and appeared in Volume 24, Issue 6, of this journal.

Each paper was specially solicited by approaching authors of major papers on verified information flow security that had recently appeared in the top conference venues. We carefully selected work that focused on *practicality*, and presented feature-rich *systems* or platforms with strong information flow guarantees, that have remained under active development, while ensuring we covered the broadest portion of the traditional hardware/software stack (hardware, operating system, programming language,

[*] Corresponding author. E-mail: toby.murray@unimelb.edu.au.

distributed system). Thus the running theme of this special issue is very much on the kinds of practical security guarantees that can be obtained, and secure systems that can be constructed, by system- and platform-enforced information flow control.

Hardware. Azevedo de Amorim et al. [1] (in Volume 24, Issue 6 of this journal) present a clean slate hardware architecture that provides special facilities for enforcing strong security properties, and the verification of operating system software that uses these facilities to enforce a strong information flow control property. This work demonstrates the kinds of verified guarantees that can be provided by deploying specialist software atop a clean-slate hardware platform.

Virtualisation software. Guanciale et al. [6] (in Volume 24, Issue 6 of this journal) present a verified virtualisation solution for ARM, which runs entire Linux operating systems while enforcing strong isolation to allow security sensitive applications to run alongside. They deploy the solution in the context of a security monitor running alongside Linux to prevent code injection attacks against the latter. This work demonstrates the kinds of guarantees that can be obtained by deploying verified, minimal virtualisation solutions to confine large legacy, untrustworthy components.

Programming languages. Broberg et al. (in this issue) present the design of the Paragon programming language for programming applications with strong, dynamic information flow guarantees. Paragon provides a set of minimal, yet highly expressive, constructs for defining dynamic, stateful, information flow control policies that are enforced via sophisticated static checking, in an extension of the Java language. Their paper focuses heavily on the design rationale of Paragon and the mechanics of its static policy enforcement. It demonstrates the kinds of guarantees that can be obtained by static checking in the context of careful extensions to mainstream programming languages.

Distributed systems. Liu et al. and Griffin et al. (both in this issue) each present the design of distributed systems, namely Fabric and Hails respectively, for enforcing strong, decentralised information flow policies. Each enables programming distributed applications composing mutually suspicious components accessing persistent data, with strong information flow control applied to how data may be accessed and transmitted through the system. Each is built as an extension of an existing information flow control (IFC) programming language: Liu et al.'s Fabric is an extension of Jif, which is itself an IFC extension of Java; and Griffin et al.'s Hails is an extension of LIO, which adds IFC to Haskell. Hails focuses on supporting distributed web applications, programmed in an extension to the traditional Model-View-Controller paradigm that incorporates security policy enforcement as a first class activity. Fabric is more along the lines of a traditional distributed system, incorporating support for distributed transactions and secure mobile code. Each represents one view on how to apply IFC languages to build platforms for implementing secure distributed systems with strong information flow control.

Acknowledgments

We would like to thank Editor in Chief Pierangela Samarati, and previous co-Editor in Chief Andrew Myers, for supporting this special issue. Our deepest thanks to the expert reviewers for the five papers summarised above, without whose critical appraisals and diligence this special issue would not have been possible, and whose attention to detail ensured each paper was uniformly of very high quality. In alphabetical order: Stephen Chong, Mads Dam, Bart van Delft, David Greve, Cătălin Hriţcu, Limin Jia, Zhong Shao, Gregor Snelting, Michael Tschantz, and Nickolai Zeldovich.

Toby would also like to thank Data61 for their support during the early stages of this project.

References

[1] A. Azevedo de Amorim, N. Collins, A. DeHon, D. Demange, C. Hriţcu, D. Pichardie, B.C. Pierce, R. Pollack and A. Tolmach, A verified information-flow architecture, *Journal of Computer Security* **24**(6) (2016), 667–688. doi:10.3233/JCS-15746.

[2] L. Bauer, S. Cai, L. Jia, T. Passaro, M. Stroucken and Y. Tian, Run-time monitoring and formal analysis of information flows in chromium, in: *Proceedings of the 22nd Annual Network & Distributed System Security Symposium*, Internet Society, 2015.

[3] T. Bauereiß, A.P. Gritti, A. Popescu and F. Raimondi, CoSMed: A confidentiality-verified social media platform, in: *International Conference on Interactive Theorem Proving*, Springer, 2016, pp. 87–106.

[4] D.E. Denning, A lattice model of secure information flow, *Communications of the ACM* **19** (1976), 236–242. doi:10.1145/360051.360056.

[5] J. Goguen and J. Meseguer, Security policies and security models, in: *IEEE Symposium on Security and Privacy*, IEEE Computer Society, 1982, pp. 11–20.

[6] R. Guanciale, H. Nemati, M. Dam and C. Baumann, A verified information-flow architecture, *Journal of Computer Security* **24**(6) (2016), 793–837. doi:10.3233/JCS-160558.

[7] D. Hedin, L. Bello and A. Sabelfeld, Information-flow security for JavaScript and its APIs, *Journal of Computer Security* **24**(2) (2016), 181–234. doi:10.3233/JCS-160544.

[8] D. Jang, Z. Tatlock and S. Lerner, Establishing browser security guarantees through formal shim verification, in: *Proceedings of the 21st USENIX Security Symposium*, USENIX Association, 2012, p. 8.

[9] L. Jia, J. Aljuraidan, E. Fragkaki, L. Bauer, M. Stroucken, K. Fukushima, S. Kiyomoto and Y. Miyake, Run-time enforcement of information-flow properties on Android (extended abstract), in: *Computer Security – ESORICS 2013: 18th European Symposium on Research in Computer Security*, Springer, 2013.

[10] S. Kanav, P. Lammich and A. Popescu, A conference management system with verified document confidentiality, in: *International Conference on Computer Aided Verification*, Springer, 2014, pp. 167–183.

[11] T. Murray, D. Matichuk, M. Brassil, P. Gammie, T. Bourke, S. Seefried, C. Lewis, X. Gao and G. Klein, seL4: From general purpose to a proof of information flow enforcement, in: *IEEE Symposium on Security and Privacy*, IEEE Computer Society, 2013, pp. 415–429. doi:10.1109/SP.2013.35.