# Combinatorial Security Testing Course

Dimitris E. Simos
SBA Research
Vienna, Austria
dsimos@sba-research.org

Rick Kuhn
NIST
Gaithersburg, MD, USA
d.kuhn@nist.gov

Yu Lei
University of Texas at Arlington
Texas, USA
ylei@cse.uta.edu

Raghu Kacker
NIST
Gaithersburg, MD, USA
raghu.kacker@nist.gov

## ABSTRACT

Combinatorial methods have attracted attention as a means of providing strong assurance at reduced cost, but when are these methods practical and cost-effective? This tutorial comprises two parts. The first introductory part will briefly explain the background, process, and tools available for combinatorial testing, including illustrations based on industry's experience with the method.

The main part, explains combinatorial testing-based techniques for effective security testing of software components and large-scale software systems. It describes quality assurance and effective re-verification for security testing of web applications and security testing of operating systems. It will further address how combinatorial testing can be applied to ensure proper error-handling of network security protocols and provide the theoretical guarantees for expelling Trojans injected in cryptographic hardware. Procedures and techniques, as well as workarounds will be presented and captured as guidelines for a broader audience. The tutorial is concluded with our vision for combinatorial security testing together with some current open research problems.

The tutorial is designed for participants with a solid IT security background but will not assume any prior knowledge on combinatorial security testing. Thus, we will quickly advance our discussion into core aspects of this field. This tutorial is a modified version of the tutorial held at HVC2017 [19] and QRS2016 [23]. It incorporates feedback and customized content.

## KEYWORDS

combinatorial testing, security testing, software quality assurance, security vulnerabilities

**ACM Reference Format:**
Dimitris E. Simos, Rick Kuhn, Yu Lei, and Raghu Kacker. 2018. Combinatorial Security Testing Course: Tutorial Proposal. In *Proceedings of Hot Topics in the Science of Security (HOTSOS) conference (HOTSOS'18).* ACM, New York, NY, USA, Article 4, 3 pages. https://doi.org/10.475/123_4


*HoTSoS '18*, April 10–11, 2018, Raleigh, NC, USA

ACM ISBN 978-1-4503-6455-3/18/04.
https://doi.org/10.1145/3190619.3191686

## 1 INTRODUCTION

Identifying vulnerabilities and ensuring security functionality by security testing is a widely applied measure to evaluate and improve the security of software, which is also an inevitable part of quality assurance. Many software security exploitations result from ordinary coding flaws, rather than design or configuration errors. One study found that 64 percent of vulnerabilities are the result of such common bugs as missing or incorrect parameter checking, which leaves applications open to common vulnerabilities including buffer overflows or SQL injection [9]. Although this statistic might be discouraging, it also means that better **functionality testing** can also significantly improve security.

In the last 50 years, combinatorial methods have had profound applications in coding theory, cryptology, networking and computer science with software testing being one of the most recent ones [4]. Covering arrays (CAs) [3] are discrete mathematical structures which, with the aid of proper software engineering techniques, have been utilized in very effective test sets in order to provide strong assurance. Yet, the application of combinatorial methods to applied computer science continues to arise and it comes as no surprise that the field of software security, in particular, provides a rich source of problems that seek solutions from mathematical methods. There has been ample evidence over the last few years to support this observation. List below are several reasons that serve as the **motivation to apply combinatorial methods in order to ensure the quality of secure software**:

- The exemplary case of the Heartbleed bug[1], which allowed anyone on the Internet to read the memory of systems protected by the OpenSSL software (e.g. banking applications), highlighted even more the great need to ensure an attack-free environment of implementations of software systems [5].
- Due to the still increasing interconnectedness of such complex software systems, it is very important to strengthen activities towards assuring their security requirements by performing security testing [27].
- The latter task is not be considered an easy process, bearing in mind that software testing may consume up to half of the overall software development cost [26].

[1] heartbleed.com

- Combinatorial explosion is a frequently occurring problem in testing [16], [1] where a test object is described by a number of parameters, each with many possible values. The effect of the combinatorial explosion is that it is infeasible to test every combination of parameter values.
- There exists an added level of complexity for security testing where the modelling of vulnerabilities is specific to the application domain and the identification of factors triggering such exploits is not easily done [18], [17].
- Finally, there are relatively few good methods for evaluating test set quality, after ensuring basic requirements-traceability [14]. Of particular importance is the task to develop methods that help estimate the residual risk that remains after testing.

In [24] the authors developed an ambitious research program aimed at bridging the gap between combinatorial testing (CT) and security testing and, in the process, established a new research field: *combinatorial security testing*. Several methods and case stud- ies presented in this tutorial, illustrate our experiences thus far and the success of the previously mentioned research program, which came as a result of the application of our combinatorial techniques in security testing. In summary, we showed in [24] that the devel- oped concept of CT applicable to security testing supersedes other testing approaches due to its advantages of generating minimal size test sets and revealing hard-to-spot errors in software systems in an automated way.

## 2 OUTLINE OF THE TUTORIAL

In this tutorial we present our work on combinatorial methods for security testing, which guarantees certain aspects of test quality
e.g. test coverage or locating faults. In particular, we formulate problems of software security testing as combinatorial problems and then use efficient algorithmic or theoretical methods to tackle them. *The central thesis of this tutorial is that combinatorial methods can make software security testing much more efficient and effective than conventional approaches - in specific application domains*.

**Brief Introduction of Combinatorial Testing:** This provides a quick overview of the history of combinatorial testing re- search, and their roots based on key publications in the field [15],
[12] and [13].

**Web Security Interaction Testing:** Here the concern is with the problem of security vulnerability detection and with the inherent, but also equally important, problem of retrieving the root cause of security vulnerabilities. We will demonstrate the process of creating attack models used for exploiting web security vulnerabilities using combinatorial methods [6], [2] and indicating methods for analyzing them [21]. The main goal of this part is to make everyone familiar with advanced combinatorial techniques for web security testing.

**Security Protocol Interaction Testing:** In this part of the tutorial we deal with the problem of certificate testing, which plays a central role in network security. We will present complex combinatorial models for creating test certificates to check for faults in the validation logic, which can result in impersonation attacks [11]. In addition, we will present recent efforts on the modelling of the TLS Handshake protocol using CT [20]. If time permits, the authors plan to analyze the TLS cipher suites of the aforementioned protocol using combinatorial coverage measurement techniques [22] and detail the implications of the findings for software security testing.

**Combinatorial Methods for Kernel Software:** The kernel of an operating system is the central authority to enforce security. The goal in this part of the tutorial is to ensure the reliability and quality assurance of kernel software. We will present two testing frameworks, ERIS [7], a combinatorial kernel testing tool, and its recent enhancement, called KERIS [8], with dynamic memory error detectors for the Linux Kernel aimed at exploiting security vulnerabili- ties. We will reproduce for the participants a security vulnerability in the Linux networking stack first discovered by Google's Project Zero team.

**Detecting Hardware Trojan Horses:** This part outlines the problem of malicious hardware logic detection. In particular, the concern is with cryptographic Trojans appearing as instances of malicious hardware. The exemplary scenario for this tutorial evolves around Trojans residing inside cryptographic circuits that perform encryption and decryption in FPGA technologies using the AES cryptographic algorithm. We will demonstrate that combinatorial testing constructs are capable of reducing the number of test cases needed for the Trojan excitation by several orders of magnitude, while at the same time activate the Trojan hundreds of times [10]. The authors will also briefly present similar patterns for AES software implementations based on a recent combinatorial analy- sis performed on AES validation tests [25]. Whether these latter patterns are also malicious is currently an open question.

**Open challenges and outlook:** The authors present and quickly discuss currently unsolved challenges.

## 3 INTENDED AUDIENCE

This **75 minutes** tutorial does not assume any prior knowledge of combinatorial testing methods for information security. We assume good general knowledge of information security and software engineering on a graduate CS student level with a focus on security. The goal of this tutorial is to present the knowledge from various sources in a structured way and provide researchers with the **practical fundamentals** of combinatorial methods for security testing and practitioners with the **scientific background**.

The **key takeaways** are: (I) the practical fundamentals of combinatorial methods for security testing, (II) a good understanding of the underlying mathematical, software engineering and security mechanics and, (III) an overview of the related literature and open problems in this field.

## 4 SHORT BIOS

*Biography of Dimitris Simos.* Dimitris E. Simos is a Key Researcher with SBA Research, Austria, for the "applied discrete mathematics for information security" research area where he is leading the combinatorial security testing team. He is also an Adjunct Lecturer with Vienna University of Technology and a Distinguished Guest Lecturer with Graz University of Technology. His research interests include combinatorial designs and their applications to software testing, combinatorial testing in particular, applied cryptography and optimization algorithms, and information security. He holds a Ph.D. in Discrete Mathematics and Combinatorics (2011) from

the National Technical University of Athens. Prior to joining SBA Research, he was within the Project Team SECRET of INRIA Paris-Rocquencourt Research Center working on the design and analysis of cryptographic algorithms. His research was supported by a 3-year Marie Curie Fellow grant (2012-2015) awarded by the ERCIM through the EU-funded "Alain Bensoussan" Fellowship Programme. He is the author of over 70 papers in discrete mathematics and their applications to computer science and a Fellow of the Institute of Combinatorics and its Applications (FTICA). He was the general chair of MACIS 2017 and also PC chair for IWCT (2017 and 2018).

*Biography of Richard Kuhn.* Rick Kuhn is a computer scientist in the Computer Security Division of the National Institute of Standards and Technology. He has authored two books and more than 150 conference or journal publications on information security, empirical studies of software failure, and software assurance, and is a Fellow of the Institute of Electrical and Electronics Engineers (IEEE). His research interests are in combinatorial methods for software testing, security and access control, and empirical studies of software failure.

*Biography of Yu Lei.* Yu Lei is a Professor of Computer Science at the University of Texas at Arlington. He was a Member of Technical Staff in Fujitsu Network Communications, Inc. from 1998 to 2001. His current interests include combinatorial testing, concurrency testing, and security testing. He served on the Program Committee of ICST 2008 – 2009 and 2012 –2015.

*Biography of Raghu Kacker.* Raghu Kacker is a mathematical statistician in the Applied and Computational Mathematics Division (ACMD) of the Information Technology Laboratory (ITL) of the US National Institute of Standards and Technology (NIST). His current interests include combinatorial testing of software and system s and evaluation of uncertainty in outputs of computational models and physical measurements. Advancing the methods and tools for combinational testing is a mission of his. He has authored or co-authored over 100 papers. He has a Ph.D. in statistics. He is a Fellow of the American Statistical Association and a Fellow of the American Society for Quality.

## 5 ACKNOWLEDGEMENTS

This research was funded by COMET K1, FFG - Austrian Research Promotion Agency, FFG BRIDGE Early Stage SPLIT and FFG BRIDGE SecWIT projects.

Disclaimer: *Products may be identified in this document, but identification does not imply recommendation or endorsement by NIST, nor that the products identified are necessarily the best available for the purpose.*

## REFERENCES

[1] Paul Ammann and Jeff Offutt. 2008. *Introduction to Software Testing* (1 ed.). Cambridge University Press, New York, NY, USA.

[2] J. Bozic, B. Garn, I. Kapsalis, D. E. Simos, S. Winkler, and F. Wotawa. 2015. Attack Pattern-Based Combinatorial Testing with Constraints for Web Security Testing. In *QRS '15: Proceedings of the 2015 IEEE International Conference on Software Quality, Reliability and Security*. 207–212.

[19] D. E. Simos. 2017. Combinatorial Security Testing: Quo Vandis?. https://www.research.ibm.com/haifa/conferences/hvc2017/tutorials.shtml.

[20] D. E. Simos, J. Bozic, F. Duan, B. Garn, K. Kleine, Y. Lei, and F. Wotawa. 2017. Testing TLS Using Combinatorial Methods and Execution Framework. In *ICTSS '17: Proceedings of the 29th International Conference on Testing Software and Systems, Lecture Notes in Computer Science*, Vol. 10533. 162–177.

[21] D. E. Simos, K. Kleine, L. Ghandehari, B. Garn, and Y. Lei. 2016. A combinatorial approach to analyzing cross-site scripting (XSS) vulnerabilities in web application security testing. In *ICTSS '16: Proceedings of the 28th International Conference on Testing Software and Systems, Lecture Notes in Computer Science*, Vol. 9976. 70–85.

[22] D. E. Simos, K. Kleine, A. G. Voyiatzis, R. Kuhn, and R. Kacker. 2016. TLS Cipher Suites Recommendations: A Combinatorial Coverage Measurement. In *QRS '16: Proceedings of the 2016 IEEE International Conference on Software Quality,*

[3] Charles J. Colbourn. 2006. Covering Arrays. In *Handbook of Combinatorial Designs* (2nd ed.), Charles J. Colbourn and Jeffrey H. Dinitz (Eds.). CRC Press, Boca Raton, Fla., 361–365.

[4] Charles J. Colbourn and Paul C. van Oorschot. 1989. Applications of Combinatorial Designs in Computer Science. *ACM Comput. Surv.* 21, 2 (1989).

[5] Zakir Durumeric, James Kasten, David Adrian, J. Alex Halderman, Michael Bailey, Frank Li, Nicolas Weaver, Johanna Amann, Jethro Beekman, Mathias Payer, and Vern Paxson. 2014. The Matter of Heartbleed. In *Proceedings of the 2014 Conference on Internet Measurement Conference (IMC '14)*. ACM, New York, NY, USA, 475–488.

[6] B. Garn, I. Kapsalis, D. E. Simos, and S. Winkler. 2014. On the Applicability of Combinatorial Testing to Web Application Security Testing: A Case Study. In *JAMAICA 14: Proceedings of the 2nd International Workshop on Joining AcadeMiA and Industry Contributions to Test Automation and Model-based Testing, collocated with ISSTA '14: International Symposium on Software Testing and Analysis, ACM*. 16–21.

[7] B. Garn and D. E. Simos. 2014. Eris: A Tool for Combinatorial Testing of the Linux System Call Interface. In *IWCT '14: Proceedings of the 3rd International Workshop on Combinatorial Testing, collocated with ICST '14: 7th IEEE International Conference on Software Testing, Verification and Validation*. 58–67.

[8] B. Garn, F. Würfl, and D. E. Simos. 2017. KERIS: A CT Tool of the Linux Kernel with Dynamic Memory Analysis Capabilities. In *HVC '17: Proceedings of the 13th Haifa Verification Conference, Lecture Notes in Computer Science*, Vol. 10629. 225–228.

[9] Jon Heffley and Pascal Meunier. 2004. Can Source Code Auditing Software Identify Common Vulnerabilities and Be Used to Evaluate Software Security?. In *HICSS*.

[10] P. Kitsos, D. E. Simos, Jose Torres-Jimenez, and A. G. Voyiatzis. 2015. Exciting FPGA Cryptographic Trojans using Combinatorial Testing. In *ISSRE '15: Proceedings of the 26th IEEE International Symposium on Software Reliability Engineering*. 69–76.

[11] K. Kleine and D. E. Simos. 2017. Coveringcerts: Combinatorial Methods for X.509 Certificate Testing. In *ICST '17: Proceedings of the 10th IEEE International Conference on Software Testing, Verification and Validation*. 69–79.

[12] D.R. Kuhn, R.N. Kacker, and Y. Lei. 2013. *Introduction to Combinatorial Testing*. Taylor & Francis.

[13] D Richard Kuhn, Renee Bryce, Feng Duan, Laleh Sh Ghandehari, Yu Lei, and Raghu N Kacker. 2015. Chapter one-combinatorial testing: Theory and practice. *Advances in Computers* 99 (2015), 1–66.

[14] D Richard Kuhn, Raghu N. Kacker, and Yu Lei. 2015. Combinatorial Coverage as an Aspect of Test Quality. *CrossTalk* 28, 2 (2015), 19–23.

[15] Rick Kuhn, Yu Lei, and Raghu Kacker. 2008. Practical combinatorial testing: Beyond pairwise. *It Professional* 10, 3 (2008).

[16] Aditya P. Mathur. 2008. *Foundations of Software Testing* (1st ed.). Addison-Wesley Professional.

[17] Gary McGraw. 2006. *Software Security: Building Security In*. Addison-Wesley Professional.

[18] B. Potter and G. McGraw. 2004. Software security testing. 2, 5 (2004), 81–85. *IEEE Security Privacy Reliability and Security*. 69–73.

[23] D. E. Simos, Rick Kuhn, Yu Lei, and Raghu Kacker. 2016. Combinatorial Security Testing. http://paris.utdallas.edu/qrs16/program/QRS-2016-Program.pdf.

[24] D. E. Simos, R. Kuhn, A. G. Voyiatzis, and R. Kacker. 2016. Combinatorial methods in security testing. *IEEE Computer* 49 (2016), 40–43.

[25] D. E. Simos, S. Mekesis, D. R. Kuhn, and R. N. Kacker. [n. d.]. Combinatorial Coverage Measurement of Test Vectors used in Cryptographic Algorithm Validation. In *STC '17: Proceedings of the 2017 IEEE Software Technology Conference*.

[26] G. Tassey. 2002. *The economic impacts of inadequate infrastructure for software testing*. National Institute of Standards and Technology.

[27] F. Wotawa. 2016. On the Automation of Security Testing. In *2016 International Conference on Software Security and Assurance (ICSSA)*. 11–16.