Research Statement

SHAR Lwin Khin
School of Computing and Information Systems,
Singapore Management University
Tel: (65) 6828-0019; Email: lkshar@smu.edu.sg
04 January 2021

Background

Software systems are pervasive in all industry sectors and have become increasingly complex and critical. As a result, the issues and challenges traditionally faced in software development are becoming more acute. They need more effective, efficient development processes, more automation, and more scalable solutions. Therefore, software engineering research is highly relevant and could have a significant impact to many industry sectors. However, there is a gap between academics research and industrial needs in software engineering. In our software engineering research community, we tend to measure success by counting publications, which naturally lead to academics producing papers that may or may not be industry-relevant. A prominent software engineering researcher stated that among the publications from the top software engineering research venues, only a small proportion of the papers stem from industry-relevant research. My research aims to be industry-relevant. For this to happen, it must be context-driven.

Context-driven research doesn't try to frame a general problem and devise universal solutions. Rather, for a given context that is precisely defined, it makes clear working assumptions and devises practical solutions that work in such a context. It also considers and makes tradeoffs that make sense in the given context to achieve practicality and scalability. Clearly, this doesn't produce solutions that generalize easily to any arbitrary software development environment. But this doesn't reduce its value because universal solutions hardly work, anyway, for different software systems that have different contextual factors such as software complexity and characteristics, domain-related criticality and compliance with standards, organization's cost and time constraints, and human factors (e.g. engineers' background). Software from telecommunication industry could be very much different from those of automotive industry.

Whether the cost of a comprehensive software validation technique is justified will depend on the criticality of the software being assessed and the standards it must comply with. For example, in Singapore, software systems that acquire user privacy data must comply with Personal Data Protection Act (PDPC); hence a validation technique that specifically focuses on the PDPC will be beneficial for such systems. That is, they require a technique that can transform the natural language form of PDPC into a machine-analyzable form, that can extract the security and privacy requirements embedded in the PDPC, and that can support

automated or semi-automated generation of test scenarios to validate these requirements. Although this may sound universal, in the nutshell, an effective technique requires domain-specific considerations — specific terms and definitions used in the PDPC must be considered for the natural language processing; testing typically requires specific set of inputs such as the inputs' form and content; and any other contextual factors such as the development process used, the cost and time constraints, etc. must be considered.

Research Areas

My research mainly focuses on security and privacy analysis of web/mobile/IoT applications, which falls under the umbrella of software validation and verification domain – a sub-discipline of software engineering. In terms of software development lifecycle that comprises requirement elicitation, specification, design, development, testing, deployment, and maintenance, my work is mainly applicable to testing and maintenance phases. Particularly, I work on analyzing malware and vulnerabilities in a given software artefact. The kinds of software artefact applicable to my research range from source code and byte code to deployed software. My research is not limited to the use of any particular technique but for the problems that I am addressing, I found that the combined use of static analysis, dynamic analysis, search-based test generation, and machine learning techniques suits the need so far. The following discusses some of the major work that achieved high impact in my research domain:

Vulnerability prediction [1-4]. The work proposed in ICSE NIER track [1] is one of the pioneer works that uses machine learning on code characteristics to predict vulnerabilities in web applications, differently from the then-approaches that mainly focused on the use of static and dynamic analyses only for detecting web vulnerabilities. In [1] and its subsequent extensions [2-4], we incorporated the use of machine learning to address the scalability and effectiveness problems of static and dynamic analyses; we proposed approaches that predict various code injection vulnerabilities (SQL injection, cross site scripting, remote code execution, path traversal, etc.) in web programs by combining static and dynamic analyses with machine learning. Static and dynamic analyses are first used to extract code features from web programs containing known vulnerabilities. The extracted features reflect information about potentially correct and incorrect validation and sanitization routines implemented in a program. Supervised, semi-supervised, unsupervised machine learning techniques are then applied to build vulnerability predictors based on these features and the known, available vulnerability information. The predictors are then applied to predict vulnerabilities in other web programs. The experimental results show that the best predictor is able to detect 90% of vulnerabilities with 12% false alarm rate.

Vulnerability Analysis [5-9]. Vulnerability predictors have been shown to be effective, but they only predict vulnerable code sections. They do not provide

comprehensive information on the deficiencies of the implementations that cause vulnerabilities. Therefore, to complement the above prediction approaches and to assist the developers in auditing the code, in [5, 6], I have also worked on a static analysis-based approach that specifically extracts security-relevant code slices for security auditing purposes. It also provides guidelines to carry out security audits based on the extracted code slices. This work addressed the technical and scalability challenges of precisely extracting the code slices for Java-based web programs. The tool that automates the proposed approach [7] is available to public and is maintained at https://github.com/julianthome/joanaudit

In [8, 9], we extended this security auditing research in the area of improving the constraint solvers. This also augments existing vulnerability detection approaches. Given code slices, a powerful constraint solver will be able to tell precisely what inputs can cause the vulnerabilities. We developed such a constraint solver addressing the challenges of solving complex string operations and mixed (integer and string) operations in a scalable and effective way. For scalability, we built several finite-state machines (FSMs) that model various Java string operations, which avoid the need for resolving such operations into basic set of operations that are often recursive and complex to solve; and then incorporated a search-based input generation algorithm into the FSMs-based constraint solver for effectiveness. The tool that implements the approach was evaluated on an industrial code base provided by HITEC Luxembourg (www.hitec.lu).

Privacy Analysis [10][11]. Even though security and privacy are often mentioned together and many approaches claimed to address both issues at the same time, there are some subtle differences that pose specific challenges for addressing privacy concerns properly. Security has somewhat commonly-accepted definitions and properties; for example, a secure communication system can be defined as the one that has confidentiality, integrity, and availability properties. By contrast, privacy is hard to be defined appropriately as it could mean differently to different people. To address this problem, in the scope of Android applications, in [10], we proposed an approach that reports anomalous data flows to users for them to make informed decisions. Firstly, using natural language processing and clustering techniques, our approach forms clusters of apps where each cluster contains a group of similar trusted, benign apps according to their functional descriptions; the approach then learns their normal behaviors by analyzing the call graphs of the app code and by running diverse test cases. We combined static analysis and genetic algorithm-based test generation so as to observe diverse behaviors as many as possible. Lastly, give a test app, the approach compares its behaviors against those of the apps in the same cluster.
Even when privacy requirements are well-defined by stakeholders for privacy-sensitive systems, it is often the case that the requirements are only visible in the requirement documents; there is no traceability between the privacy specifications and the actual implementations. Hence, in this context, I have worked on a European industrial & academic research project called EDLAH2 (http://edlah2.eu), which attempts to address this problem. In this work, we

developed a modeling method for specifying the security and privacy requirements in a traceable way [11].

Future Plan

Going forward, my long term plan is to improve the practice of software engineering through a context-driven research. From the successful collaborations with industry partners in my research, I have proved that context-driven software engineering research can have a high impact on industrial practice. Given an opportunity to form a research team, I aim to realize this vision by conducting context-driven research in security testing in general with industrial collaborations, developing effective, efficient, (semi-)automated, and/or scalable solutions for testing their complex software systems under precise contexts.

As a concrete plan for upcoming years: given the rise of smart devices and IoT apps and the pervasiveness of vulnerabilities and malware in this new computing paradigm, there is a need of applicable security analysis methods and tools. Traditional vulnerability and malware analysis tools currently have limitations in terms of compatibility, scalability, and effectiveness, due to the unique nature of IoT ecosystem such as high diversity and high interactivity. I plan to devise novel methods and tools to address these limitations.

Selected Publications and Outputs

[1] L. K. Shar, H. B. K. Tan. Mining input sanitization patterns for predicting SQL injection and cross site scripting vulnerabilities. In International Conference on Software Engineering (ICSE), pp. 1293-1296, 2012.

[2] L. K. Shar, H. B. K. Tan, and L. Briand. Mining SQL injection and cross site scripting vulnerabilities using hybrid program analysis. In International Conference on Software Engineering (ICSE), pages 642–651, 2013.

[3] L. K. Shar, L. Briand, and H. B. K. Tan. Web application vulnerability prediction using hybrid program analysis and machine learning. In IEEE Transactions on Dependable and Secure Computing, 12(6):688–707, 2015.

[4] L. K. Shar, H. B. K. Tan. Predicting SQL injection and cross site scripting vulnerabilities through mining input sanitization patterns. In Information and Software Technology, 55(10), 1767-1780, 2013.

[5] J. Thome, L. K. Shar, and L. Briand. Security Slicing for Auditing XML, XPath, and SQL Injection Vulnerabilities. In 26th IEEE International Symposium on Software Reliability Engineering (ISSRE), 2015.

[6] J. Thome, L. K. Shar, D. Bianculli, and L. Briand. Security slicing for auditing common injection vulnerabilities. In Journal of Systems and Software, 137, 766-783, 2018.

[7] J. Thome, L. K. Shar, D. Bianculli, L. Briand. JoanAudit: A Tool for Auditing Common Injection Vulnerabilities. In ACM SIGSOFT Symposium on the Foundations of Software Engineering (ESEC/FSE), pp. 1004-1008, 2018.

[8] J. Thome, L. K. Shar, D. Bianculli, and L. Briand. Search-driven String Constraint Solving for Vulnerability Detection. In International Conference on Software Engineering (ICSE), pp. 198-208, 2017.

[9] J. Thome, L. K. Shar, D. Bianculli, L. Briand. An Integrated Approach for Effective Injection Vulnerability Analysis of Web Applications through Security Slicing and Hybrid Constraint Solving. In IEEE Transactions on Software Engineering, 2018.

[10] B. F. Demissie, M. Ceccato, and L. K. Shar. AnFlo: Detecting Anomalous Sensitive Information Flows in Android Apps. In 5th IEEE/ACM International Conference on Mobile Software Engineering and Systems (MobileSoft), pp. 24-34, 2018.

[11] X. P. Mai, A. Goknil, L. K. shar, F. Pastore, L. Briand, S. Shaame. Modeling Security and Privacy Requirements: a Use Case-Driven Approach. Information and Software Technology, vol. 100, 165-182, 2018