Research Statement

Yue Duan
School of Computing and Information Systems, Singapore Management University
Tel: (65) 6826-4807; Email: yueduan@smu.edu.sg
16 Dec 2024

Background

My research interests mainly lie in computer security, deep learning, and blockchain. As computer systems (e.g., mobile devices, IoT devices) and software have been deeply integrated into our daily lives, security has become a major concern. My goal is to make computer systems and software more secure by combining various static and dynamic program analysis techniques with other techniques, such as deep learning, model checking, and natural language processing (NLP). My research has enabled novel techniques to improve the security of systems and software in different domains, including computer systems, mobile devices, and blockchain, and has led to new techniques, algorithms, and important discoveries in the field. The following sections elaborate on my research in three areas and conclude with a future plan.

Research Areas

1. AI for Security

Recently, large language models (LLMs) have garnered significant attention from the security community due to their remarkable capabilities in code understanding and analysis. However, researchers have recognized that LLMs are not easily applicable to security challenges in a direct manner. My research focuses on bridging this gap by combining the power of AI with my domain expertise in security to address real-world security problems.

CodeBreaker [5]. LLM have transformed code completion tasks, providing context-based suggestions to boost developer productivity in software engineering. As users often fine-tune these models for specific applications, poisoning and backdoor attacks can covertly alter the model outputs. To address this critical security challenge, we introduced Codebreaker, a pioneering LLM-assisted backdoor attack framework on code completion models. It leverages LLMs (e.g., GPT-4) for sophisticated payload transformation (without affecting functionalities), ensuring that both the poisoned data for fine-tuning and generated code can evade strong vulnerability detection. The paper was published in USENIX Security'24.

DisasLLM [7]. Disassembly is a challenging task, particularly for obfuscated executables containing junk bytes, which is designed to induce disassembly errors. Existing solutions rely on heuristics or leverage machine learning techniques but only achieve limited success. Fundamentally, such obfuscation cannot be defeated without in-depth understanding of the binary executable's semantics, which is made possible by the emergence of large language models (LLMs). We presented DisasLLM, a novel LLM-driven dissembler to overcome the barriers in analyzing obfuscated executables. The paper is under submission.

SemPRE [8]. Protocol reverse engineering is essential yet challenging in many security application scenarios, such as malware analysis. Existing PRE techniques depend on brittle, hand-written rules, ignore the contextual dependencies among neighboring fields, and break down when a message embeds nested sub-protocols. We propose SemPRE, a fully automated PRE framework that integrates fine-grained dynamic program analysis with a recursive LLM-based reasoning process. The paper is under submission.

2. Blockchain Security

Smart contracts in blockchains are holding cryptocurrencies and tokens valued at billions of USD. One of its core applications is to support a variety of decentralized applications (DApps), such as NFT and DeFi. I have investigated two topics in this area: DApps safety vetting and upgradeable smart contract study.

VetSC [3]. Very little work has been done to target high-level semantic issues (e.g., unfair business models) in DApps. Existing techniques suffer from low efficiency and require extensive manual effort to create specific policies for each contract by understanding the source code. Hence, they are highly impractical. To overcome the limitations, we designed VetSC [5], a novel UI-driven, program analysis-guided model-checking technique that can automatically extract contract semantics in DApps and perform safety vetting. From the smart contract code, we extract business model graphs that capture its intrinsic business and safety logic to facilitate model checking. We further leverage textual semantics from DApp user interfaces to help validate the UI-logic consistency and detect any discrepancies. We have successfully discovered 19 new safety risks in the wild, such as expired lottery tickets and double voting. This work won the Best Paper Honorable Mention award at ACM CCS'22.

UscHunt [2]. Upgradeable Smart Contract (USC) has become the future of smart contract development as it allows developers to perform upgrades on the contract logic. To fully understand the status quo of USCs in the wild and reveal their security implications, we conducted the first-ever systematic study on USCs in the real world. To achieve this goal, we developed UscHunt, an automatic smart contract analysis framework that supports accurate USC detection and detailed behavior analysis. To systematically characterize USCs, for the first time, we proposed a USC taxonomy that comprehensively describes the unique behaviors of USCs at both syntactic and semantic levels. We analyzed 800K+ smart contracts with UscHunt and discovered 2,500+ real-world security and safety risks. The paper was published in USENIX Security'23.

DiffUSC [6]. Smart contract upgrades can accidentally introduce serious vulnerabilities, causing millions of USD losses. To detect security issues brought about by upgrades, we proposed a novel system named **DiffUSC** that discovers behavioral discrepancies between two versions of a smart contract by performing differential fuzzing that focuses on testing the changed code. Once the discrepancies are extracted, we further perform postprocessing to select and report the most important discrepancies to the developers. In the evaluation, we showed that our tool was able to effectively and efficiently detect all four known vulnerabilities related to upgrades. The paper was submitted to DSN'25.

3. Binary Analysis

Researchers have utilized different binary analysis techniques to solve security issues. The two topics I investigated are binary differential analysis and vulnerability discovery.

DeepBinDiff [4]. Binary diffing quantitatively measures the differences between two given binaries and produces fine-grained basic block matching. It has been widely used to enable different kinds of critical security analyses, including security patch analysis and malware analysis. However, all existing techniques need more accuracy, better scalability, and coarse granularity or require extensive labeled training data to function. To address the limitations, we designed DeepBinDiff [2], an unsupervised program-wide code-representation-learning technique that relies on both the code semantics information and the program-wide control flow information to generate block embeddings. We further proposed a k-hop greedy matching algorithm to find the optimal diffing results using the generated block embeddings. The evaluation showed that DeepBinDiff outperforms the state-of-the-art by a large margin for both cross-version and cross-optimization-level diffing. The paper was published in NDSS'20.

SigmaDiff [1]. Furthermore, we invented a novel technique called **SigmaDiff** to perform binary diffing at the pseudocode level for better precision and human readability. To this end, we first perform diffing at the IR level and map the IR-level diffing results up to the pseudocode level. We construct interprocedural program dependency graphs and extract a symbolic expression for each IR to capture the semantic meanings of a binary. After that, we model the program-wide IR diffing problem as a graph-matching problem on two IPDGs from the two given binaries by leveraging the deep graph matching consensus (DGMC) model to fully exploit the neighboring contextual information. The paper has been accepted in NDSS'24.

Future Directions

I expect to continue my current research and expand the effort to solve other emerging security issues

1. Directed Fuzzing Scheduling

Unlike traditional grey-box fuzzing that explores whole program states, directed fuzzing takes as input a list of pre-defined potential vulnerable code locations and dedicates all its resources to directing the testing to these locations and triggering potential vulnerabilities. However, one big research question: how to effectively schedule these locations, remains to be answered. Therefore, we propose to model the fuzzing campaign using statistical methods and estimate a probability for each location, to which fuzzers could successfully exploit the vulnerability, to schedule directed fuzzing to work on more promising locations and utilize limited resources more effectively.

2. IoT Security

The security and privacy issues in IoT devices, ubiquitous in our lives, have become huge concerns for end users. Analyzing IoT devices face unique challenges in that they are heterogeneous by nature: people only have access to the mobile apps that

interact with the IoT devices, but the firmware controls the devices. Therefore, hidden behaviors in IoT could exist and users may be completely unaware of them. I plan to use the NLP technique to extract semantic information from the GUI components in mobile apps and leverage Android app analysis and fuzzing techniques to identify hidden behaviors in IoT devices.

Selected Publications and Outputs

[1] [NDSS'24] Lian Gao, Yu Qu, Sheng Yu, **Yue Duan**, and Heng Yin. "SigmaDiff: Semantics-Aware Deep Graph Matching for Pseudocode Diffing", to appear in Proceedings of the 31st Network and Distributed System Security Symposium, February 2024.

[2] [USENIX Security'23] William E Bodell III, Sajad Meisami, and Yue Duan. "Proxy Hunting: Understanding and Characterizing Proxy-based Upgradeable Smart Contracts in Blockchains", in Proceedings of the 32nd USENIX Security Symposium, August 2023.

[3] [CCS'22] Yue Duan, Xin Zhao, Yu Pan, Shucheng Li, Minghao Li, Fengyuan Xu, Mu Zhang. "Towards Automated Safety Vetting of Smart Contracts in Decentralized Applications", in Proceedings of the 29st ACM Conference on Computer and Communications Security, November 2022.

[4] [NDSS'20] Yue Duan, Xuezixiang Li, Jinghan Wang, and Heng Yin. "DeepBinDiff: Learning Program-Wide Code Representations for Binary Diffing", in Proceedings of the 27th Network and Distributed System Security Symposium, February 2020.

[5] [USENIX Security'24] Shenao Yan, Shen Wang, Yue Duan, Hanbin Hong, Kiho Lee, Doowon Kim, and Yuan Hong. "An LLM-Assisted Easy-to-Trigger Poisoning Attack on Code Completion Models: Injecting Disguised Vulnerabilities against Strong Detection", in Proceedings of the 33rd USENIX Security Symposium, August 2024.

[6] [Submitted] William E Bodell III, Josselin Feist, Gustavo Grieco, and Yue Duan. "Differential Fuzzing of Smart Contracts in Solidity"

[7] [Submitted] Huanyao Rong, Yue Duan, Hang Zhang, XiaoFeng Wang, Hongbo Chen, Shengchen Duan, Shen Wang. "Disassembling Obfuscated Executables with LLM"

[8] [Submitted] Yihua Xu, Wenjie Huang, XiaoFeng Wang, Hang Zhang, Binghui Wang, and Yue Duan, "SEMPRE: Semantic-Aware Protocol Reverse Engineering via Consistency-Driven Joint Optimization"