QINGKAI SHI

Email: shi553@purdue.edu **Homepage:** https://qingkaishi.github.io/ **Twitter:** @QingkaiS

EDUCATION AND EMPLOYMENT

- **Purdue University** 2021 - Now
  Postdoc Research Associate

- **The Hong Kong University of Science and Technology & Sourcebrella Inc.** 2015 - 2021
  Ph.D. in Computer Science (2015 - 2020) & Co-founder of Sourcebrella Inc.† (2015 - 2021)
  Ph.D. Dissertation: Precise and Scalable Static Bug Finding for Industrial-Sized Code
  †Sourcebrella Inc. commercialized my research on static bug finding
  †Sourcebrella Inc. was acquired by Ant Group in 2020

- **Nanjing University** 2008 - 2015
  B.S. in Software Engineering (2008 - 2012) & Research Assistant (2012 - 2015)

RESEARCH AND HONORS

- My research interest centers around programming language, software engineering, and cybersecurity, focusing on the use of programming language and compiler techniques for ensuring software security and reliability.

- My research has been commercialized and received nearly 200 million CNY investment. The tool for hunting software vulnerabilities, known as *Pinpoint Static Code Analyzer*, has been deployed in many Global 500 companies. The startup that commercializes my research has been acquired by Ant Group.

- My research has won me an ACM SIGSOFT Distinguished Paper Award (2019), three champions in NASAC prototype competitions (2016, 2018a, and 2018b), a Hong Kong PhD Fellowship (2015), and two China National Scholarships (2010, 2015).

PUBLICATIONS

I contributed to the following papers in the area of programming language, cybersecurity, and software engineering. First-author papers and corresponding-author papers (papers under my supervision) are made bold.

	2016-2019	2020	2021	2022
Programming Langauge	**PLDI**		**PLDI**, **OOPSLA**	OOPSLA
Cybersecurity		**S&P**		**S&P**
Software Engineering	**TSE**, **TRel**, ICSE	**ICSEx2**, **ISSTAx2**, ISSTAx2	ISSTA, ESEC/FSE	ICSE

[1] Heqing Huang, Yiyuan Guo, **Qingkai Shi***, Peisen Yao, Rongxin Wu, and Charles Zhang. Beacon: Directed Grey-Box Fuzzing with Provable Path Pruning. In *Proceedings of the 43rd IEEE Symposium on Security and Privacy (S&P'22)*. IEEE, 2022.

[2] Chengpeng Wang, Peisen Yao, Wensheng Tang, **Qingkai Shi**, and Charles Zhang. Complexity-Guided Container Replacement Synthesis. In *Proceedings of the 37th ACM SIGPLAN Conference on Object Oriented Programming, Systems, Languages and Applications (OOPSLA'22)*. ACM, 2022.

[3] Yiyuan Guo, Jinguo Zhou, Peisen Yao, **Qingkai Shi**, Charles Zhang. Precise Divide-By-Zero Detection with Affirmative Evidence. In *Proceedings of the 44nd ACM/IEEE International Conference on Software Engineering (ICSE'22)*. ACM, 2022.

[4] Peisen Yao, **Qingkai Shi***, Heqing Huang, and Charles Zhang. Program Analysis via Efficient Symbolic Abstraction. In *Proceedings of the 36th ACM SIGPLAN Conference on Object Oriented Programming, Systems, Languages and Applications (OOPSLA'21)*. ACM, 2021. https://doi.org/10.1145/3485495

[5] Peisen Yao, Heqing Huang, Wensheng Tang, **Qingkai Shi**, Rongxin Wu, and Charles Zhang. Skeletal Approximation Enumeration for SMT Solver Testing. In *Proceedings of the 29th ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering (ESEC/FSE'21)*. pp 1141-1153. ACM, 2021. https://doi.org/10.1145/3468264.3468540

[6] Peisen Yao, Heqing Huang, Wensheng Tang, **Qingkai Shi**, Rongxin Wu, and Charles Zhang. Fuzzing SMT Solvers via Two-Dimensional Input Space Exploration. In *Proceedings of the 30th ACM SIGSOFT International Symposium on Software Testing and Analysis (ISSTA'21)*. pp 322–335. ACM, 2021. https://doi.org/10.1145/3460319.3464803

[7] **Qingkai Shi**, Peisen Yao, Rongxin Wu, and Charles Zhang. Path-Sensitive Sparse Analysis without Path Conditions. In *Proceedings of the 42nd ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI'21)*. pp 930–943. ACM, 2021. https://doi.org/10.1145/3453483.3454086

[8] Peisen Yao, **Qingkai Shi***, Heqing Huang, and Charles Zhang. Fast Bit-Vector Satisfiability. In *Proceedings of the 29th ACM SIGSOFT International Symposium on Software Testing and Analysis (ISSTA'20)*. pp 38–50. ACM, 2020. https://doi.org/10.1145/3395363.3397378

[9] Gang Fan, Chengpeng Wang, Rongxin Wu, Xiao Xiao, **Qingkai Shi**, and Charles Zhang. Escaping Dependency Hell: Finding Build Dependency Errors with the Unified Dependency Graph. In *Proceedings of the 29th ACM SIGSOFT International Symposium on Software Testing and Analysis (ISSTA'20)*. pp 463–474. ACM, 2020. https://doi.org/10.1145/3395363.3397388

[10] Chunrong Fang, Zixi Liu, Yangyang Shi, Jeff Huang, and **Qingkai Shi**. Functional Code Clone Detection with Syntax and Semantics Fusion Learning. In *Proceedings of the 29th ACM SIGSOFT International Symposium on Software Testing and Analysis (ISSTA'20)*. pp 516–527. ACM, 2020. https://doi.org/10.1145/3395363.3397362

[11] Yang Feng, **Qingkai Shi***, Xinyu Gao, Jun Wan, Chunrong Fang, and Zhenyu Chen. DeepGini: Prioritizing Massive Tests to Enhance the Robustness of Deep Neural Networks. In *Proceedings of the 29th ACM SIGSOFT International Symposium on Software Testing and Analysis (ISSTA'20)*. pp 177–188. ACM, 2020. https://doi.org/10.1145/3395363.3397357

[12] **Qingkai Shi** and Charles Zhang. Pipelining Bottom-up Data Flow Analysis. In *Proceedings of the 42nd ACM/IEEE International Conference on Software Engineering (ICSE'20)*. pp 835–847. ACM, 2020. https://doi.org/10.1145/3377811.3380425

[13] **Qingkai Shi**, Rongxin Wu, Gang Fan, and Charles Zhang. Conquering the Extensional Scalability Problem for Value-Flow Analysis Frameworks. In *Proceedings of the 42nd ACM/IEEE International Conference on Software Engineering (ICSE'20)*. pp 812–823. ACM, 2020. https://doi.org/10.1145/3377811.3380346

[14] Heqing Huang, Peisen Yao, Rongxin Wu, **Qingkai Shi***, and Charles Zhang. Pangolin: Incremental Hybrid Fuzzing with Polyhedral Path Abstraction. In *Proceedings of the 41st IEEE Symposium on Security and Privacy (S&P'20)*. pp 1613-1627. IEEE, 2020. https://doi.org/10.1109/SP40000.2020.00063

[15] Gang Fan, Rongxin Wu, **Qingkai Shi**, Xiao Xiao, Jinguo Zhou, and Charles Zhang. SMOKE: Scalable Path-Sensitive Memory Leak Detection for Millions of Lines of Code. In *Proceedings of the 41st ACM/IEEE International Conference on Software Engineering (ICSE'19)*. pp 72–82. IEEE, 2019. https://doi.org/10.

1109/ICSE.2019.00025 **ACM SIGSOFT Distinguished Paper Award**

[16] **Qingkai Shi**, Xiao Xiao, Rongxin Wu, Jinguo Zhou, Gang Fan, and Charles Zhang. Pinpoint: Fast and Precise Sparse Value Flow Analysis for Million Lines of Code. In *Proceedings of the 39th ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI'18)*. pp 693–706. ACM, 2018. https://doi.org/10.1145/3192366.3192418

[17] **Qingkai Shi**, Zhenyu Chen, Chunrong Fang, Yang Feng, and Baowen Xu. Measuring the Diversity of a Test Set with Distance Entropy. In *IEEE Transactions on Reliability (TRel'16)*, Vol. 65, No. 1. pp 19-27. IEEE, 2016. https://doi.org/10.1109/TR.2015.2434953

[18] **Qingkai Shi**, Jeff Huang, Zhenyu Chen, and Baowen Xu. Verifying Synchronization for Atomicity Violation Fixing. In *IEEE Transactions on Software Engineering (TSE'16)*, Vol. 42, No. 3. pp 280-296. IEEE, 2016. https://doi.org/10.1109/TSE.2015.2477820

INVITED TALKS AND PRESENTATIONS

- "Path-Sensitive Sparse Analysis without Path Conditions", the 42nd ACM SIGPLAN conference on Programming Language Design and Implementation (PLDI'21), Virtual, June, 2021. **Presentation**.
- "Fast and Precise Static Bug Detection for Industrial-Sized Code", Texas A&M University, TX, the United States, July, 2021. **Invited Talk**.
- "Taming Path-Sensitivity for Static Code Analysis on Industrial Scale", Nanjing University, Nanjing, China, April, 2021. **Invited Talk**.
- "Pipelining Bottom-up Data Flow Analysis", the 42nd ACM/IEEE International Conference on Software Engineering (ICSE'20), Virtual, July, 2020. **Presentation**.
- "Conquering the Extensional Scalability Problem for Value-Flow Analysis Frameworks", the 42nd ACM/IEEE International Conference on Software Engineering (ICSE'20), Virtual, July, 2020. **Presentation**.
- "The Next Generation Software Security Analysis", Nanjing University, Nanjing, China, May, 2020. **Invited Talk**.
- "Marrying Taint Analysis with Big Data Analytics", Southern University of Science and Technology, Shenzhen, China, November, 2018. **Invited Talk**.
- "Fast and Precise Sparse Value Flow Analysis for Million Lines of Code", the 39th ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI'18), Philadelphia, the United States, June, 2018. **Presentation**.

CVE IDENTIFIERS

[1] CVE-2017-14739: ImageMagick 7.0.7–4 mishandles failed memory allocation, which allows remote attackers to cause a denial of service.

[2] CVE-2017-14952: International Components for Unicode (ICU) for C/C++ through 59.1 contains a double free that allows remote attackers to execute arbitrary code.

[3] CVE-2017-15096: GlusterFS in versions prior to 3.10 contains a null pointer dereference that may cause denial of service.

[4] CVE-2017-16892: Bftpd 4.6 contains a memory leak which occurs if a mal-crafted sequence of FTP requests are received.

[5] CVE-2017-1000445: ImageMagick 7.0.7–1 and older version are vulnerable to null pointer dereference in the MagickCore component and might lead to denial of service.

[6] CVE-2018-20786: libvterm through 0+bzr726, as used in Vim and other products, mishandles certain out-of-memory conditions, leading to a denial of service (application crash), related to screen.c, etc.

[7] CVE-2019-13238: An issue was discovered in Bento4 1.5.1.0. A memory allocation failure is unhandled in Core/Ap4SdpAtom.cpp and leads to crashes. When parsing input video, the program allocates a new buffer to parse an atom in the stream. The unhandled memory allocation failure causes a direct copy to a NULL pointer.

[8] CVE-2019-13959: In Bento4 1.5.1-627, AP4_DataBuffer::SetDataSize does not handle reallocation failures, leading to a memory copy into a NULL pointer.

[9] CVE-2019-13960: In libjpeg-turbo 2.0.2, a large amount of memory can be used during processing of an invalid progressive JPEG image containing incorrect width and height values in the image header.

[10] CVE-2020-19715: An integer overflow vulnerability in the getUShort function of Exiv2 0.27.1 results in segmentation faults within the application, leading to a denial of service (DOS).

[11] CVE-2020-19716: A buffer overflow vulnerability in the Databuf function in types.cpp of Exiv2 v0.27.1 leads to a denial of service (DOS).

[12] CVE-2020-19717: An unhandled memory allocation failure in Core/Ap48bdlAtom.cpp of Bento 1.5.1-628 causes a NULL pointer dereference, leading to a denial of service (DOS).

[13] CVE-2020-19718: An unhandled memory allocation failure in Core/Ap4Atom.cpp of Bento 1.5.1-628 causes a NULL pointer dereference, leading to a denial of service (DOS).

[14] CVE-2020-19719: A buffer overflow vulnerability in Ap4ElstAtom.cpp of Bento 1.5.1-628 leads to a denial of service (DOS).

[15] CVE-2020-19720: An unhandled memory allocation failure in Core/AP4IkmsAtom.cpp of Bento 1.5.1-628 causes a NULL pointer dereference, leading to a denial of service (DOS).

[16] CVE-2020-19721: A heap buffer overflow vulnerability in Ap4TrunAtom.cpp of Bento 1.5.1-628 may lead to an out-of-bounds write while running mp42aac, leading to system crashes and a denial of service (DOS).

[17] CVE-2020-19722: An unhandled memory allocation failure in Core/Ap4Atom.cpp of Bento 1.5.1-628 causes a direct copy to NULL pointer dereference, leading to a denial of service (DOS).

PATENTS

[1] Defect detection method, device. system, and computer readable medium. (US Patent No. 20190108003, China Patent No. 201811013103.6).

[2] Use-after-free detection method, device, system, and computer readable medium. (China Patent No. 201811013000.X).

[3] SQL-injection detection method, device, system, and computer readable medium. (China Patent No. 201811015751.5).

[4] A method of obtaining the feedback on the teaching of Java unit testing. (China Patent No. 2016102941812).

[5] Inter-procedural null dereference detection method, device, system, and computer readable medium. (China Patent No. 2018110146864).

[6] A method of verifying synchronization for atomicity violation fixing. (China Patent No. 2014107099836).

[7] A method of randomly selecting co-diversified test cases. (China Patent No. 2012100526910).

[8] A method of controlling computer with mobile phone's inner sensors. (China Patent No. 2011104124584).

PROFESSIONAL SERVICES

- Reviewer:
  - IEEE Transactions on Reliability.
  - IEEE Transactions on Dependable and Secure Computing.
- Artifact Evaluation Committee:
  - The 49th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages (POPL'22), 16 - 22 January 2022, Philadelphia, Pennsylvania, the United States.
- Student Volunteer:
  - The 37th ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI'16), 13 - 17 June 2016, Santa Barbara, California, the United States.
  - The 33rd ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI'12), 11 - 16 June 2012, Beijing, China.

TEACHING EXPERIENCE

- Teaching Assistant for COMP3111/3111H: Software Engineering (Fall 2018)
- Teaching Assistant for COMP4111: Software Engineering Practices (Spring 2018)
- Teaching Assistant for COMP4111: Software Engineering Practices (Fall 2016)