Yonathan Fisseha

https://yonathanfisseha.com
yonathan@umich.edu

Research

HashStep: Data and Computation Integrity Enforcement Under Submission

***Y. Fisseha**, *Q. Tan, *S. Chen, M. Demissie, L. Biernacki, J.B. Jeannin, S. Malik, T. Austin* *Under Submission*

Unlike prior work, we provide confidentiality of data, integrity of data, and integrity of computation. We use the cryptographic properties of collision-resistant hashing and message-authentication to ensure only the expected program is executed. We formalize the instruction set's extension as a smallstep semantics and mechanize our main security result in the Lean Theorem Prover.

Formal Verification of Low-trust Architectures 2023

Q. Tan, ***Y. Fisseha, *S. Chen, L. Biernacki, J.B. Jeannin, S. Malik, T. Austin* *CCS*

Low-trust architectures are an emerging class of designs that provide an alternative to expensive homomorphic encryption (HE) techniques. Providing a formal verification of low-trust architectures makes it easier to trust the sensitive hardware components, and, fortunately, the small and isolated component design also makes formal verification viable. The project provides ISA-level semantics proofs that are followed by implementation-level model checking proofs. I was co-first author on this project and we won a *distinguished paper award* at CCS'23.

HardKAT Present

***Y. Fisseha**, S. Chen, J.B. Jeannin, T. Austin* *In Progress*

HardKAT is a formally defined intermediate language (IR) for accelerator design languages (ADL). It is built on the algebraic framework Partially Concurrent Kleene Algebra (POCKA). HardKAT includes many important language features for accelerator design, such as concurrent queues, concurrent global storage, and signals. We show that the semantics is sound and are finalizing a completeness proof. A version of this paper was presented in *TechCon 2022*.

Tabular Sat Present

Y. Fisseha*, J. Chen*, M. Fleury*, A. Biere, S. Karem* *In Progress*

SAT solvers have a rich history driven by multiple annual constraint solving competitions. Consequently, there is a proliferation of techniques to improve performance. In this project, we start with a blank slate and carefully categorize various techniques and build very simple solvers that combine these categories. We are running various large scale performance tests on recent benchmarks from these competitions. "Revisiting Clause Vivification" presents an implementation improvement based on our experiments; presented in the POS'25 workshop by our team members.

Twine: A Chisel Extension for Component-level Heterogeneous Design 2022

*S. Chen **Y. Fisseha**, J.B. Jeannin, T. Austin* *DATE*

Twine is an extension to the Chisel hardware generator language. It improves re-usability and composability of hardware components, allowing designers to stay productive in heterogeneous design. It provides well defined interfaces and automates control-flow communication and data type and bandwidth conversions between components implementing these interfaces. Some of this is accomplished at the library level while some of the conversions require compiler passes in FIRRTL, the underlying IR for Chisel. This work has appeared in DATE'22 as Twine and TechCon 2021 as SimpleChisel.

Non-peer Reviewed Reports

Migrating Browser Automation Programs for End-to-End Testing 2021
***Y. Fisseha**, Y. Li* *University of Michigan*

While web applications have dominated the software development industry, it is still hard to achieve end-to-end testing for complex web architectures. Browser automation allows the tests to input data just like a user would and make assertions on the displayed result. However, the tests driving such automation are brittle and easily break when the frontend of the application changes even in minor ways. Migrating tests from one version of the frontend to a new version is manual process at the moment. This project applies program synthesis techniques supported by a MaxSMT solver to automate a majority of such cases. We demonstrate how to generate a new Python test program that works on the new version of the frontend given the old version of the frontend HTML and the old Python test program.

Compression-aware Algorithms 2020
***Y. Fisseha**, N. Brunelle* *University of Virginia*

Compression is usually paired with a decompression scheme to allow algorithms to compute on the data. This is unfortunate since the decompression step requires both space and time resources. Compression-aware algorithms are a class of algorithms proposed by N. Brunelle where the algorithm is aware that the data is compressed and thus takes the necessary steps to extract the data without fully decompressing the data. In this work, I studied if the LZ-family of string-alignment algorithms can be redesigned to be compression aware. We propose algorithms that work on various class of input strings and experimentally characterize the input strings that can be processed by such algorithms.

Secure Data Summarization for Stream Data 2019
***Y. Fisseha**, A. Hithnawi* *University of California Berkeley*

In this project, we studied if and how (partially) homomorphic encryption can be applied for large-scale streaming data. Such data can properly characterize data from IoT devices, such as sensors. IoT data is temporal by nature, data summarization techniques can be applied to reduce the data size before PHE schemes are applied to secure the data moving forward. We proposed a system design that allows secure summarization of streaming data.

Work Experience

Intel, GPU Compilers Team 2025
Research Intern *Santa Clara, CA*

I formalized a subset of the execution model for the next generation of Intel GPUs. Particularly, I was focused on understanding the behavior of programs under a different progress models at the sub-workgroup level. I used Alloy6 to model the execution by exploiting the recently added temporal logic features. The implementation serves as a first step in the formalization process and also a formal ground-truth for other tools, such as internal simulators.

Intel, GPU Compilers Team 2024
Research Intern *Santa Clara, CA*

I formalized the memory model for pISA, the new Intel GPU abstract instruction set. This involved discussions with various groups across the compilers and architecture teams. The language was modified as various limitations were discovered in the process of formalizing it. The work culminated in a soundness proof for the memory consistency model.

Microsoft Intune 2018
Software Engineer Intern *San Jose, CA*

I designed and implemented Network-Fencing and Geo-Fencing testing tools for the Intune Android client using Hyper-V and Android Mock Locations. The testing tool was integrated into the existing CI infrastructure allowing developers to automatically test code that was previously tested manually.

Impellia 2016-17
Software Developer *Remote*

I worked with the startup implementing multiple projects that leverage algorithms related to athletics designed and licensed by various university. The projects helped validate the algorithms for future products.

University of Colorado School of Medicine 2015-17
Software Developer *Aurora, CO*

I worked with doctors and researchers to quickly validate clinical product concepts through prototypes and user testing. The products involved tracking patient data, numerical analysis, billing, and hour tracking.

Education

University of Michigan 2020-Present
Doctor of Philosophy in Computer Science and Engineering *Ann Arbor, MI*

I'm co-advised by Professor Todd Austin and Professor Jean-Baptiste Jeannin. I currently hold a GSRA position.

University of Virginia 2016-2020
Bachelor of Science in Computer Science with High Distinction *Charlottesville, VA*

Teaching Service

TEALS Volunteer Teacher 2022
Fenton High School *Remote*

I was one of the teaching volunteers at Fenton High School. We are leading a class of 16 students. This is an AP 1 Computer Science full year class using Java. Content covered starts with basics of the Java language leading up to OOP topics and cumulating in recursion.

Lead TEALS Volunteer Teacher 2021
Eppler Junior High School *Remote*

I was the lead volunteer instructor for a class of about 15 students. A full time teacher was in the classroom providing in person support while I gave the lectures and helped students with projects. We covered the Snap language in the first half of the year and transitioned into Python in the second half.

Teaching Assistant 2018-2020
University of Virginia *Charlottesville, VA*

I was a TA for Algorithms, Advanced Software Development, Programming Languages for Web Applications, and Operating Systems. I held office hours 2-4 times a week helping students with homeworks and exam preparation. I also graded exams and various other course assignments.

Awards & Honors

Jane Street Fellowship Finalist 2024
Jane Street Capital

Rackham Merit Fellowship 2020
University of Michigan

University Honors 2019, 2020
University of Virginia

QuestBridge Scholar 2016
QuestBridge

Daniels Scholar 2016
Daniels Fund

Citizenship

Status: *United States Citizen*