I want to build secure systems using cryptography and check the security properties of real-world technology. I'm excited about applying cryptographic tools to allow people to communicate freely and privately because those are fundamental rights under active threat. For instance, I want to build fast censorship-resistant networks and encrypted messaging protocols resilient to "harvest now, decrypt later" adversaries. Similarly, since election systems are a vital part of democracy, I will study them to ensure their integrity and innovate on them to promote greater transparency. In my academic research career, I want to empower and protect users by working on pressing problems and influencing technology policy.

As evidenced by Dr. Daniel Genkin's side-channel attacks like Spectre, a program's behavior may leak enough information about secret values to cause serious vulnerabilities. Last summer, I worked with Dr. Marcel Böhme on **quantifying the information leaked by a program's execution traces** at the Max Planck Institute for Security and Privacy. In particular, we estimated the mutual information (MI) between a program's execution trace outputs and its secret values. State-of-the-art MI estimators perform poorly when the number of samples is small, producing estimates lower than the actual MI and giving users a false sense of security. We used Chao's multinomial distribution estimator, which is effective when few samples are available, to remedy this underestimation. After thoroughly reading about other MI estimators like Jackknife and the Miller-Madow, I implemented them in Python to compare them to our method. When my initial attempt at running our experiments took over 20 hours, I narrowed down the culprit to my Jackknife implementation. I then rewrote it to use fewer subsamples and to process them in parallel, making its runtime complexity linear in the number of observed outputs. Additionally, I tested different subjects in parallel, resulting in a total runtime of just two hours. I plotted our method's MI estimates and their error relative to the ground truth MI as we varied the number of samples considered. Through these plots, I spotted cases where our method unexpectedly performed worse than the baseline naive estimator. We noticed that this issue occurred when we treated undetected events as more likely than detected ones and fixed it with an additional refinement step. In October, we submitted a paper describing our contributions to ICST 2024.

Over the summer of 2022, I worked with Dr. Dan Wallach on implementing Microsoft's ElectionGuard toolkit in TypeScript for web browsers. ElectionGuard enables **end-to-end verifiable elections using homomorphic encryption and zero-knowledge proofs**. Encrypted ballots consist of ElGamal ciphertexts for each candidate, where each plaintext vote corresponds to either a one or a zero. The encrypted ballots also include Chaum-Pedersen proofs that the plaintext votes are valid and add up to one, i.e., precisely one candidate was selected. One of my significant contributions was the robust conversion of ballots and election records to JSON and back. Our goal was close compatibility with the Python reference implementation, which was challenging when the hashes we produced didn't match. I worked my way upwards from the hashes of low-level objects to those of high-level encapsulating objects, adjusting our serialization to match that of the reference. In the process, I encountered and patched a bug in the reference implementation's interface that caused missing selections on random test ballots. I also wrote property-based tests that achieved statement coverage of over 90%, giving us greater confidence in the behavior of our library through fuzzing. Based on my work, we wrote to the ElectionGuard team, encouraging them to explicitly specify edge cases and provide test vectors to promote interoperability. At an ElectionGuard convening at Microsoft Research, Redmond, I collaborated with several community members, including cryptographers and engineers, to improve the ElectionGuard specification. In August 2022, we released a well-tested, production-ready open-source library for use by our partner, Enhanced Voting.

Participating in **Capture the Flag (CTF) competitions** lets me solve fun security challenges with other hackers at my university. I revived Rice's competitive hacking team and club in 2022, building a community of over 50 students since. In my experience playing over 20 CTFs, I practiced reverse engineering with Ghidra, exploit development with pwntools, and packet sniffing with Wireshark. One of my favorite challenges involved looking for the domain associated with a partial TLS certificate. After brainstorming with my teammates, I took a lengthy dive into certificate formats and executed custom queries on a certificate transparency database until I finally found the domain. Our team has placed well in several competitions, including 8th in UTCTF 2023, 21st in idekCTF 2023, and top 15 in the CSAW CTF 2022 US-Canada qualifiers. Thanks to my CTF instincts, I view systems—from auto-grading scripts to identity cards—from an inquisitive and adversarial perspective.

Hoping to introduce more fellow students to CTFs, I **designed and taught a course about CTFs and security basics** in spring 2023. Over twelve weeks, we covered introductory topics in web security, reverse engineering, and more. Each week, I designed 4–8 CTF-style problems that built on the content I covered in class. I kept my lectures engaging by incorporating walkthroughs, demonstrations of tools like GDB, and anecdotes from my CTF experience. At times, adequately motivating the content and homework problems was challenging. In response, I made my material relatable by simulating applications like forums, online shopping websites, and feedback portals. Through my course, I had a chance to convey my enthusiasm for security and encourage my students to dig deeper into CTFs. In my course evaluations, all ten respondents indicated that they either agreed or strongly agreed that they were "challenged to extend [their] capabilities".

Like Dr. Michael A. Specter, I'm interested in applied cryptography and systems security, especially in applications related to public policy. Working on ElectionGuard with Dr. Wallach, I gained a footing in applied cryptography and witnessed the potential of systems that promote transparency and accountability. I would be thrilled to work with Dr. Specter on analyzing election system security, much like his Voatz and OmniBallot work. Many topics covered in his fall 2023 course *Security, Privacy, & Democracy* relate to my research interests, including anti-surveillance technologies and censorship resistance. I would enjoy working with Dr. Specter on the interesting challenges associated with end-to-end encryption, such as sender-anonymous blocklisting and content moderation. I'm excited about applying my background in quantitative information flow to Dr. Daniel Genkin's side-channel attack research. I'm also interested in studying secure multi-party computation and zero-knowledge proofs with Dr. Genkin and using them to build privacy-preserving systems. My enthusiasm for real-world cryptography applications makes me a good fit for working with Dr. Joseph Jaeger. I'm especially excited about solving problems related to encryption and secure messaging with him because of their importance in protecting essential rights—an overarching goal of my research career.