I am an interdisciplinary researcher combining security, operating systems, and software engineering. Complex computer systems are linchpins in modern society — a compromised security vulnerability can spell disaster for small and large organizations and governments alike. Advanced attackers cannot be entirely precluded from gaining unauthorized access to a system. Instead, I believe robust system design requires assuming attackers will gain access at some point and attempt to conceal their behavior. My research focuses on three high-level thrusts to address these concerns: (1) leveraging hardware support and Trusted Execution Environments (TEEs) to conduct *robust system analyses* to understand and defend against advanced evasive malware, (2) adapting and deploying TEEs to detect attacks and remove vulnerabilities in *large systems, cloud infrastructure, and autonomous vehicles*, and (3) empirical human studies to *understand how developers design and comprehend software*. This combination of research has led me to appreciate both rigorous systems building for scalable data collection as well as careful, ethical protocol design for human studies. I will discuss my contributions and methodology associated with each of these thrusts, as well as my plans for future research.

Overall, I have contributed to 32 peer-reviewed publications. I have also led, formulated, and contributed to multiple funded grants from AFRL, MIT Lincoln Laboratory, DARPA, FHWA, Hitachi, and DoE, totaling $7.5M ($2.4M our portion). In addition, I am fortunate to have mentored 18 graduate students and 10 undergraduate students and research engineers leading to 10 peer-reviewed publications. I have published in top conferences across a variety of areas, including IEEE S&P, NDSS, DSN, ICSE, FSE, UbiComp, and EMNLP. I have received an *ACM Distinguished Paper award* for our ICSE 2019 paper as well as a *Best Paper Runner Up Award* for our DSN 2020 paper. Finally, my research contributions have led to one patent and two provisional patents.

Robust System Analysis with Trusted Execution Environments

My research has combined concepts from operating systems, hardware design, and Trusted Execution Environments that has enabled understanding system behavior in the presence of a stealthy, evasive adversary.

My early doctoral research predominantly focused on developing low-level techniques for highly-robust system introspection for the purposes of discovering advanced malicious activity. The security arms race has led to the development of *evasive malware*, which hides malicious behavior by actively detecting or subverting ordinary analyses. When adversaries compromises the operating system, many traditional debugging or analysis techniques can no longer provide reliable information about what the system is doing — an adversary may hide their behavior by leveraging functions or services to check whether analysis tools (e.g., debuggers, virtualization) are present. As a result, I leveraged specialized hardware support and Trusted Execution Environments to measure and understand what systems were doing. First, I helped to build LO-PHI [**?**], a custom FPGA device capable of high-throughput acquisition of system memory dumps and disk behavior. LO-PHI helped collect data about stealthy malware and rootkit execution without perturbing the execution environment. Second, I contributed to Spectre [**?**] and Malt [**?**, **?**], techniques leveraging TEEs on Intel platforms to instrument each instruction executed by a system. Spectre and Malt use Intel System Management Mode to faithfully interrupt execution after each instruction, allowing rich data collection, even with a compromised OS.

Together, these lines of research contributed to advancing the understanding of evasive malware samples that actively detect when they are being analyzed. It led to a patent (10,127,137) and the techniques are part of the basis for one startup company (Allthenticate). Moreover, I led a human study [**?**] demonstrating that security-focused reverse engineers are more productive when given

information provided by both techniques. LO-PHI provides high-throughput acquisition of system state in a way that produced no measurable differences between a plain target system and an instrumented system. On the other hand, Spectre and Malt provides a very fine-grained instrumetation of the entire system — using SMM guarantees that each instruction could be instrumented. As a result, both discovered novel rootkits that, until publication, were previously undetectable with extant approaches. These techniques combined insights from hardware design, operating systems, and binary analysis.

Robust Deployment in Systems, Cloud Infrustructure, and Autonomous Vehicles

My research has been applied to myriad domains. I have worked to secure large cloud infrastructure as well as realtime, resource-constrained autonomous vehicle platforms. My work has also advanced kernel hot-patching, secure authentication, and secure peripheral usage.

While LO-PHI, Spectre, and Malt instrument a single target computer system, my next line of research transitioned these technologies to more distributed, diverse platforms. In particular, I developed Scotch [**?**], a technique that employs TEEs to *secure cloud infrastructure*. Hypervisor software stacks such as Xen are vulnerable against *resource accounting attacks* that permit attackers to steal resources such as CPU time or memory usage from benign guests. As a result, attackers can both (1) effectively disable benign guests, and (2) gain free access to cloud resources. Scotch reliably accounts for resource consumption among potentially-malicious cloud guests that could exploit such vulnerabilities in Xen virtualization.

Next, with the Air Force Research Laboratory, I helped to build trustworthy autonomous rovers and quad-rotor drones adapting Malt and Spectre to apply to this novel domain [**?**]. Attackers may exploit vulnerabilities in the vehicle control software to compromise the vehicle. To address this threat, we built a runtime monitoring system using TEEs on ARM- and x86-based autonomous vehicles that detects when an attacker compromises the vehicle remotely. Our key contribution was that we automatically evolved *and deployed* a repaired version of the software that was immune to the original vulnerability *while the vehicle executed the mission without significant interruption and without human intervention.* We leverage TEEs to execute our monitoring and repair algorithms, allowing us to be sure that we could safely construct a repaired software stack even when an attacker was present.

I have a demonstrated track record of adapting novel research ideas across disciplines and platforms to increase research impact. In addition to cloud infrastructure and autonomous vehicles, my research on robust deployment has led to advances in peripheral security [**?**,**?**], identity security [**?**], and kernel hot-patching [**?**].

Human Studies of Developers

My recent research has involved comprehensive and ethical human subject experiments. I have led studies that investigate neural correlates of software development activities as well as security-focused reverse engineering tasks.

More recently, my work has been driven by the insight that addressing systems security problems is not enough: because software artifacts and decisions are made by human developers, there is a need to understand that decision-making process. I have helped to design, execute, and analye data from a number of studies that leverage functional brain imaging [**?**, **?**, **?**] to understand how software developers comprehend and write code. Ultimately, I believe this line of medical imaging rsearch will affect how we build developer tools and train engineers to create better, more secure software.

In particular, I designed the first study of fMRI-based code writing in which participants used a full QWERTY keyboard to write code in an IDE environment *while undergoing a functional MRI* [**?**]. We built a custom keyboard without metallic components that separated the control logic to allow participants to write code and prose during scans. We found a statistically significant difference in brain activity associated with writing code (strong activation in areas of the brain associated with mental imagery, visual information, and problem solving) compared to writing prose (strong activation in areas associated with spoken language processing and letter recognition). These findings overturned established wisdom about *reading* code versus prose — as a result, I intend to pursue related human studies that focus on problems in computer security and security decision-making.

Future Work

I want to leverage my most recent line of functional brain imaging work to understand security-related decision making in software development. Our findings about the differences between code and prose writing in the brain lead me to believe that the security community could benefit from similar insights. For example, I want to understand what neural processes may contribute to an engineer unintentionally creating a vulnerability. I believe that this understanding will help lead the community to developing better tools, practices, or insights that help engineers (1) produce higher quality, less vulnerable software, and (2) respond more quickly to security-related incidents that will invariably occur in an era of ubiquitous, safety-critical computing.

In the short term, I plan to combine my three research thrusts to gain an understanding of how security engineers comprehend and respond to security-related software decisions. I will extend my medical imaging work (from my third thrust) by examining differences in neural activity between novice engineers and well-practiced security engineers. I intend to carry out a study in which participants are presented with vulnerable autonomous vehicle code (from my second research thrust) and must identify the source of the vulnerability when provided introspection information (from my first research thrust). This will help us understand neural activity associated with noticing vulnerabilities so that we might better target training activities for avoiding them.

In the long term, my vision is to impact how we approach security in computing. Rather than focusing on the development of tools and techniques to prevent or detect attacks and intrusions, I want to understand how the human — the fallible link in so many computer security failures — can be better trained to avoid or respond to software security threats. While my short-term plan focuses on developers, I want to understand *how humans in general approach security.* How does an IT specialist notice that their corporate network has been compromised? How can a pilot tell if their aircraft is under cyber attack? Can we train end users to more quickly notice when their financial data might have been stolen? If we can understand security in computing from this angle, we may better equip society for preventing and addressing critical software security failures. My research program will cross boundaries between software engineering, security, and psychology — this fruitful endeavor will, I believe, change the landscape of security research.

References

(Note: A full list of publications is available in my CV. The list below contains references cited in the text above.)