Research Statement

Connor Glosner

Modern computer systems rely on a layered sequence of hardware and firmware components that execute long before any operating system appears. These early stages establish all subsequent security guarantees, yet remain some of the most vague and least protected parts of computer systems. My research focuses on securing this hardware–firmware boundary by studying how systems establish trust from when the first instruction is executed at power on.

My goal is to improve the security of low-level systems through practical analysis tools, deployable mitigations, and empirical studies that help industry and academia build more trustworthy platforms. To do this, I combine systems security, firmware analysis, and hardware–software interface reasoning to uncover vulnerabilities and strengthen isolation across diverse device classes.

Research Focus

My work asks a central question: *How do we build secure systems starting from a device is powered on?* I pursue this through three directions:

1. **Boot-chain security.** I study how early boot components establish platform trust, since weaknesses here undermine all later defenses.
2. **Analysis and vulnerability discovery tooling.** I develop fuzzers, sanitizers, and verification methods designed for early-boot environments, where traditional tools cannot operate.
3. **Improving firmware security.** I build deployable mitigations such as compiler-backed defenses, hardened memory isolation, and safer hardware–software interfaces.

Research Contributions

My current work spans early-boot analysis, secure compilation, and systematization of firmware security practices.

UEFI Fuzzing and Memory-Safety Instrumentation

UEFI firmware remains difficult to analyze because of its atypical execution environment. I developed fuzzers targeting UEFI components and SMM communication paths, uncovering unsafe data flows and isolation flaws exploitable before the OS loads (*FuzzUEr: Enabling Fuzzing of UEFI Interfaces on EDK-2*). To support this effort, I adapted compiler-based sanitizers (*e.g.*, AddressSanitizer) to UEFI by introducing firmware-compatible shadow memory and custom runtime support, enabling reliable detection of memory errors during development. This work brings modern debugging capabilities into a domain largely unsupported by existing tools.

Compiler-Based Vulnerability Prevention for Firmware

I extended 3C—an automated Checked C conversion tool—to operate within UEFI's freestanding environment (*Adding Spatial Memory Safety to EDK II through Checked C (Experience Paper)*). This required redesigning bounds models, annotations, and runtime assumptions so that type-safe code could compile and execute within firmware constraints. The results show that strong, automated memory-safety guarantees are achievable even in early-boot software.

Bootloader and Boot-Chain Security (SoK)

To contextualize these technical contributions, I conducted a systematization-of-knowledge study across bootloader architectures (*Paper Under Review*). This work identifies shared weaknesses, inconsistent assumptions, and opportunities for standardized mitigations, offering industry a roadmap for strengthening platform trust from power-on through OS launch.

Research Vision

My long-term vision is to build practical foundations for securing every stage of the system lifecycle.

- **Verified trust from power on to OS handoff.** I aim to develop tools that verify correctness and safety across each transition in the boot chain, catching state inconsistencies and privilege-boundary violations invisible to current techniques.
- **Robust hardware software interfaces.** I plan to formalize and enforce interface contracts that reduce reliance on undocumented assumptions in firmware.
- **Scalable security for everyday devices.** I will extend analysis and mitigation frameworks to the embedded systems that underpin consumer and industrial technology (*Current Goal*).

Broader Impact

Strengthening early boot and firmware security reduces the risk of long-lived, difficult-to-detect attacks that threaten privacy, device integrity, and critical infrastructure. By advancing both the scientific understanding and practical tooling for securing the earliest layers of execution, my research supports the development of more transparent and trustworthy computing ecosystems.