





**My research interests are at the intersection of systems security and computer architecture.** Today, security is a foundational requirement for all computing systems: the annual cost of cyber-crime is currently over \$1 trillion (1% of world's economy) and growing. In recent years, several critical vulnerabilities (like Spectre, Rowhammer, and others) have been discovered in computing hardware designs, affecting millions of computers. As software is only as secure as the hardware it runs on, a hardware-agnostic approach to cybersecurity is no longer sufficient. While past decades saw computer architectures optimized primarily for speed and efficiency, security is a first-order metric for architectures of this decade and beyond.

*My research envisions secure hardware designs for future computer architectures, and securing software through co-design with trustworthy hardware, to ensure security is low-cost and widely deployable in systems.*

**My Ph.D. research has enabled principled security for hardware and software systems.** As security vulnerabilities have been discovered at various levels of the hardware and systems stack, my contributions on security have also been across the stack, including in processor caches, main-memory, software runtimes, and testing. My research at the intersection of security and computer architecture has enabled:

- (a) new cache covert-channel attacks that are *the fastest cache attacks in half-a-decade*.
- (b) principled secure cache defenses resilient to data leakage via side-channels that *end an arms race in this space*.
- (c) a memory system hardened against transient data leakage *that has influenced defenses against attacks like Spectre*.
- (d) DRAM integrity solutions *for low-cost detection of DRAM data tampering via physical or Rowhammer attacks*
- (e) a principled Rowhammer mitigation for current & future attacks, and *one of the first for new Rowhammer variants*
- (f) hardware for memory safety & fuzzing, *enabling software resilient to >50% SW vulnerabilities prevalent today*.

		<u>Problem</u>	<u>Contribution</u>
Software	Applications	 <b>Memory Safety Bugs, Fuzz-Testing Coverage</b>	<b>Low-Cost Bug Detection Hardware</b> [ACM-TACO'22, CCS'21]
	DRAM	 <b>Rowhammer, Physical Attacks</b>	<b>Rowhammer Mitigation</b> [ASPLOS'22] <b>DRAM Integrity</b> [HPCA'18, MICRO'18]
Hardware	Caches	 <b>Cache Side-Channel Attacks</b>	<b>Fastest Known Attack</b> [ASPLOS'21] <b>Principled Defenses</b> [SEC'21, SEED'21]
	Processor Core	 <b>Transient Execution Attacks</b>	<b>Preventing Transient Leakage via Caches</b> [MICRO'19]

The common theme across my research has been leveraging computer architecture techniques that were historically used for efficiency and applying them to enable security properties lacking in today's computer architectures. This enables principled security that is also low-cost and practically deployable in future architectures. My research thus bridges the fields of hardware and security and has been published at top conferences in both computer architecture (HPCA, MICRO, ASPLOS) and systems security (USENIX Security, CCS). My research has also been recognized in both areas: with an **IEEE MICRO Top Picks Honorable Mention** and **Georgia Tech IIS Cybersecurity Fellowship**.

In the long term, my research vision is to make secure systems pervasive by building in security at the hardware level. However, this is a challenging proposition, given that new hardware vulnerabilities continue to emerge before existing ones (like Rowhammer or Spectre) get resolved. Over the next half-decade, I envision a research program with the following thrusts: (a) practical and verifiable security in hardware against current and emerging threats, (b) new hardware/software interfaces for discovering micro-architectural side-channel leakage (c) fast and bug-free software execution with hardware support, (d) privacy-preserving execution of machine learning with trustworthy hardware.

## Current Research

### R1: Caches Side-Channel Attacks and Defenses [ASPLOS'21, SEC'21, SEED'21]

Processor caches allow fast access to data on-chip and avoid slow access to main-memory. But this timing difference results in *side-channel* leakage: an adversary program can monitor the shared cache state and learn a victim's data accesses. Cache side-channels can cause security breaches like leaking encryption keys and covert-channel attacks, where a trojan, infecting a system, can use the cache channel to leak data to an external spy.

**Fastest Known Cache Attack:** While analyzing the limitations of current attacks, my research developed a new cache covert channel attack called Streamline [1] (ASPLOS'21), the first cache attack to achieve >1 MB/s transmission rate (3-6x faster than previous best). Streamline's key insight is an asynchronous transmission protocol using a large sequence of cache lines to transmit successive bits, which automatically resets the channel for subsequent transmission without any additional "*cache line flush*" instructions. To our knowledge, Streamline is the fastest attack in more than

half-a-decade. Additionally, its flush-less nature makes it universally applicable to all ISAs, micro-architectures, and environments like Javascript without flush instruction support, unlike prior-best flush-based attacks.

*Impact:* Streamline is the fastest cache attack in half-a-decade and is universally applicable. Already, it is used in emerging transient-execution-attacks from Javascript [2], highlighting need for effective defenses across platforms.

**Principled Secure Cache Defenses:** In recent years, successive defenses for cache side-channels using randomization (which randomize the mapping of addresses to cache locations) were broken by newer attacks. My research has enabled MIRAGE [3] (SEC’21), a principled defense that ended this arms race, by breaking the spatial co-relation between installed addresses and evicted addresses observed by an adversary. The key insight enabling this is intelligent load-balancing hashing of addresses to cache locations, based on power-of-2-random-choices, which completely randomizes evictions from the cache while preserving practical lookup of cached addresses in finite locations. This provides the illusion of a “fully associative randomized cache” to the software where a new address may evict any random cache address. These benefits come without software changes and a low cost of 2% slowdown.

*Impact:* While 2018-2020 saw several defenses broken by half-a-dozen attacks, MIRAGE’s security has been unbroken since its public release in 2020, signaling an end to this arms race in secure randomized caches [4].

**Interfaces for Cache Isolation:** My research has also designed interfaces for software to obtain “cache enclaves” or insulated partitions within the cache free from all cache side-channels. My work, Bespoke Cache Enclaves (BCE) [5] (SEED’21) provides hundreds of such enclaves of customizable sizes in last-level caches. The key insight of BCE is a dynamic cache-indexing function that redirects addresses to allocated cache enclaves. BCE not only provides secure cache partitioning, but its customizable sizing also boosts performance by up to 40%. Thus, it is scalable & more practical than prior solutions, which only provide few partitions or fixed-sized cache allocations. Ongoing research is exploring extensions of BCE to web browsers to prevent cache-attacks threatening user privacy during web browsing.

*Impact:* BCE can potentially enhance all software-sandboxing solutions like Google’s NaCl, Chrome Site-Isolation, or Virtual-Machines / Containers in the cloud and protect software sandboxes from cache attacks.

## **R2: Hardening Memory-System against Transient-Execution Attacks** [MICRO’19]

Transient-execution attacks like Spectre, Meltdown, and others, discovered in 2018, continue to affect millions of high-performance processors today. These attacks exploit processor speculation (a fundamental feature contributing to performance) to illegally access secrets and typically leak these out through transient changes to processor caches. The key insight of my work, CleanupSpec [6] (MICRO’19) is that just as current processors clear speculative register-state on detection of a mis-speculation, if speculative modifications to caches can also be cleared up, then transient leakage via caches can be limited. To achieve this, CleanupSpec tracks and reverses transient cache-state changes or randomizes them to obfuscate them. This technique hardens the memory system against transient attacks with minimal performance impact (5% slowdown), 3x-10x lower than prior solutions disabling speculation or caching.

*Impact:* CleanupSpec is one of the first hardware mitigations after the public disclosure of transient-execution attacks like Spectre and has influenced subsequent defenses explored by cloud providers, like Microsoft Azure [7].

## **R3: DRAM Integrity: Prevents Rowhammer/Physical Attacks.** [HPCA’18, MICRO’18, ASPLOS’22]

Physical attacks on DRAM in datacenters and software-based attacks like Rowhammer (which flip bits in data) pose a severe data-integrity threat for DRAM today. The Rowhammer threat only worsens as technology scales (modules today are 10x prone than 5 years ago). Secure memories, such as Intel’s SGX, provide DRAM integrity with cryptographic “signatures” that detect data tampering. However, such solutions have limited the DRAM Integrity protection to small memory footprints due to the significant performance overheads of accessing the security metadata.

My research enables efficient organizations for secure memory, providing >30% better performance, >30% energy-saving, 2x lower storage, and 180x better reliability. Synergy [8] (HPCA’18) rethinks the organization of datacenter memories provisioned with error-correction-codes (ECC) and co-designs data integrity with ECC. This allows reusing ECC resources for data integrity and improves performance by 20% and reliability by 185x. My subsequent work Morph [9] (MICRO-2018) reduces the storage overhead by 2x (and boosts performance by 14%), by applying compression to Merkle Trees, a security metadata preventing physical data-tampering via replaying old signatures.

My current work focuses on mitigating Rowhammer attacks, as DRAM’s vulnerability to it is only worsening. My solution targets the root cause of Rowhammer bit-flips: charge leakage on rapid access to a single location (aggressor). As mitigation, my solution, Randomized Row Swap (RRS) [10] (ASPLOS-2022) successively remaps locations of aggressors in memory to limit charge leakage in any single location. This allows principled mitigation of current and future Rowhammer attacks and is one of the first defenses against emerging attacks like Google’s Half-Double attack.

*Impact:* Synergy enables the standard design for DRAM-integrity in datacenter memories by boosting reliability and performance and received IEEE Micro Top-Picks Honorable Mention for “potential long-term impact”. RRS is one of the first defenses for emerging Rowhammer attacks like Google’s Half Double, which breaks several prior defenses.

**R4: Hardware Support for Safe and Bug-Free Software Execution** [TACO’22, CCS’21]

Memory safety bugs like buffer-overflow and use-after-free are the leading cause of vulnerabilities in production software (>50% for Microsoft and Google). These problems have continued to plague C/C++ for several decades as software-based solutions have had limited adoption due to limited coverage, high overheads, or invasive changes.

In independent research collaborations, I developed hardware for low-cost and minimally invasive software bug detection. With IBM Research, my work developed HeapCheck [11] (TACO’22), a low-cost memory-safety checker in hardware for heap objects, where a majority of bugs occur. HeapCheck guarantees spatial and temporal safety using SW/HW co-design for bounds-checking to limit slowdown to <2%, enabling always-on safety for production software.

With security researchers at Georgia Tech, I designed SNAP [12] (CCS’21), one of the first RISC-V hardware-based frameworks to improve performance and precision of coverage-guided fuzz-testing, the standard for software bug-discovery at test-time. SNAP enables 228x higher fuzzing throughput than current software-based solutions by introducing coverage tracking in hardware and uses micro-architectural state to enable more precise coverage tracking. SNAP promises better coverage and significant cost savings for fuzzing services like Google’s OSS-Fuzz.

*Impact:* This research can enable always-on-safety for in-production software and make low-cost fuzzing pervasive at test-time. Together, these works define a new role for hardware to play in enabling a safe software ecosystem.

**Future Research**

My research has two key themes: security for hardware, i.e., mitigating attacks on vulnerable hardware, and hardware for security, i.e., hardware support to secure software at low cost. On these lines, I plan the following research thrusts:

**T1: Hardware-Based Safety for Emerging Programming Languages:** Guaranteeing that software is safe and bug-free is critical for secure systems. Recently, there is a growing preference towards safe languages like Rust (with built-in memory safety checks) for browser and kernel development. However, such languages can have high performance overhead due to the SW-based safety checks and suffer from security bugs in “unsafe” code (where no safety checks are inserted). Building on my past research on hardware for memory safety [11,12], my future research will explore hardware-based safety checks for Rust programs. This research will explore on-demand HW-enforced safety checks to boost Rust performance. This also provides better security as safety checks can be enabled in HW even for “unsafe” code where Rust provides no safety. With better performance and security, this research can accelerate Rust adoption.

**T2: HW/SW Interfaces for Side-Channel Discovery in Software:** As per current hardware-software interfaces (ISA), semantically correct software is still vulnerable to hardware side-channels and transient execution exploits. Future hardware (incorporating mitigations from my [3,5,6] and others’ research) will be more resilient, but the aggregated performance costs of defenses can be prohibitive. Additionally, some software may not even be vulnerable as they do not have the gadgets required for leakage and may not desire the cost of such mitigations. Software-based fuzzing is a promising technique for automatic detection of hardware side-channels. However, such software techniques are slow and ineffective because they lack the rich micro-architectural information in hardware. My future research will explore how ISAs like RISC-V can be enhanced with new interfaces to expose micro-architectural state (e.g., branch-predictor state) and timing influences (e.g., cache state) to software. This research will enable faster discovery of side-channels in software and surgical invocation of mitigations to minimize costs. This can also discover new, previously unknown side-channels, as fuzzing becomes more precise with microarchitectural state, as shown in my past research [12].

**T3: Practical and Verifiable Mitigations of Emerging Hardware Vulnerabilities:** Hardware vulnerabilities like Rowhammer are here to stay and likely to worsen in newer DRAM modules, as the root cause is fundamental interference arising from DRAM scaling. Several recent attacks like ECCploit, TRRespass, and Half-Double have broken deployed DRAM defenses, and no principled and practical solutions currently exist. Similarly, new side-channels and transient-execution attacks are likely to be discovered as our understanding of the complexity in today’s hardware increases. This drives the continued need for new hardware security solutions. My future research will explore hardware security primitives like resource isolation for prevention [5], randomization for obfuscation [3], and principled detection [8] of fault-injection and micro-architectural interferences, building on my past research. The key focus of this research will be verifiability to enable future-proof defenses with accelerated commercial adoption.

**T4: Privacy-Preserving Machine Learning with Trustworthy Hardware:** Application domains like machine learning (ML) and healthcare involving data-intensive computations desire both data privacy and high performance. Trusted-execution environments (TEEs) in hardware provide data privacy but are currently not well-suited for ML applications due their inflexibility: TEEs are restricted to CPUs and unable to leverage GPUs or domain-specific accelerators, leading to poor performance. This research will explore customizable TEEs with flexible resource allocation for data-intensive applications, building on my prior work on bespoke enclaves [5]. The key insight enabling customizable TEEs is that data-intensive tasks are often repetitive; hence profile-guided resource partitioning and access control can ensure privacy while boosting performance in the absence of contention. This research will enable customizable TEEs which utilize GPUs and accelerators on-demand, providing speed and privacy for data-intensive applications.

## References

1. **G Saileshwar**, C Fletcher, and M. Qureshi. "Streamline: a Fast, Flushless Cache Covert-channel Attack by Enabling Asynchronous Collusion." In 26th ACM International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS'21), 2021.
2. Hany Ragab, et al. "Rage Against the Machine Clear: A Systematic Analysis of Machine Clears and Their Implications for Transient Execution Attacks." In 30th USENIX Security Symposium (SEC'21), 2021.
3. **G Saileshwar** and M Qureshi. "MIRAGE: Mitigating Conflict-Based Cache Attacks with a Practical Fully-Associative Design". In 30th USENIX Security Symposium (SEC'21), 2021.
4. **G Saileshwar**. "Battle for Secure Caches: Attacks and Defenses on Randomized Caches". ACM SIGARCH Blog, <https://www.sigarch.org/battle-for-secure-caches-attacks-and-defenses-on-randomized-caches/>, 2021.
5. **G Saileshwar**, S Kariyappa, and M. Qureshi. "Bespoke Cache Enclaves: Fine-Grained and Scalable Isolation from Cache Side-Channels via Flexible Set-Partitioning." IEEE International Symposium on Secure and Private Execution Environment Design (SEED'21), 2021.
6. **G Saileshwar** and M Qureshi. "CleanupSpec: An Undo Approach to Safe Speculation". In IEEE/ACM International Symposium on Microarchitecture (MICRO'19), 2019.
7. **G Saileshwar** and M Choudhary. Microsoft. "Speculative information flow tracking". US Patent App. 16/669,027, 2021.
8. **G Saileshwar**, P Nair, P Ramrakhyani, W Elsasser, M Qureshi "SYNERGY: Rethinking Secure-Memory Design for Error-Correcting Memories". In IEEE International Symposium on High Performance Computer Architecture (HPCA'18), 2018.
9. **G Saileshwar**, P Nair, P Ramrakhyani, W Elsasser, J Joao, M Qureshi. "Morphable Counters: Enabling Compact Integrity Trees for Low-Overhead Secure Memories". In IEEE/ACM International Symposium on Microarchitecture (MICRO'18), 2018.
10. **G Saileshwar**, B Wang, M. Qureshi, P Nair. "Randomized Row Swap: Mitigating Rowhammer by Breaking Spatial Correlation Between Aggressor and Victim Rows." In 27th ACM International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS'22), 2022. (*To Appear*)
11. **G Saileshwar**, R Boivie, T Chen, B Segal, A Buyuktosunoglu. "HeapCheck: Low-Cost Hardware Support for Memory Safety". In ACM Transactions on Architecture and Code Optimization (TACO'22), 2022. (*To Appear*)
12. R Ding, Y Kim, F Sang, W Xu, **G Saileshwar** and T Kim. "Hardware Support to Improve Fuzzing Performance and Precision". In 28<sup>th</sup> ACM Conference on Computer and Communications Security (CCS'21), 2021.