Matthew Hicks - Research Accomplishments

Associate Professor, Computer Science
Virginia Tech

January 2026

1 Hardware Security

Overview of the research area:

- 24 papers [5, 29, 6, 20, 1, 42, 43, 2, 28, 32, 30, 31, 15, 7, 8, 35, 33, 10, 9, 22, 11, 13, 12, 23]
- 1,690 citations
- Awards:
  - 2025 Communications of the ACM Research Highlights article [8, 13]
  - 2023 Arm's response UnTrustZone [10] `https://developer.arm.com/documentation/109291/latest`
  - 2022 Top Picks in Hardware and Embedded Security [42]
  - 2021 NSF CAREER Award on novel analog-domain hardware Trojans and a defense mechanism
  - 2021 DARPA Director's Fellowship on transistor aging based attacks
  - 2020 R&D 100 award for a routing-congestion-based technique to defend against untrusted foundries [32]
  - 2019 DARPA Young Faculty Award on uncovering and erasing latent analog-domain information in memory circuits
  - 2017 Communications of the ACM research highlights article [42, 43]
  - 2016 IEEE Security and Privacy Best Paper Award [42]
  - 2016 Pwnie Awards Most Innovative Research Award finalist [42]
  - 2011 Invited USENIX login; article [5]
- Design-time hardware Trojans and defenses:
  - First to propose the idea of security-oriented functional verification [5]
  - First to provide a way to isolate untrusted circuits and make forward progress at the software level by routing execution around the isolated circuits [5]
  - First to point-out the attackers need to avoid impacting functionality during functional verification [5]
  - First to present the MUX model of hardware Trojan integration that models the Trojan trigger's decision to insert the payload or not [5]
  - Created the most stealthy and controllable design-time hardware Trojans [5]
  - First to show that it is not possible to guarantee that a hardware design in free from hardware Trojans, short of full functional verification [29]
  - First to build a real-world-processor-errata-inspired dynamic verification of security-critical invariants to protect against hardware Trojans and hardware security vulnerabilities [6]
  - First to propose the idea of post-fabrication security-oriented processor patching [28]
  - First to propose hardware Trojan-specific security-oriented verification [31]
  - Expanded the ticking time-bomb hardware Trojan attacker design space by 50% [31]
  - First to formally model a class of design-time hardware Trojan [31]
  - The first to propose the idea of translating hardware descriptions to software implementations for use in existing software testing tools as opposed to bringing, in piece-wise fashion, software testing advancements to existing hardware testing tools [33]
  - The first open source hardware fuzzer [33]

  - The highest performance and most effective fuzzer for Verilog designs [33]
- Fabrication-time hardware Trojans and defenses:
  - First hardware Trojan to exploit analog-domain functionality to create an ultra stealthy, but attacker-controllable hardware Trojan [42]
  - First tool to quantify the hardware Trojan design space for an integrated circuit [30]
  - First to propose using routing congestion to prevent the insertion of fabrication-time hardware Trojans [32]
- Other hardware security:
  - First paper to show that a consequence of approximate computing is the exposure of analog-domain artifacts that attackers can used to eliminate user privacy [20]
  - First software-only defense to rowhammer attack [1]
  - First attack to exploit separated voltage domains as a short-term data retention attack [8]
  - First steganographic channel using systematic imprinting of data in SRAM's analog domain (as revealed through power-on states) [7]
  - First long-term data retention attack on Arm TrustZone devices [10]
  - First paper to propose and implement a time-dependent Physical Unclonable Function (PUF), enabling time-dependent functionality [9]
  - First hardware watermarking scheme to include cross-device, per-device, and time-dependent watermarks [9]
  - First violation of Texas Instruments IP encapsulation technology that enables arbitrary read, write, and execute access to secure memory [22]
  - First use of precisely controlled interrupt timing to create return oriented programming gadgets in a trusted execution environment [22]
  - First to leverage the relatively stable manufacturing-time variation inherent to otherwise digital circuits as a Physical Unclonable Function (PUF) to fingerprint FPGAs [11]
  - First to use SRAM data retention voltage as an indicator of device age [12]
  - First to use SRAM data retention time as a controllable, fine-grain mechanism to control system voltage to measure device age [12]
  - First to use a large data set of real new and aged devices to model the impact of transistor aging on key analog domain properties [12]
  - First paper to close the confused deputy loophole [22] that undermines all commercial and academic embedded device trusted execution environments [23]

2 Energy Harvesting/Intermittent Computation

Overview of the research area:

- 13 papers [34, 4, 14, 40, 41, 15, 18, 19, 24, 37, 36, 38, 39]
- 609 citations
- Awards: ISCA 50-year Retrospective selected paper [4]
- First to relate challenges in intermittent computation to database ACID properties; observe that memory durability differences cause consistency problems during re-execution of a code region [34]
- First to use idempotence as a central support mechanism for intermittent computation [34]
- First automatic (i.e., compiler only) intermittent computation system [34]
- First hardware support mechanism that allows intermittent computation without processor modification [4]
- First hardware support for tracking software-level idempotence [4]
- First and only intermittent computation system that runs on the more ubiquitous and highest performance Flash-based devices [40]
- First to propose using latent charge in the system's energy storage capacitor to retain software state [40]
- First to analyze and leverage properties about the off time of energy harvesting systems [40]
- First to use energy harvesting to increase software security through side-channel isolation and quanti-

zation [14]
- First comprehensive analysis of the security requirements of checkpointing mechanisms for intermittent computation/energy harvesting systems [24]
- First deployable dynamic energy buffering mechanism that provides responsiveness and persistence in the presence of unpredictable energy availability and dynamic software [37]
- First true random number generator and physical uncloanable function for energy harvesting devices [15]
- First testbed of ultra-low size, weight, and power devices targeted at energy harvesting deployments [38]
- First to use spare SRAM main memory as a software managed instruction cache for embedded devices [39]
- First continuous checkpointing intermittent computation support system that exploits SRAM's time-dependent non-volatility [36].

3 Mutational Coverage-guided Fuzz Testing

Overview of the research area:

- 8 papers [16, 26, 17, 33, 25, 27, 21, 3]
- 555 citations
- Awards: Best poster award [25]
- First to expose coverage tracing as the primary source of run-time overhead in fuzzing [16]
- First to show that after even a short time fuzzing, most test cases are uninteresting and discarded without impacting the fuzzing control loop [16]
- First to propose the idea of encoding the code coverage frontier in the binary itself to avoid coverage tracing in the common case of uninteresting test cases [16]
- The fast fuzzing system for binary-only fuzzing in the world, improving existing systems 7x [16]
- The first fuzzing-oriented black-box binary rewriter [26]
- The first to propose making fuzzing-specific binary transformations a first-class abstraction in a binary rewriter [26]
- The fastest binary rewriter for fuzzing use cases, improving upon existing rewriters by 2x [26]
- The first to get both edge and hit count code coverage without always-on coverage tracing [17]
- The highest performance and most effective software fuzzer [17]
- The first hardware fuzzer to leverage existing software fuzzers [33]
- The first to propose the idea of translating hardware descriptions to software implementations for use in existing software testing tools as opposed to bringing, in piece-wise fashion, software testing advancements to existing hardware testing tools [33]
- The first open source hardware fuzzer [33]
- The highest performance and most effective fuzzer for Verilog designs [33]
- The first target-tailored directed fuzzer [25]
- The first to show the dangers of state-of-the-art persistent fuzzing [27, 21]
- The first to propose the idea of self-resetting programs that undo persistent state changes for correct single-process fuzzing [27, 21]
- The highest performance, most effective fuzzer for Windows and Linux programs [27, 21]
- First fuzzer designed specifically for Digital Signal Processors (DSPs) [3]
- The first variant of coverage-guided tracing [16, 17] that combines the oracle and coverage binary and dynamically re-instruments the combined binary as new coverage is found [3]
- The first semi-hosted fuzzer to push most fuzzing tasks on-device, minimizing host-device communication and coordination [3]

4 Open-Source Research Artifact Repositories

References

[1] Zelalem Birhanu Aweke, Salessawi Ferede Yitbarek, Rui Qiao, Reetuparna Das, Matthew Hicks, Yossi Oren, and Todd Austin. ANVIL: Software-Based Protection Against Next-Generation Rowhammer Attacks. In *International Conference on Architectural Support for Programming Languages and Operating Systems*, ASPLOS, pages 743–755, March 2016.

[2] Evan Chavis, Harrison Davis, Yijun Hou, Matthew Hicks, Salessawi Ferede Yitbarek, Todd Austin, and Valeria Bertacco. SNIFFER: A high-accuracy malware detector for enterprise-based systems. In *IEEE International Verification and Security Workshop*, IVSW, pages 70–75, July 2017.

[3] Cameron Garcia and Matthew Hicks. Signal breaker: Fuzzing digital signal processors. In *International Conference on Architectural Support for P rogramming Languages and Operating Systems*, ASPLOS, March 2026.

[4] Matthew Hicks. Clank: Architectural support for intermittent computation. In *ACM/IEEE Annual International Symposium on Computer Architecture*, ISCA, pages 228–240, June 2017.

[5] Matthew Hicks, Murph Finnicum, Samuel T. King, Milo M. K. Martin, and Jonathan M. Smith. Overcoming an Untrusted Computing Base: Detecting and Removing Malicious Hardware Automatically. In *IEEE Symposium on Security and Privacy*, Oakland, pages 159–172, May 2010.

[6] Matthew Hicks, Cynthia Sturton, Samuel T. King, and Jonathan M. Smith. SPECS: A Lightweight Runtime Mechanism for Protecting Software from Security-Critical Processor Bugs. In *International Conference on Architectural Support for Programming Languages and Operating Systems*, ASPLOS, pages 517–529, March 2015.

[7] Jubayer Mahmod and Matthew Hicks. Invisible bits: Hiding secret messages in sram's analog domain. In *International Conference on Architectural Support for Programming Languages and Operating Systems*, ASPLOS, March 2022.

[8] Jubayer Mahmod and Matthew Hicks. Sram has no chill: Exploiting power domain separation to steal on-chip secrets. In *International Conference on Architectural Support for Programming Languages and Operating Systems*, ASPLOS, March 2022.

[9] Jubayer Mahmod and Matthew Hicks. Sram imprinting for system protection and differentiation. In *ACM ASIA Conference on Computer and Communications Security*, AsiaCCS, July 2024.

[10] Jubayer Mahmod and Matthew Hicks. Untrustzone: Systematic accelerated aging to expose on-chip secrets. In *IEEE Symposium on Security and Privacy*, Oakland, May 2024.

[11] Jubayer Mahmod and Matthew Hicks. Phaseprint: Exposing cloud fpga fingerprints by inducing timing faults at runtime. In *International Conference on Architectural Support for Programming Languages and Operating Systems*, ASPLOS, pages 831–844, March 2025.

[12] Jubayer Mahmod and Matthew Hicks. Retain the date: Detecting recycled chips in the supply chain through sram's data retention behavior. In *Annual Computer Security Applications Conference*, ACSAC, December 2025.

[13] Jubayer Mahmod and Matthew Hicks. SRAM has no chill: Exploiting power domain separation to steal on-chip secrets. *Communications of the ACM*, 68(8):82–90, July 2025.

[14] Michael Moukarzel and Matthew Hicks. Reap What You Store: Side-channel Resilient Computing Through Energy Harvesting. In *ACM International Workshop on Energy Harvesting and Energy-Neutral Sensing Systems*, ENSsys, pages 21–26, November 2017.

[15] Michael Moukarzel and Matthew Hicks. Ringram: A unified hardware security primitive for iot devices that gets better with age. In *Annual Computer Security Applications Conference*, ACSAC, December 2021.

[16] Stefan Nagy and Matthew Hicks. Full-speed fuzzing: Reducing fuzzing overhead through coverage-guided tracing. In *IEEE Symposium on Security and Privacy*, Oakland, pages 787–802, May 2019.

[17] Stefan Nagy, Anh Nguyen-Tuong, Jason Hiser, Jack Davidson, and Matthew Hicks. Same coverage, less bloat: Accelerating binary-only fuzzing with coverage-preserving coverage-guided tracing. In *ACM Conference on Computer and Communications Security*, CCS, November 2021.

[18] Fariborz Lohrabi Pour, Harrison Williams, Matthew Hicks, and Dong Sam Ha. Practical considerations of energy harvesting source in minimization of age of information with updating erasures. In *IEEE International Symposium on Circuits and Systems*, ISCAS, May 2023.

[19] Fariborz Lohrabi Pour, Harrison Williams, Matthew Hicks, and Dong Sam Ha. Rf energy harvesting in minimization of age of information with updating erasures. In *IEEE International Symposium on Circuits and Systems*, ISCAS, May 2023.

[20] Amir Rahmati, Matthew Hicks, Daniel E. Holcomb, and Kevin Fu. Probable Cause: The Deanonymizing Effects of Approximate DRAM. In *ACM/IEEE Annual International Symposium on Computer Architecture*, ISCA, pages 604–615, June 2015.

[21] Rishi Ranjan, Ian Paterson, and Matthew Hicks. ClosureX: Compiler support for correct persistent fuzzing. In *International Conference on Architectural Support for Programming Languages and Operating Systems*, ASPLOS, pages 151–163, March 2025.

[22] Prakhar Sah and Matthew Hicks. RIPencapsulation: Defeating IP encapsulation on TI MSP devices. In *USENIX WOOT Conference on Offensive Technologies*, WOOT, pages 117–132, August 2024.

[23] Prakhar Sah and Matthew Hicks. Sheriff: Arresting the confused deputy undermining ultra-low swap device trusted execution environments. In *ASIA Conference on Computer and Communications Security*, AsiaCCS, June 2026.

[24] Prakhar Sha and Matthew Hicks. Hitchhiker's guide to secure checkpointing on energy-harvesting systems. In *ACM International Workshop on Energy Harvesting and Energy-Neutral Sensing Systems*, ENSsys, November 2023.

[25] Prashast Srivastava, Stefan Nagy, Matthew Hicks, Antonio Bianchi, and Mathias Payer. One fuzz doesn't fit all: Optimizing directed fuzzing via target-tailored program state restriction. In *Annual Computer Security Applications Conference*, ACSAC, December 2022.

[26] Stefan Nagy, Anh Nguyen-Tuong, Jason Hiser, Jack Davidson, and Matthew Hicks. Breaking Through Binaries: Compiler-quality Instrumentation for Better Binary-only Fuzzing. In *USENIX Security Symposium*, USENIX, pages 1–15, August 2021.

[27] Leo Stone, Rishi Ranjan, Stefan Nagy, and Matthew Hicks. No linux, no problem: Fast and stateful windows binary fuzzing via target-embedded snapshotting. In *USENIX Security Symposium*, USENIX, August 2023.

[28] Cynthia Sturton, Matthew Hicks, Samuel T. King, and Jonathan M. Smith. FinalFilter: Asserting Security Properties of a Processor at Runtime. *IEEE Micro*, 39(4):35–42, July 2019.

[29] Cynthia Sturton, Matthew Hicks, David Wagner, and Samuel T King. Defeating UCI: Building stealthy and malicious hardware. In *IEEE Symposium on Security and Privacy*, Oakland, pages 64–77, May 2011.

[30] Timothy Trippel, Kang Shin, Kevin B. Bush, and Matthew Hicks. ICAS: an Extensible Framework for Characterizing the Fabrication-Time Attack Surface of Integrated Circuit Layouts. In *IEEE Symposium on Security and Privacy*, Oakland, pages 1742–1759, May 2020.

[31] Timothy Trippel, Kang Shin, Kevin B. Bush, and Matthew Hicks. Bomberman: Defining and Defeating Hardware Ticking Timebombs at Design-time. In *IEEE Symposium on Security and Privacy*, Oakland, pages 1–17, May 2021.

[32] Timothy Trippel, Kang G. Shin, Kevin B. Bush, and Matthew Hicks. T-TER: Defeating A2 trojans with targeted tamper-evident routing. In *ACM ASIA Conference on Computer and Communications Security*, AsiaCCS, July 2023.

[33] Timothy Trippel, Kang G. Shin, Alex Chernyakhovsky, Garret Kelly, Dominic Rizzo, and Matthew Hicks. Fuzzing hardware like software. In *USENIX Security Symposium*, USENIX, August 2022.

[34] Joel Van Der Woude and Matthew Hicks. Intermittent Computation without Hardware Support or Programmer Intervention. In *USENIX Conference on Operating Systems Design and Implementation*, OSDI, pages 17–32, November 2016.

[35] Xin Wang, Daulet Talapkaliyev, Matthew Hicks, and Xun Jian. Self-reinforcing memoization for cryptography calculations in secure memory systems. In *International Symposium on Microarchitecture*, MICRO, October 2022.

[36] Harrison Williams, Saim Ahmad, and Matthew Hicks. A difference world: High-performance, NVM-invariant, software-only intermittent computation. In *USENIX Annual Technical Conference*, ATC, pages 1223–1238, July 2024.

[37] Harrison Williams and Matthew Hicks. Energy-adaptive buffering for efficient, responsive, and persistent batteryless systems. In *International Conference on Architectural Support for Programming Languages and Operating Systems*, ASPLOS, March 2024.

[38] Harrison Williams and Matthew Hicks. A survey of prototyping platforms for intermittent computing research. In *International Workshop on Energy Harvesting and Energy-Neutral Sensing Systems*, ENSsys, pages 8–14, November 2024.

[39] Harrison Williams and Matthew Hicks. A software caching runtime for embedded NVRAM systems. In *International Conference on Architectural Support for Programming Languages and Operating Systems*, ASPLOS, pages 1–16, March 2025.

[40] Harrison Williams, Xun Jian, and Matthew Hicks. Forget Failure: Exploiting SRAM Data Remanence for Low-Overhead Intermittent Computation. In *International Conference on Architectural Support for Programming Languages and Operating Systems*, ASPLOS, pages 69–84, March 2020.

[41] Harrison Williams, Michael Moukarzel, and Matthew Hicks. Failure Sentinels: Ubiquitous just-in-time intermittent computation via hardware support for continuous, low-cost, fine-grain voltage monitoring. In *International Symposium on Computer Architecture*, ISCA, June 2021.

[42] Kaiyuan Yang, Matthew Hicks, Qing Dong, Todd Austin, and Dennis Sylvester. A2: Analog malicious hardware. In *IEEE symposium on security and privacy*, Oakland, pages 18–37, May 2016.

[43] Kaiyuan Yang, Matthew Hicks, Qing Dong, Todd Austin, and Dennis Sylvester. Exploiting the analog properties of digital circuits for malicious hardware. *Communications of the ACM*, 60(9):83–91, 2017.