

# HPC-Based Malware Detectors Actually Work: Transition to Practice After a Decade of Research

**Charalambos Konstantinou**

Computer, Electrical and Mathematical Science and Engineering Division,  
King Abdullah University of Science and Technology (KAUST),  
Thuwal 23955-6900, Saudi Arabia

**Xueyang Wang**

Alibaba Group, Shanghai 201210, China

**Prashanth Krishnamurthy and Farshad Khorrami**

Department of Electrical and Computer Engineering,  
New York University Tandon School of Engineering,  
Brooklyn, NY 11201 USA

**Michail Maniatakos**

New York University Abu Dhabi,  
Abu Dhabi 129188, United Arab Emirates

**Ramesh Karri**

Department of Electrical and Computer Engineering,  
New York University Tandon School of Engineering,  
Brooklyn, NY 11201 USA

*Editor's notes:*

This article details how hardware performance counters can help detect malware in a system, along with a case study on power grid infrastructure.

—Jeyavijayan “JV” Rajendran, Texas A&M University

instructions retired. HPCs are available on all modern processors, including Intel, ARM, AMD, and Nvidia. Intel processors incorporate HPCs into the performance monitoring

■ **MODERN AND LEGACY** processors include dedicated registers, called *hardware performance counters* (HPCs), to track low-level microarchitectural events to monitor and measure events of processes executing on the system. Such events include, for example, number of branch-misses, CPU cycles, and

toring unit (PMU). The types of events and the number of HPCs vary among different processor architectures. Recent CPUs from Intel can record over 1,000 events including tens of events related to L3 cache misses, data translation lookaside buffer (DTLB) load misses, and resource stalls. HPCs were introduced and primarily used to debug and tune application performance using detailed performance-related data that they yield with low overhead and high accuracy compared to software profilers.

Digital Object Identifier 10.1109/MDAT.2022.3143438

Date of publication: 14 January 2022; date of current version: 22 June 2022.

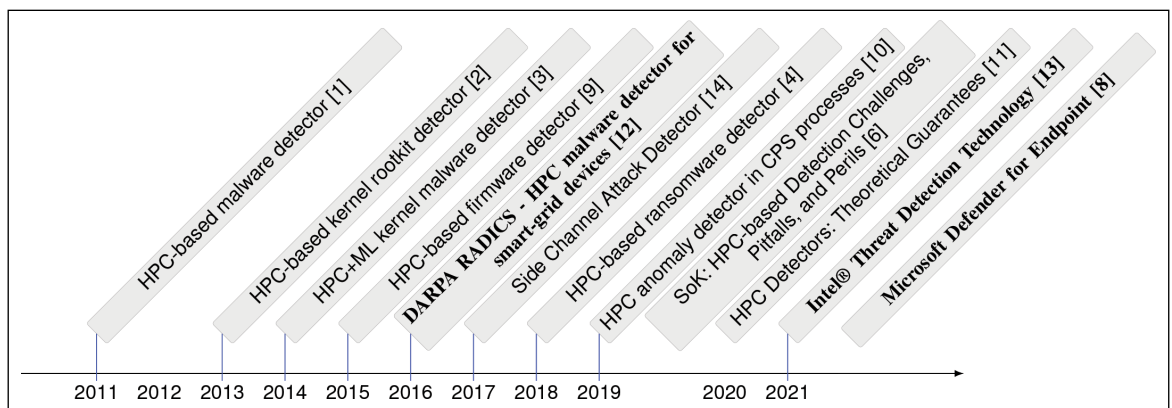
Over a decade ago, in 2011, researchers proposed using HPCs to evaluate static and dynamic integrity of program codes to detect malicious program modifications at load time and at run time, respectively [1]. Extending this paradigm, HPCs can be used for offensive and defensive purposes: they can detect malicious firmware and software [2], [3], ransomware [4], and cryptojacking [5]. Recently, a systematization of knowledge survey paper summarized HPC-based detectors, attacks, and scenarios, pointing out ideal use cases of HPCs as well as highlighting drawbacks and scenarios where HPCs do not provide the expected benefits [6]. After approximately 10 years of academic research, however, Intel productized HPC-based threat detection [7] and Microsoft is employing it in its contemporary security products [8].

In this article, we trace important milestones in how HPCs have been used in research and present case studies of industry and government adoption. Figure 1 presents the chronology of transition of research, on malware detection using HPCs, to practice. Since their introduction for malware detection [1], HPCs have been used to detect variants of malware such as kernel-level rootkits [2], [3], firmware modifications [9], and malware in multithreaded cyber-physical system (CPS) processes [10]. Studies also demonstrated the theoretical guarantees of HPC detectors [11]. The case studies of DARPA Rapid Attack Detection, Isolation and Characterization Systems (RADICS) [12], Intel Threat Detection Technology (TDT) [13], and Microsoft Defender [8] show the transition of academic research to practice (TTP).

## HPCs for embedded system security

Existing work in hardware-assisted technologies has shown that such techniques can be utilized for security purposes in terms of *malware prevention and detection* purposes. For instance, hardware-based trusted execution environments (TEEs), including Intel Software Guard Extensions (SGX), ARM TrustZone, and AMD Secure Encrypted Virtualization (SEV), built into computing and server platforms from different hardware “industry” vendors, offer a secure execution space that can *prevent* malware and provide a higher level of security for trusted applications running on the computing device. There are also “academic” efforts toward providing hardware-assisted root-of-trust, allowing attestation and secure task loading at run time, for example, TyTan [15], C-FLAT [16], Keystone [17], and so on. As for hardware-assisted malware *detection*, existing work has been focused on collecting hardware traces from different avenues, such as the Embedded Trace Buffer (ETB) and HPCs [18]. Among hardware-based prevention and detection solutions, a few of them apply to embedded systems due to the overlooked security risks and critically the high costs for hardware modifications. In this work, we focus on HPCs to track microarchitectural events which are utilized towardfor malware detection.

Since HPCs are available in all modern processors, including Intel x86 and x86-64, ARM, MIPS, PowerPC, and Nvidia GPUs, they can yield near-zero-overhead approach to count hardware events of applications running on the platform. Therefore, HPCs offer an attractive and flexible capability to verify application integrity with negligible performance overhead.



**Figure 1. Chronology of transition of research on HPC-based malware detection to practice.**

While it is not possible to read all available HPCs at the same time, one can time-multiplex the measurements to read more HPC measurement streams, albeit with the overhead of multiplexing. Code execution can be characterized, for instance, by total occurrences of hardware events and by temporal patterns and relations among events. NYU researchers pioneered the use of HPCs to detect malicious modifications [1], rootkits [19], [20], and firmware modifications [9], [21].

1) *ConFirm*: In 2015, researchers extended HPC-based malware detection to detect firmware modifications in embedded systems using HPCs [9]. The motivation for ConFirm is that embedded devices are integrated in several domains, including power grid, home and automation networks, and smart/connected cars. These devices are constrained in terms of performance and resources and hence cannot employ the same heavyweight security measures used in general-purpose computers. ConFirm is a low-cost, HPC-based technique to detect malicious modifications in the firmware of embedded control systems. The ConFirm study evaluated the detection capability and performance overhead on various real-world firmware running on ARM and PowerPC embedded processors. ConFirm was the first work to introduce HPCs to secure the firmware both as a design-for-security concept and as an add-on feature. It was also the first evaluation of an HPC-based security scheme on real-world firmware images used in embedded devices of critical infrastructure.

Before ConFirm, numerous mechanisms were proposed to detect malicious firmware. These mechanisms required either extra hardware components (e.g., Trusted Platform Module) or complex verification methodologies, impacting resource-constrained embedded devices (e.g. computation resources, communication bandwidth, power consumption, and memory use). ConFirm overcame these limitations by observing that a program is a sequence of various types of instructions which during execution can be monitored by low-level hardware events. The behavior of the firmware running on resource-constrained embedded systems can be uniquely characterized by using HPCs at run time. Moreover, one can monitor the relationship between the counts of

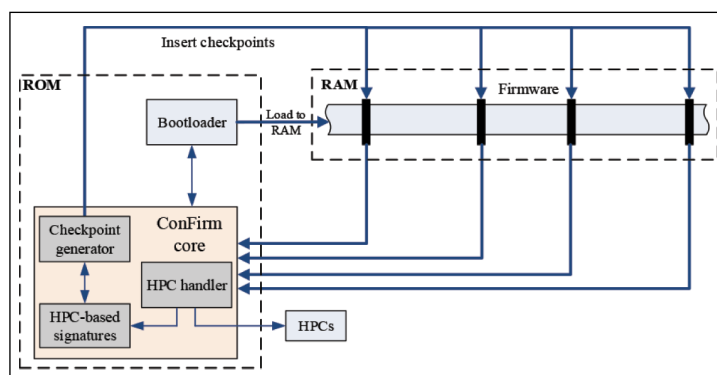
the different events. Finally, ConFirm does not need extra hardware components/ports for deployment to legacy devices supporting HPCs.

ConFirm is a host-based tool that leverages built-in hardware features (HPCs) to detect malicious modifications in embedded firmware. The high-level structure of ConFirm HPC-based monitoring is shown in Figure 2. The work extended a legacy bootloader with ConFirm with three components: 1) an insertion module that places checkpoints in the monitored firmware; 2) an HPC handler that drives and collects HPC statistics; and 3) a database that stores valid HPC-based signatures. These components are stored in write-protected nonvolatile memory to prevent attacks from compromising ConFirm, while still allowing authorized updates. Comparing with software solutions, HPC-based solutions rooted in hardware have lower performance overhead and is tamper-resistant.

#### Follow-up additional academic research

Researchers explored a new direction for hardware-based security in embedded systems by reusing hardware features (HPCs) for detecting malicious firmware and software modifications. This research was extended by researchers along many directions and systematized in [6]. We outline four emerging directions in HPC-based security research.

1) *HPC-based monitoring for security of CPS*: HPCs can be used for real-time monitoring of software running on embedded CPS processors [10]. They



**Figure 2. ConFirm has three components: a module that inserts checkpoints into the monitored firmware, an HPC handler that drives HPCs, and a database that stores HPC-based signatures [9].**

- can apply to detect anomalies in power grid CPS [12], [19], [22].
- 2) *Using HPCs to detect different types of attacks:* Repurposing HPCs for detecting malicious firmware has inspired researchers to explore the feasibility of using HPCs for detecting attacks such as return-oriented programming, ransomware, and side-channels [4], [14]. HPCs can measure mispredicted return events to detect ROP attacks at run time. Profiling HPCs can detect microarchitectural side-channel attacks [14]. HPCs can detect ransomware [4].
  - 3) *Feasibility of other built-in hardware components for security:* Modern computer systems have on-chip sensors including thermal, voltage, and frequency sensors and associated reporting interfaces. Readings from these sensors correlate to the behavior of running programs and can be adapted for security monitoring. Combining thermal profiles of processors with HPCs can detect malicious changes due to software and hardware attacks [23].
  - 4) *HPCs can be abused as a security backdoor:* While HPCs can monitor malicious behavior of a program, they also open an avenue for attackers to collect security-sensitive information. Concerns have been raised that HPCs might be abused as a security backdoor. Research efforts have been published along this line. Spisak [24] introduced a hardware-assisted rootkit on ARM and Intel x86-64 architectures. This rootkit allows an attacker to redirect control flow to malicious code by using HPCs count specific events. Alam et al. [14] present a micro-architectural side-channel attack by analyzing HPC counts when executing encryption algorithms.

## Transition to practice case studies

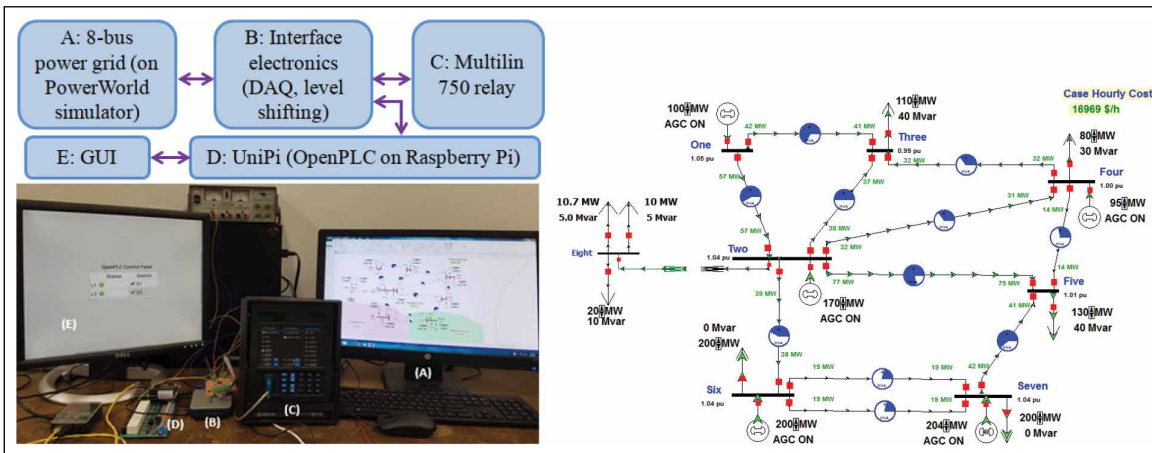
### DARPA RADICS

The DARPA RADICS program developed systems to restore power following a cyber-attack on the power grid [12]. DARPA assembled several teams with over 100 participants to solve this critical problem. SRI led the Threat Intelligence for Grid Recovery (TIGR) with NYU on the team [25]. RADICS kicked-off in August 2016 and ended in February 2021. NYU's HPC-based malware detector was part of a defense in depth.

Red-team and blue-team rehearsals and deployment on a small-scale substation testbed established effectiveness of the malware detector. DARPA used a substation on Plum Island, off Long Island, NY, USA, as a close-to-live testbed. RADICS teams demonstrated technologies, with red-teams injecting malicious code and blue-teams detecting malware and restoring substation devices to create the crank path to restore power.<sup>1</sup> The exercises tested the anomaly detectors in as close to real operational setting as possible since it is infeasible to intentionally bring down all/part of a real power grid or inject malware into operating controller devices for testing.

- 1) *HIL power grid testbed:* We implemented some of the RADICS red-team attacks on an Open-PLC controller in a hardware-in-the-loop (HIL) testbed in Figure 3. This HIL testbed emulates an 8-bus power grid in a PowerWorld simulator that interfaces with a physical GE Multilin 750 relay. The relay is controlled through UniPi, a Raspberry Pi single board computer with an additional board providing input-output (I/O) interfaces for the programmable logic controller (PLC) including analog and digital I/O. Multilin 750 has analog I/O interfaces for monitoring RS232, RS485/RS422, and ethernet ports. A graphical user interface (GUI) is implemented on a second Raspberry Pi using the pvbrowse process visualization browser software. Via this GUI, the operator opens/closes the relay. The operator terminal and the UniPi are connected via ethernet and communicate using Modbus protocol. The command signal of the OpenPLC passes through a data acquisition board to change the status of the relay in the PowerWorld simulator. The relay protects and controls feeders on solidly grounded, high-impedance grounded, or resonant grounded systems.
- 2) *A stealthy attack on power grid:* The interplay of cyber and physical components in the power grid CPS allows an attack to exploit cyber vulnerabilities to impact the physical power grid processes [22]. PLCs are common in all CPS including the power grid. While networking these components benefit maintenance and upgrade, networking admits remote attacks on physical processes

<sup>1</sup><https://www.businessinsider.com/darpa-runs-mock-cyber-attacks-on-small-government-owned-island-2019-5>



**Figure 3. HIL power grid testbed. (a) Power grid simulator for an 8-bus power grid (shown on the right side). (b) Data acquisition board with level shifting. (c) GE multilin 750 relay. (d) UniPi SCADA PLC emulator; Raspberry Pi that controls the OpenPLC. (e) GUI terminal to monitor relay status using operator interface running on another Raspberry Pi. UniPi communicates with the GUI via Modbus.**

[26]. An attacker can exploit vulnerabilities in the computation, communication, and network nodes. These include vulnerabilities in the supply chain where the malicious code is “designed” into the controllers, remaining latent until triggered. An attacker may: 1) modify the controller logic to impact stability, performance, or safety of the physical process; 2) spoof information from sensors, spoof outputs of actuators, or communication between operator control stations and the controllers; and 3) exfiltrate sensitive data, add backdoors, or give access to unauthorized users.

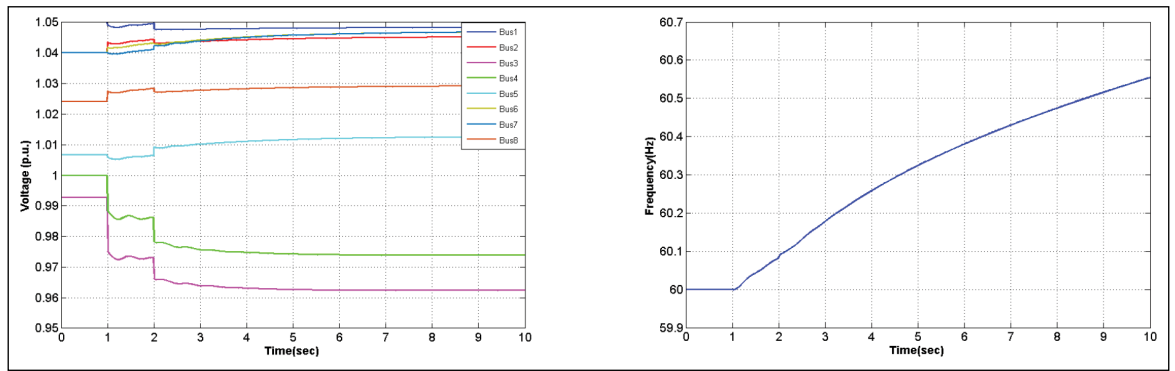
In one of the attacks on the power grid, a stealthy rootkit enables malware in embedded controllers. The attack seeks to undermine stability, efficiency, and safety of the grid. The malware injects a dynamically loaded library into the controller software to overwrite commands from an operator to a relay in the grid. Using open-source kernel- and user-space rootkit libraries, the attacker can create a daemon to replace the OpenPLC controller process with a process that has a malicious library hooked to it. This malicious library hooks into the process to override crucial I/O routines in the controller code. Furthermore, it sends modified operator commands to the physical I/O and incorrect status messages to the operator control station. We validated the attack in an HIL testbed, as shown in Figure 3.

The adversary can modify a process that runs the control logic on a PLC. This control logic is specified via a structured text program or a graphical ladder logic and is loaded onto the PLC as an executable. In one embodiment, the adversary gains unauthorized access to the PLC using a vulnerability in the network protocol used to program the PLC and implants the malware. While the attack spoofs actuator commands, it can spoof sensor readings, modify variables and logic controller altering control behavior, and exfiltrate sensitive data [27].

- 3) *Effects of attack on the power grid:* An informed attacker can calculate the betweenness index and reactance of the lines of the grid to determine that, for example, the transmission line connecting nodes 2 and 3 is critical (Figure 3). Excluding this line from the network and recomputing the betweenness can yield the next critical line, for example, between nodes 4 and 5. These two lines are good attack targets.

During the attack of opening–closing of relays on critical lines, power flow through the line connecting buses 1 and 3 reaches 84% of the tolerable power flow and power flow through the line connecting buses 2 and 5 reaches 87% of the tolerable power flow. Congestion in the lines increases the maintenance cost and nodal price of the buses corresponding to the congested





**Figure 4. (a) Voltage on buses when relays on  $L_{23}$ ,  $L_{45}$  open at  $t = 1$  and, 2 s. (b) Frequency when relays on  $L_{23}$ ,  $L_{45}$  open at  $t = 1$  and, 2 s.**

lines. Figure 4a shows the impacts of attacks on transient response of the bus voltages. The voltage level at different buses in normal condition should be between 1.05–0.95 per unit. Although a steep change is seen in the voltage level of the buses when the transmission lines disconnect from the network, all bus voltages remain in range. Figure 4b shows the frequency response. Transient stability simulations do not show regulation of frequency back to nominal (60 Hz). Besides attacking relays on transmission lines, the attacker can target load relays to degrade performance and stability.

- 4) *HPC-based detection of stealthy malware*: Preventing and detecting attacks requires a multilayered approach [28] from a myriad of vantage points: network, on-device, and process-aware monitors [29]. An ensemble of anomaly detectors can detect the malware. While some detect anomalies relative to a baseline, others do not require measurements from a known good device. HPCs detect changes in run-time characteristics of code execution. Listings of mapped memory regions of a process detect unexpected dynamically loaded libraries or changes in libraries. Reading the system call table detects changes in memory addresses in the system call table or in the system call handler functions. Techniques to detect anomalies in process listings and file system entries uncover processes and file system entries hidden by a rootkit. We will highlight the HPC-based detection next.
- 5) The quad-core ARM Cortex on the Raspberry Pi allows simultaneous reading of six HPCs. We

use numbers of instructions, branches, stores, cycles, L1 instruction cache misses, and L2 data cache misses in this study. We read these HPCs at a sampling rate of 1 kHz. The HPCs are collected per-thread for each of the three threads in the OpenPLC controller process using a measurer process. The HPC measurer uses the PAPI library, connects to the target process, and reads the time series of HPCs corresponding to each of the threads. We implemented the HPC measurer in C++, compile it on a separate computer into a statically linked executable, and deploy it to OpenPLC.

- 6) A baseline data set of HPC readings over a time interval of 120 s was collected from a good and trusted device. Half of this data set was used to train a one-class support vector machine (SVM) classifier [10]. A sliding time window 0.25 s long was considered (with a time shift of 0.01 s between successive time windows). One of the three threads in the OpenPLC controller was quiescent. The mean and standard deviations, over the time window, of the measured HPCs for the two nonquiescent threads were used to construct the feature vector for the one-class SVM classification of normal/outlier. A 120-s-long data set was collected after deploying the malware.

The time series of HPC measurements from the good device and the malware-infested device are in Figure 5. While these time series look similar from a macroscopic view except for intermittent nondeterministic spikes, the one-class SVM trained on baseline data accurately distinguishes between

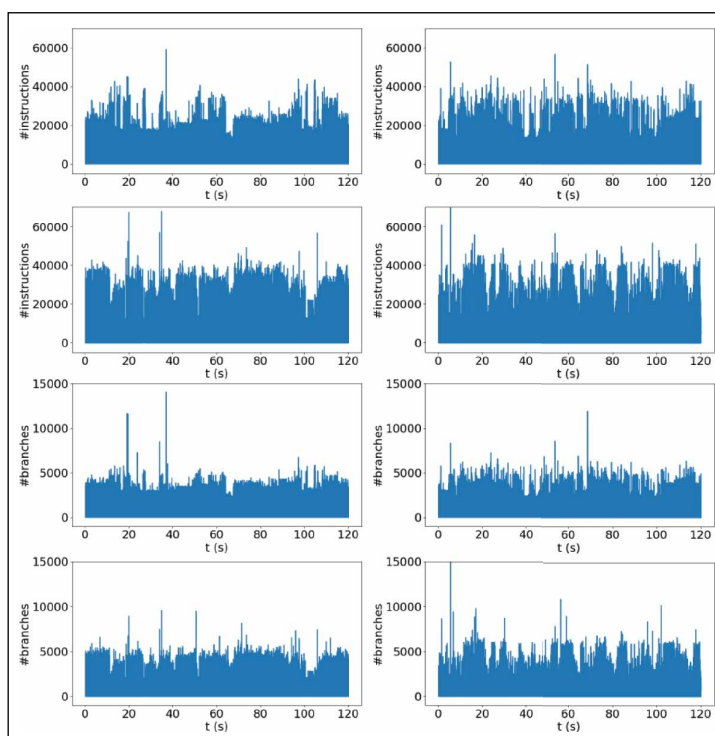
data from the good and malware-infected devices. The one-class SVM is trained only on data from the baseline device. The SVM was tested on the second half of the baseline data set and the data set collected under malware. A sliding window of 40 normal/outlier detection with majority voting is used to output normal/outlier labels. The classification accuracy for baseline data is 90.06% (i.e., 10% false positives where normal data is marked anomalous) and classification accuracy for data from the malware-infested device was 93.94% ( $\approx 6.0\%$  of false negatives where anomalous data is marked normal).

## Intel TDT

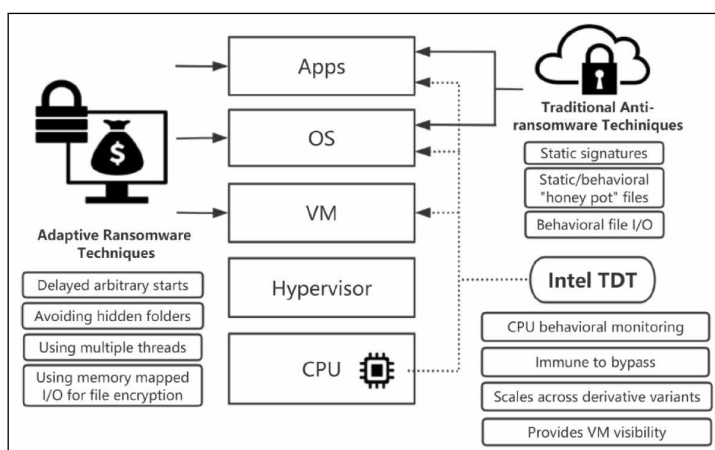
After a decade of academic research and proof-of-concept demonstrations, Intel as a major processor vendor has taken an important step to unlock the capabilities of HPC-based malware detection. Intel developed TDT as part of its Hardware Shield suite. Intel TDT leverages HPCs and hardware acceleration of machine learning on an integrated GPU to collect, profile, and detect malicious activities [13].

HPC-based TDT can detect ransomware [7]. Ransomware has emerged as the most prominent threat in cybersecurity, causing millions of dollars in losses due to ransom payments annually. A typical ransomware, such as WannaCry, encrypts the files on an infected computer using a private key encryption algorithm. WannaCry uses AES-128 in cipher block chaining mode with a randomly generated key. These keys are encrypted with an RSA public key that is specific to an attack [30]. Software-based ransomware detectors focus on classes of encryption algorithms, require modifications to the operating systems and have high detection latency. As presented in Figure 6, software-based detection can be bypassed by adaptive ransomware using delayed arbitrary starts and memory mapped I/O-based file encryption [7].

HPC-based TDT is agnostic to encryption algorithms and does not have a performance constraint, as it is implemented in hardware. Ransomware operations involve accessing  $\rightarrow$  opening  $\rightarrow$  encrypting  $\rightarrow$  closing files one after another. Encryption algorithms can be detected by the special patterns in cache-references, cache-misses, branches, and branch-misses that can be monitored using HPCs. Figure 7 shows two HPC-related events for the WannaCry ransomware [30].



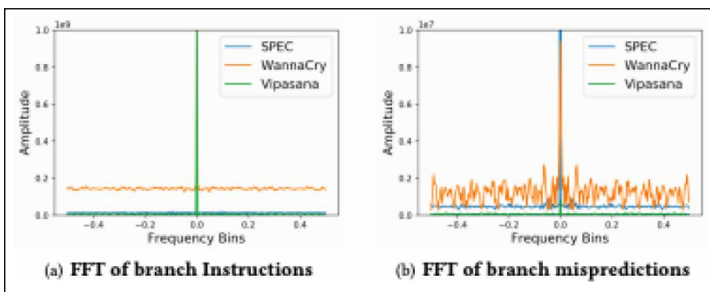
**Figure 5. HPC measurements for instructions and branches for the two nonquiescent threads in the OpenPLC under normal (left column) and malware-infested (right column) conditions.**



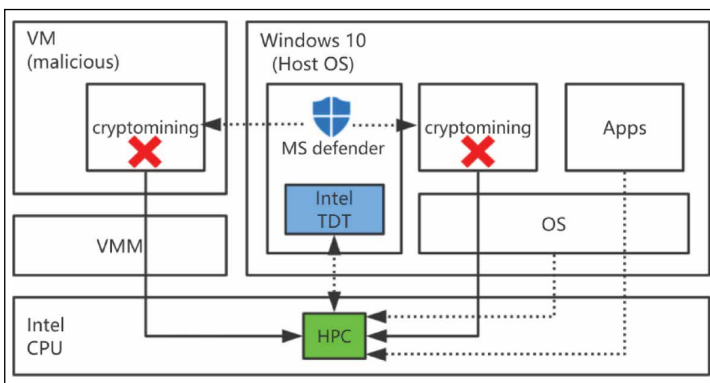
**Figure 6. Software-based ransomware detection is slow and can be bypassed by adaptive ransomware. HPC-based detectors are of low latency, cannot be tampered, and scale to variants of ransomware.**

## Microsoft Defender for Endpoints

Microsoft Defender for Endpoints uses Intel TDT to detect unauthorized crypto mining [8]. Such cryptojacking entails maliciously co-opting



**Figure 7. Branch instruction and misprediction HPCs for SPEC benchmarks and Wannacry ransomware [30].**



**Figure 8. Microsoft Defender for Endpoint uses Intel TDT to detect unauthorized crypto mining.**

computing resources of community to mine crypto currencies [8]. This is a formidable threat. Malicious cryptominers can be deployed as native applications that are transferred to other systems with the victim's credentials, or they can come in the form of browser-based malware when the victim visits the malicious website. Software-based detection techniques flag malicious crypto mining by checking high CPU usage and overheating and this can be evaded [5].

Crypto mining repeatedly uses mathematical operations, and these activity patterns can be detected by monitoring HPCs. Microsoft uses Intel TDT for HPC-based monitoring in their commercial malware defense tool. By analyzing the values of the selected HPCs, the system determines whether someone is mining with or without the owner's consent. As illustrated in Figure 8, when malicious crypto mining launches in the host operating system or in a guest virtual machine (VM), the CPU monitors the HPCs when the application or the VM is executing. Microsoft Defender uses Intel TDT to collect HPC

data and perform ML analysis. If malicious activity is flagged, Microsoft Defender remediates it. Since TDT offloads ML analysis to the integrated GPU, Microsoft Defender can continuously monitor with low performance overhead.

**IN THIS ARTICLE**, we described the important milestones that enabled the use of HPC-based security solutions for the protection of the power grid (by the U.S. Government) and for endpoint protection (by Microsoft using Intel TDT). HPCs were first proposed by NYU researchers in 2011, and a wealth of follow-up academic work showcased the potential of HPCs to protect against various threats, including firmware modifications, kernel rootkits, malware, and ransomware. After 10 years, HPC-based security solutions have successfully transitioned to practice. There exist, however, challenges that need to be addressed in the future, including: 1) efficient utilization of HPCs without requiring vendor instrumentation; 2) integrate application-aware traces in coordination with HPC values toward minimization of false positives in detection algorithms; and 3) adjust and develop HPC detection algorithm able to capture complex file infectors such as polymorphic viruses.

## References

- [1] C. Malone, M. Zahran, and R. Karri, "Are hardware performance counters a cost effective way for integrity checking of programs?" in *Proc. ACM Workshop Scalable Trusted Comput.*, 2011, pp. 71–76.
- [2] X. Wang and R. Karri, "NumChecker: Detecting kernel control-flow modifying rootkits by using hardware performance counters," in *Proc. 50th Annu. Design Autom. Conf. (DAC)*, 2013, pp. 1–7.
- [3] A. Tang, S. Sethumadhavan, and S. Stolfo, "Unsupervised anomaly-based malware detection using hardware features," in *Research in Attacks, Intrusions and Defenses* (Lecture Notes in Computer Science), vol. 8688. Cham, Switzerland: Springer, 2014, pp. 109–129.
- [4] M. Alam et al., "RAPPER: Ransomware prevention via performance counters," 2020, *arXiv:2004.01712*.
- [5] G. Mani et al., "DeCrypto pro: Deep learning based cryptomining malware detection using performance counters," in *Proc. IEEE Int. Conf. Auton. Comput. Self-Organizing Syst. (ACSOS)*, Aug. 2020, pp. 109–118.
- [6] S. Das et al., "SoK: The challenges, pitfalls, and perils of using hardware performance counters for security,"



- in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2019, pp. 20–38.
- [7] *Detect Ransomware and Other Advanced Threats With Intel Threat Detection Technology*. Accessed: Dec. 1, 2021. [Online]. Available: <https://www.intel.com/content/www/us/en/architecture-and-technology/threat-detection-technology-brief.html>
- [8] Intel. *Intel Collaborates With Microsoft Against Cryptojacking*. Accessed: Dec. 1, 2021. [Online]. Available: <https://www.intel.com/content/www/us/en/newsroom/news/intel-microsoft-scale-threat-detection-cryptojacking.html>
- [9] X. Wang et al., “ConFirm: Detecting firmware modifications in embedded systems using hardware performance counters,” in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Design (ICCAD)*, Nov. 2015, pp. 544–551.
- [10] P. Krishnamurthy, R. Karri, and F. Khorrami, “Anomaly detection in real-time multi-threaded processes using hardware performance counters,” *IEEE Trans. Inf. Forensics Security*, vol. 15, pp. 666–680, 2020.
- [11] K. Basu et al., “A theoretical study of hardware performance counters-based malware detection,” *IEEE Trans. Inf. Forensics Security*, vol. 15, pp. 512–525, Jun. 2019.
- [12] *Rapid Attack Detection, Isolation and Characterization Systems (RADICS)*. Accessed: Dec. 1, 2021. [Online]. Available: <http://www.darpa.mil/>
- [13] Intel. *Intel® Threat Detection Technology (Intel® TDT) Product Brief*. Accessed: Dec. 1, 2021. [Online]. Available: <https://www.intel.com/content/dam/www/public/us/en/documents/product-briefs/tdt-product-brief.pdf>
- [14] M. Alam et al., “Performance counters to rescue: A machine learning based safeguard against micro-architectural side-channel-attacks,” *IACR Cryptol. ePrint Arch.*, vol. 2017, p. 564, Jul. 2017.
- [15] F. Brasser et al., “TyTAN: Tiny trust anchor for tiny devices,” in *Proc. 52nd Annu. Design Autom. Conf.*, Jun. 2015, pp. 1–6.
- [16] T. Abera et al., “C-FLAT: Control-flow attestation for embedded systems software,” in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, Oct. 2016, pp. 743–754.
- [17] D. Lee et al., “Keystone: An open framework for architecting trusted execution environments,” in *Proc. 15th Eur. Conf. Comput. Syst.*, Apr. 2020, pp. 1–16.
- [18] Z. Pan et al., “Hardware-assisted malware detection using machine learning,” in *Proc. Design Autom. Test Eur. (DATE)*, 2021, pp. 1775–1780.
- [19] P. Krishnamurthy et al., “Stealthy rootkits in smart grid controllers,” in *Proc. IEEE 37th Int. Conf. Comput. Design (ICCD)*, Nov. 2019, pp. 20–28.
- [20] X. Wang and R. Karri, “Reusing hardware performance counters to detect and identify kernel control-flow modifying rootkits,” *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 35, no. 3, pp. 485–498, Aug. 2016.
- [21] X. Wang et al., “Malicious firmware detection with hardware performance counters,” *IEEE Trans. Multi-Scale Comput. Syst.*, vol. 2, no. 3, pp. 160–173, Jul. 2016.
- [22] I. Zografopoulos et al., “Cyberphysical energy systems security: Threat modeling, risk assessment, resources, metrics, and case studies,” *IEEE Access*, vol. 9, pp. 29775–29818, 2021.
- [23] N. K. Patel et al., “Towards a new thermal monitoring based framework for embedded CPS device security,” *IEEE Trans. Dependable Secure Comput.*, vol. 19, no. 1, pp. 524–536, Jan./Feb. 2020.
- [24] M. Spisak, “Hardware-assisted rootkits: Abusing performance counters on the ARM and x86 architectures,” in *Proc. 10th USENIX Workshop Offensive Technol. (WOOT)*, 2016, pp. 79–90.
- [25] *SRI International to Lead Program to Develop Technology for Restoring Power to a Grid Facing a Cyberattack*. Accessed: Dec. 1, 2021. [Online]. Available: <https://www.prnewswire.com/news-releases/sri-international-to-lead-program-to-develop-technology-for-restoring-power-to-a-grid-facing-a-cyberattack-300365086.html>
- [26] C. Xenofontos et al., “Consumer, commercial and industrial IoT (in)security: Attack taxonomy and case studies,” *IEEE Internet Things J.*, vol. 9, no. 1, pp. 199–221, Jan. 2022.
- [27] P. Krishnamurthy et al., “Process-aware covert channels using physical instrumentation in cyber-physical systems,” *IEEE Trans. Inf. Forensics Security*, vol. 13, no. 11, pp. 2761–2771, Nov. 2018.
- [28] S. McLaughlin et al., “The cybersecurity landscape in industrial control systems,” *Proc. IEEE*, vol. 104, no. 5, 2016, Art. no. 10391057.
- [29] F. Khorrami, P. Krishnamurthy, and R. Karri, “Cybersecurity for control systems: A process-aware perspective,” *IEEE Design Test. Comput.*, vol. 33, no. 5, pp. 75–83, Oct. 2016.
- [30] M. Alam et al., “RAPPER: Ransomware prevention via performance counters,” 2018, *arXiv:1802.03909*.

**Charalambos Konstantinou** is an Assistant Professor at the Computer, Electrical and Mathematical Science and Engineering Division, King Abdullah University of Science and Technology (KAUST), Thuwal, Saudi Arabia. He is the Principal Investigator of the Secure Next Generation Resilient Systems Laboratory ([sentry.kaust.edu.sa](http://sentry.kaust.edu.sa)). His research interests include cyber-physical and embedded systems security and resilience. Konstantinou has a PhD in electrical engineering from New York University, New York, NY, USA. He is a Senior Member of IEEE, a member of ACM, and an ACM Distinguished Speaker.

**Xueyang Wang** is a Security Architect/Researcher at Alibaba Group, Shanghai, China. His research interests include secure architectures, hardware support for software security, and hardware security. Wang has a PhD in electrical engineering from the Tandon School of Engineering, New York University, New York, NY, USA.

**Prashanth Krishnamurthy** is a Research Scientist and an Adjunct Faculty with the Department of Electrical and Computer Engineering, New York University (NYU), New York, NY, USA. His research interests include autonomous and cyber-physical systems and machine learning. Krishnamurthy has a PhD in electrical engineering from Polytechnic University (now NYU). He is a member of IEEE.

**Farshad Khorrami** is a Professor of electrical and computer engineering at New York University, New York, NY, USA. His research interests include control systems, robotics, cyber-physical systems security, autonomous systems, and machine learning. Khorrami has a PhD in electrical engineering from The Ohio State University, Columbus, OH, USA. He is a Senior Member of IEEE.

**Michail (Mihalis) Maniatakos** is an Associate Professor of electrical and computer engineering at New York University Abu Dhabi, Abu Dhabi, United Arab Emirates. His research interests include privacy-preserving computation and industrial control systems security. Maniatakos has a PhD in electrical engineering from Yale University, New Haven, CT, USA. He is a Senior Member of IEEE.

**Ramesh Karri** is a Professor with the Electrical and Computer Engineering Department, New York University, Brooklyn, NY, USA. His research interests include trustworthy hardware (integrated circuits to processor architectures), very large scale integration (VLSI) design and test, and interaction between security and reliability. Karri has a PhD in computer science and engineering from the University of California at San Diego, La Jolla, CA, USA. He is a Fellow of IEEE.

■ Direct questions and comments about this article to Charalambos Konstantinou, Computer, Electrical and Mathematical Science and Engineering (CEMSE) Division, King Abdullah University of Science and Technology (KAUST), Thuwal 23955-6900, Saudi Arabia; [charalambos.konstantinou@kaust.edu.sa](mailto:charalambos.konstantinou@kaust.edu.sa).