RISC-V Program Execution Monitoring & Control (PEMon)

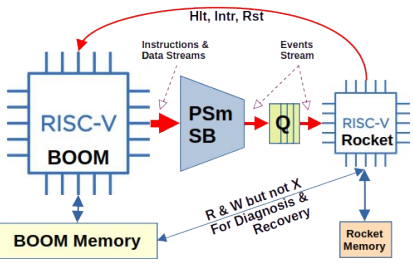
J. S. Mertoguno, Georgia Institute of Technology

Monitoring at Instruction Level Granularity enable real-time detection and prevention of exploits as they progress, often before they can achieve foothold and compromise the system. Without hardware support software will have to execute the software under analysis (SUA) in either emulator (with or without dynamic binary instrumentation) or debugger, which is not practical for deployed/production software/applications. Software-based monitoring generally operates at higher level granularity, and often loses the opportunity to stop exploits on their track. Memory forensic is often required for analyzing and discovering the root cause and entry point of exploits, postmortem.

General purpose real-time (on the fly) SUA's instruction level monitoring can be constructed by having a monitoring processor core to watch over another (the monitored) processor core, with a dedicated circuit forwarding the monitored processor's instruction streams into a buffer for the monitoring processor to evaluate. For each monitored processor instruction, monitoring processor needs an order of magnitude more instructions for reading/checking validity, masking, comparing with a value (may have to load value to register) and based on the comparison result, perform some action. Hence the monitoring processors need to be an order of magnitude faster or more powerful than the monitored processor to keep pace. This is obviously not an ideal solution.

Real-time instruction level monitoring can be practical with hardware supports. Cognizant engine [4] presents a powerful instruction level program execution monitoring, capable of recognizing any sequence of instructions (& their address) patterns and data (& their address) access patterns. Given control over the monitored processor resources, cognizant engine, when necessary, can steer program beyond software and malware intended path, and can be used to prevent or stop and recover from exploit in progress.

Application specific monitoring accelerator, such as Cognizant Engine[4], raised the programming abstraction and uses programmable higher abstraction (co-) processor to perform initial (first level) monitoring to keep pace with the monitored processor, and generate stream of events of interest at much lower frequency for further processing. A relatively weaker processor (or micro-controller) than the monitored processor can be used for the later stage where less frequent but potentially more complex processing is often required. Alternatively, a custom processing logic, the programmable monitor, being used to perform said later stage processing.



GaTech proposes to develop PEMon, in the spirit of Cognizant engine[4], targeted toward securing CPS application. PEMon will be capable of monitoring several data and instruction patterns simultaneously. We propose to instantiate two different RISC-V cores, the higher performance, superscalar BOOM architecture [6] as the monitored core (slave) and the simple single issue Rocket architecture [5] as the monitoring core (master), with programmable state machine performing the initial filtering and event detection, enabling Rocket to monitor and control the more powerful BOOM. GaTech will designs PEMon to showcase a new capability for program execution monitoring at instruction level granularity, without impacting the performance of the monitored processor. PEMon opens a new opportunity for detecting and mitigating low-level events and attacks that are either too expensive and/or too faint for software to monitor and detect.

Hardware assisted cyber-security monitoring and protection, such as PEMon, is practical for embedded controllers applications. Models and deterministic behavior expectations are often existed in CPS. There is also strict latency requirements preventing the use of heavy software protection techniques in this area. Expected behaviors can be used as reference model for enforcing by monitoring sequence of state invariants. Beyond the cyber physical systems (CPS) specific cyber protections above, general cyber security protections, such as control flow integrity (CPI) enforcement can also be efficiently and practically implemented using PEMon, within embedded controller operational environment. PEMon provides defense against control flow hijacking and firmware corruption and malicious reprogramming.

References:

1. C. Yagemann, M. A. Noureddine, W. U. Hassan, S. Chung, A. Bates, and W. Lee, “Validating the integrity of audit logs against execution repartitioning attacks,” in Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security, ser. CCS ’21
2. C. Yagemann, M. Pruett, S. P. Chung, K. Bittick, B. Saltaformaggio, and W. Lee, “ARCUS: Symbolic root cause analysis of exploits in production systems,” in 30th USENIX Security Symposium (USENIX Security 21), USENIX Association, Aug. 2021, pp. 1989–2006
3. C. Yagemann, S. P. Chung, B. Saltaformaggio, and W. Lee, “Automated bug hunting with data-driven symbolic root cause analysis,” ser. CCS ’21, Virtual Event, Republic of Korea: Association for Computing Machinery, 2021
4. Sukarno Mertoguno, “Cognizant engines: Systems and methods for enabling program observability and controlability at instruction level granularity”, 2008. https://www.freepatentsonline.com/y2010/0107252.html.
5. Rocket Chip Generator https://bar.eecs.berkeley.edu/projects/rocket_chip.html
6. Welcome to RISCV-BOOM’s documentation https://docs.boom-core.org/en/latest/
7. Meng Xu, Kangjie Lu, Taesoo Kim, and Wenke Lee, “Bunshin: Compositing Security Mechanisms through Diversification”, 2017, arXiv preprint arXiv:1705.09165
8. J.G. Rivera, A.A. Danylyszyn, C.B. Weinstock, L.R. Sha, M.J. Gagliardi, “An Architectural Description of the Simplex Architecture”, CMU/SEI-96-TR-006, https://resources.sei.cmu.edu/library/asset-view.cfm?assetid=12521