System Infrastructure for SMM-based Runtime Integrity Measurement

Karen L. Karavanic, Portland State University
(karavan@pdx.edu)

www.cs.pdx.edu/~karavan/research/SMM.html

With Brian Delgado (also Intel), John Fastabend, Cody Shepherd (also Canonical), Tejaswini Vibhute (now Intel), Dylan Abraham, Alex Freed, Payal Joshi, Stephen Rivera, and high school interns Sonja Johansen and Ozzy Sanchez-Aldana

Is it feasible to use SMM-based Runtime Integrity Measurement in a production environment?

One 2018 estimate puts the cost of malicious cyber activity in the US economy between $57B - $109B in 2016 (White House, 2018). Detecting unexpected changes in a system's runtime environment is critical to resilience.

A repurposing of System Management Mode (SMM) for runtime security inspections has been proposed, due to SMM's high privilege and protected memory. However, key challenges prevented SMM's adoption for this purpose in production-level environments.

We introduce a Runtime Integrity Measurement framework, EPA-RIMM, for both native Linux and Xen platforms, that includes several novel features to solve these challenges. Our Linux and Xen prototype results show that EPA-RIMM meets performance goals while continuously monitoring code and data for signs of attack, and that it is effective at detecting a number of recent exploits.

Key Challenges and their Solution with EPA-RIMM

Challenge	Our Solution
Flexible and extensible measurement framework • Avoid storing static OS layout information in SMM • Allow measurements to vary over time without firmware updates	• Extensible Check specification • Tunable knobs for monitoring frequency and duration
Avoid negative system impacts (latency)	• Bin Size determines interval spent in SMM • Checks are decomposed into Tasks to fit maximum SMM interval
Limit workload degradation during continuous monitoring (throughput)	• Developed a multicore version of SMM: Coreboot EPA-RIMM
Measurement agent resilience	• Use Intel SMI Transfer Monitor (STM) to restrict EPA-RIMM capabilities
Support for CPU-based virtualization (Intel VT) • Semantic gap between host software and SMM	• Used Intel STM to locate correct context information

SMM RIMM	SMI Duration	Frequency
HyperCheck	40ms	1 per second
HyperSentry	35ms	1 per 8 or 16 second
SPECTRE	5 to 32ms	16 per second to 1 per 5 seconds
Upper Bound on SMM cost	1.5ms	Not specified
EPA-RIMM (with STM) Minnowboard	0.28ms+	Dynamic
EPA-RIMM (no STM) Minnowboard	0.26ms+	Dynamic
Intel BIOS BITS Guideline	0.15ms	Not specified

This material is based upon work supported by the National Science Foundation under Grant No. 1528185. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

Portland State UNIVERSITY