Project Proposal:
Container Integrity Measurement and Monitoring using eBPF

Avery Blanchard, agb2178

Introduction

Lightweight container-based virtualization allowing for agile scaling is ubiquitous in modern computing. The ability to ensure trust and integrity in a container remains a pertinent security issue. The Linux kernel's mechanism for run-time integrity monitoring does not support Linux containers. This leaves a substantial gap in container observability, integrity, and trust. In this project, I propose to address this problem by researching and implementing a run-time solution for continuous integrity monitoring of Linux containers without changes to the kernel. My work will include: (1) identifying and implementing the appropriate mechanisms for bootstrapping container integrity measurements, (2) designing a system to provide monitoring on the host operating system, and (3) demonstrating the usage and security benefits of container integrity measurements.

Problem

The Linux Integrity Measurement Architecture (IMA) is an open-source Trusted Computing component that maintains a run-time integrity list anchored in a hardware Trusted Platform Module (TPM). Measurements from IMA are used to attest to the integrity of a system. IMA is implemented in the Linux kernel as a security module that hooks into the mmap() operation. Whenever a file is mapped with an executable mode, IMA creates a hash of the file. IMA sends each of these hashes to a Platform Configuration Register (PCR) on the TPM. Through this process, IMA creates a record of immutable file states that can be used for remote attestation, trusted boot, and run-time integrity monitoring. Because IMA does not distinguish between container and host, it cannot be used for integrity checking in a container. Work is being done to implement IMA namespacing through a number of kernel patches, which would extend the support of IMA to Linux containers.[1] Other implementations of container IMA modify the operating system to add a namesapce parsing mechanism on top of IMA to differentiate between the container and the host.[2]. I aim to implement container integrity measurements by using an eBPF KProbe to measure the state of executable files in a container and monitor these measurements to ensure integrity.

Methods

I will attempt to use of eBPF to implement container observability and monitoring functions with minimal overhead and no changes to the Linux kernel. eBPF is a technology that safely runs sandboxed programs in a privileged context. Researchers have used eBPF for implementing a virtual tap on inter-VM traffic[3], in-Kernel Storage functions [4] and non-intrusive performance monitoring [2]. Here, eBPF will be used to trace system calls related to container executable files through the use of a KProbe. A KProbe is a debugging mechanism in the Linux kernel that can be attached to eBPF programs in kernel and user space. Each time an executable file is mapped, a cryptographic hash of the file is created. The TPM will then sign the list of recorded container hashes and make these measurements available as a TPM quotes. A loadable kernel module will be used to instantiate the eBPF probe, create a cryptographic hash of container executable files, establish the hardware root of trust, and monitor container state.

References

[1] S. Berger, "[patch v11 00/27] ima: Namespace ima with audit support in ima-ns."

[2] W. Luo, Q. Shen, Y. Xia, and Z. Wu, "Container-IMA: A privacy-preserving integrity measurement architecture for containers," in *22nd International Symposium on Research in Attacks, Intrusions and Defenses (RAID 2019)*, (Chaoyang District, Beijing), pp. 487–500, USENIX Association, Sept. 2019.

[3] B. Stelte, R. Koch, and M. Ullmann, "Towards integrity measurement in virtualized environments — a hypervisor based sensory integrity measurement architecture (sima)," in *2010 IEEE International Conference on Technologies for Homeland Security (HST)*, pp. 106–112, 2010.

[4] Y. Zhong, H. Li, Y. J. Wu, I. Zarkadas, J. Tao, E. Mesterhazy, M. Makris, J. Yang, A. Tai, R. Stutsman, and A. Cidon, "XRP: In-Kernel storage functions with eBPF," in *16th USENIX Symposium on Operating Systems Design and Implementation (OSDI 22)*, (Carlsbad, CA), pp. 375–393, USENIX Association, July 2022.

[5] L. Jacquin, A. L. Shaw, and C. Dalton, "Towards trusted software-defined networks using a hardware-based integrity measurement architecture," in *Proceedings of the 2015 1st IEEE Conference on Network Softwarization (NetSoft)*, pp. 1–6, 2015.

[6] J. Hong, S. Jeong, J.-H. Yoo, and J. W.-K. Hong, "Design and implementation of ebpf-based virtual tap for inter-vm traffic monitoring," in *2018 14th International Conference on Network and Service Management (CNSM)*, pp. 402–407, 2018.

[7] C. Cassagnes, L. Trestioreanu, C. Joly, and R. State, "The rise of ebpf for non-intrusive performance monitoring," in *NOMS 2020 - 2020 IEEE/IFIP Network Operations and Management Symposium*, pp. 1–7, 2020.