----------------------- REVIEW 1 ---------------------

PAPER: 185

TITLE: HyperMI: A Privilege-level Secure Execution Environment for VM Isolation against Compromised Hypervisor

AUTHORS: Wenqing Liu, Min Zhu, Kun Zhang and Bibo Tu

Overall evaluation: -1 (weak reject)

----------- Overall evaluation -----------

The paper proposes a page-table based isolation mechanism to create a separate secure environment for security tools that runs side-by-side to the hypervisor. The tool can prevent the hypervisor from accessing the guest memory in case of security breach.

The idea of page-table based isolation is not new at all. So is the parallel secure execution environment. In addition, there are some design issues. In particularly, how do you intercept the hypervisor's access to privileged resources, such as the control registers? This is an important detail that can affect the system security. For example, you can use instrumentation but that does not work because the attacker can use ROP for the same purpose. The fact that the paper never mentions ROP is concerning. Overall, the same-privilege isolation is rather fragile. It is hard to exhaust all possible attack vectors in the over-complicated Intel platform. For example, what about MSRs? Intel uses MSRs to control many aspects of the system.

----------------------- REVIEW 2 ---------------------

PAPER: 185

TITLE: HyperMI: A Privilege-level Secure Execution Environment for VM Isolation against Compromised Hypervisor

AUTHORS: Wenqing Liu, Min Zhu, Kun Zhang and Bibo Tu

Overall evaluation: -1 (weak reject)

----------- Overall evaluation -----------

This paper proposes a secure execution environment for VMs against possible attacks from hypervisors (and OS as well). To do this, the authors have designed the HyperMI world (a kind of trust world in a VM environment) and implemented a prototype system in KVM.

I agree that hypervisor and even OS could be compromised and thus we need to devise a security framework for protecting VMs. However, I don't think the proposed approach in this paper is quite effective compared with other related studies. Specifically, SGX may be the most decent architecture that can address this issue in these days.

While the authors claim that "approaches based on hardware is complex and non-feasible", I cannot agree on this. There are many recent works using SGX to protect processes, VMs, containers from attacks (even from compromised OS).

For example, SCONE (OSDI 16) suggests a framework that protecting container environments (with SGX) from attacks. In addition, Graphene-SGX (ATC 17) provides a way of developing SGX applications easily. All those works clearly show that hardware approaches are quite feasible and easy to use.

I believe that the authors should present that what are the advantages of the proposed system compared with those H/W based approaches.

In addition, (some minor thing) there are some other approaches to monitor OS/hypervisor to understand whether it is compromised or not (with third-party H/W). They can effectively detect attacks from hypervisor/OS and require minimum changes of hypervisor/OS (e.g., KI-MON Usenix Security 13). I think that those approaches need to be considered as well.

----------------------- REVIEW 3 ---------------------

PAPER: 185

TITLE: HyperMI: A Privilege-level Secure Execution Environment for VM Isolation against Compromised Hypervisor

AUTHORS: Wenqing Liu, Min Zhu, Kun Zhang and Bibo Tu

Overall evaluation: -1 (weak reject)

----------- Overall evaluation -----------

[Summary]

This paper presents HyperMI, a framework that aims to support a secure isolated execution environment at the same privilege level and in the same space with hypervisor. HperMI is designed to provide isolation protection for virtual machines and conduct event-driven runtime monitoring.

[Strengths]

+ TEE is a hot topic, this paper aims to develop a TEE in hypervisor-level without trusting it. The direction of this work seems interesting to me.

[Weaknesses]

- The main concern that I have for this paper is that the proposed isolation mechanism seems to have a lot flaws. This is very critical since achieving isolation is the foundation for this work.

- The isolation of HyperMI is not clear to me. The paper claims that HyperMI "pre-allocates some space during trusted boot which kernel can’t access directly through MMU, record all entries that map to critical data in HyperMI world in this space". If an attacker compromise the system software stack including hypervisor and OS, he or she can allocate (i.e., remapping) this memory space again since there is no fundamental protection mechanism to protect this memory region.

- It seems that the isolated "secure execution environment" (i.e., HyperMI) is implemented via configuring page table settings. Though the paper presents a few approaches to protect the page tables and related registers such as pre-allocating space, secure switching, etc. However, these approaches are not secure and can be easy bypassed by attackers as mentioned above.

- A minor thing, the writing of this paper needs further improvement. E.g.,

"an secure isolation execution environment" -> "a secure isolation execution environment"