Contribution ID: **67** Type: **not specified**

Touch but don't look: Running the kernel in execute only memory

Monday, September 9, 2019 10:45 AM (45 minutes)

Execute only memory can protect from attacks that involve reading executable code. This feature already exists on some CPUs and is enabled for userspace.

This talk will explain how we are working on creating a virtualized "not-readable"permission bit for guest page tables for x86 and the impact to the kernel. This bit can be used to create execute-only memory for userspace programs as done on other architectures, but newly also kernel text itself. This project has a working POC, but requires extra care being taking in the kernel going forward around certain code patterns in order for the kernel to run in execute only. This will be the main "call to action"of the talk.

The talk will cover three areas:

-Benefits of execute only memory

As was covered in the talk last year by Kristen Accardi, execute only memory can protect code diversification schemes like KASLR, ASLR, and especially fined grained ASLR. This would be a brief summary and will also touch on some attacks that involve reading kernel text

-How we are implementing this across QEMU, KVM, and the guest Linux Kernel.

The solution is sort of novel and interesting it itself, but most of the talk will be about kernel impact of this feature on not the hypervisor implementation. The gist of the solution involves pretending to the guest that the CPU has one less physical address bit than it actually does, so what looks to the guest like a reserved bit looks to the CPU like a physical address bit. Our proposed new KVM APIs can allow userspace VMMs to duplicate memory such that this bit selects from differently permission-ed copies of the same guest physical memory. Intel EPT has the ability to create execute only guest physical memory, so by having the second half of the memory as execute only, we can make a bit that can mark guest virtual memory as execute only.

-Proposed APIs for using execute only memory in userspace and changes and restrictions required to the Linux kernel in order for it to map its own executable code as execute only.

Our POC required making surprisingly few changes to the Linux kernel, however there were impacts especially around features that involve modifying or mapping new executable code. Long term, however, supporting this feature fully would involve the community agreeing that going forward, code patterns that violate execute only memory would not be allowed in the kernel.

I agree to abide by the anti-harassment policy

Yes

Primary author: EDGECOMBE, Rick (Intel)

Presenter: EDGECOMBE, Rick (Intel)

Session Classification: Kernel Summit Track

Track Classification: Kernel Summit talk