Review “SPROBES”

Name: Wenhui Zhang

Paper Name :

Ge, X., Vijayakumar, H., & Jaeger, T. (2014). Sprobes: Enforcing kernel code integrity on the trustzone architecture. *arXiv preprint arXiv:1410.7747*.

Contribution:

To defence kernel rootkits, kernel code integrity is enforced on ARM TrustZone architecture. The following four security guarantees are enforced on normal kernel execution: (1) Execution of user space code from the kernel must never be allowed; (2) W⊕X protection employed by the operating system must always be enabled; (3) The page table base address must always correspond to a legitimate page table; (4) Any modification to the page table entry must be mediated and verified; (5) MMU must be kept enabled to ensure all existing memory protections function properly.

Motivation:

Operating system might be compromised, and thus is vulnerable to iago attacks and rootkits. There exists hardware support that eliminates need to trust the operating system, such as SGX and Trustzone.

Related works:

Iago attacks paper – Checkoway and Shacham [ASPLOS 2013]; SecVisor and NICKLE are VMM-based approaches for protecting kernel code integrity; State-based control flow integrity (SBCFI) monitor; VMM-based monitor like Livewire; some works on coprocessor based ones.

Methodology:

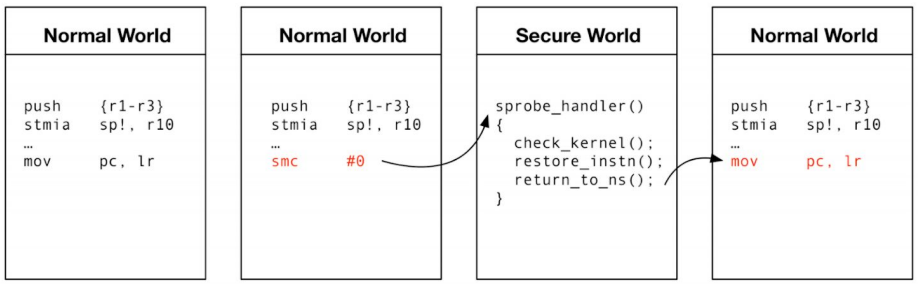

Fig. 3: World switches during SPROBE hit

Results:

SPROBES are inserted for ARM Fast MOdels emulator, and 12 SPROBES is enough to satisfy the 5 constraints. These 12 SPROBES includes: (1) The 6 SPROBES that protect the SCTLR containing the WXN and MMU Enable bit; (2)The 4 SPROBES that protect the TTBR containing the base address of the page table; (3) An SPROBE that protects the TTBCR to enforce usage of only one TTBR (i.e., TTBR0), as required by Linux; (4) The SPROBE that is inserted at the first instruction of the page fault handler.

Take away:

Use secure world to validate normal world.