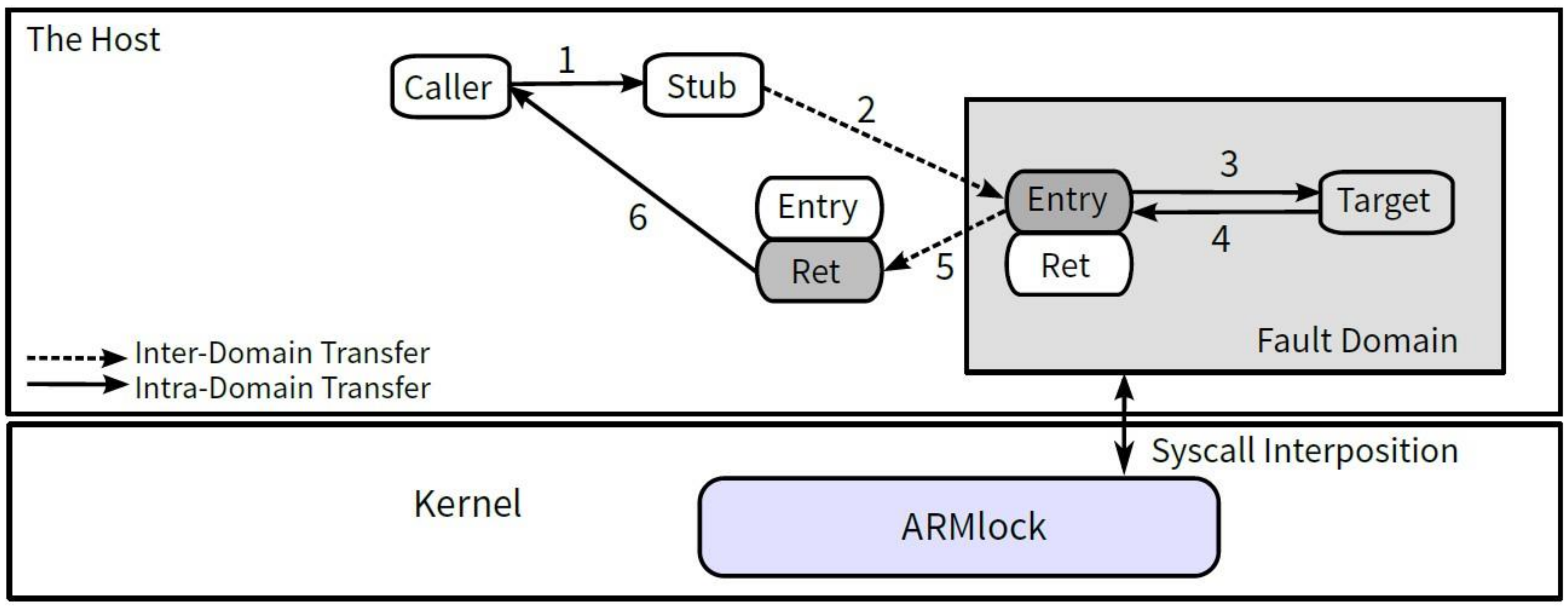
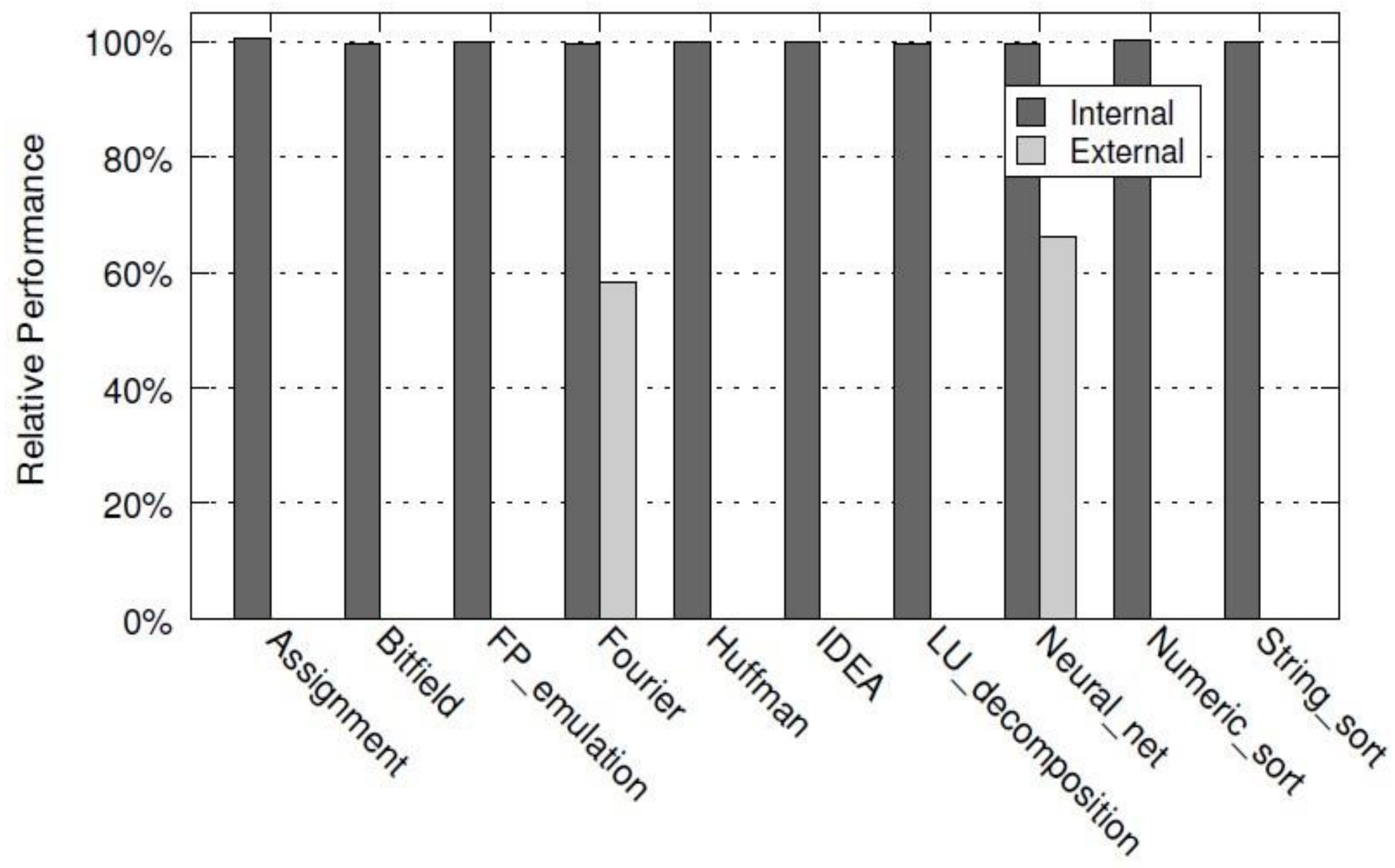
ARMlock

Hardware-assisted Software Fault Isolation for ARM



An Overview of ARMlock

- **Problem:** How to run untrusted code securely with almost no overhead for ARM?
- **Solution:** Leverage hardware features of ARM.
- **Detail:** ARMlock leverages memory domain in the page table of commodity ARM processors to create multiple sandboxes. Memory accesses by the untrusted module (including read, write, and execution) are strictly confined by the hardware, and instructions running inside the sandbox execute at the same speed as those outside it.

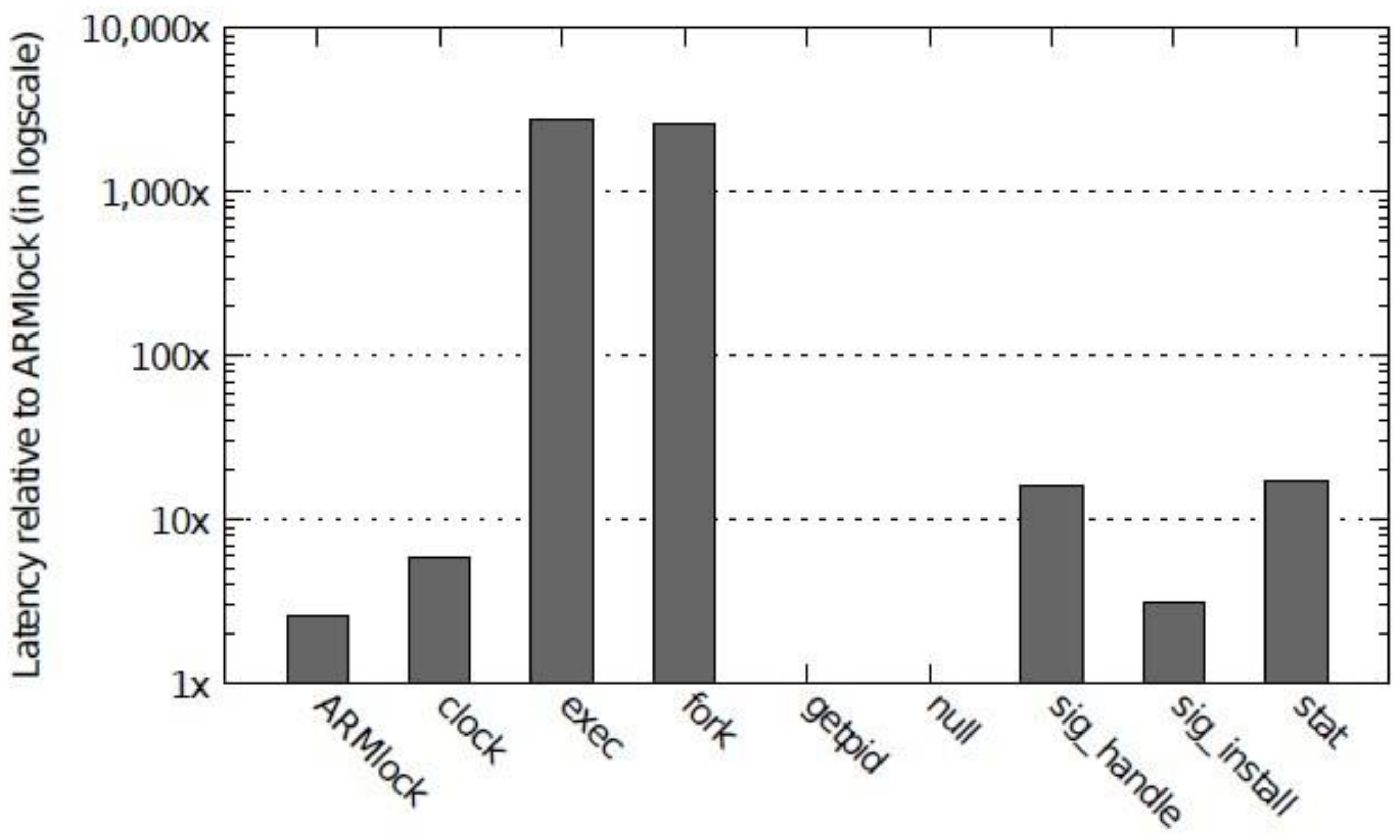


Relative Performance of nbench in ARMlock.

In ARMlock, instructions running inside the sandbox is **as fast as** outside the sandbox.

Advantages:

- ARMlock poses no constraints on untrusted modules. In particular, it support advanced features such as self-modifying code, just-in-time compiling, exception delivery, and making system calls, which are interposed by ARMlock to enforce the policy set by the host.
- Instructions running inside the sandbox virtually has no performance overhead compared to those running outside it.
- Efficient domain switch (perform more than 903,000 domain switches each second even on the low-end ARM processor).



ARMlock domain switch latency relative to that of getpid and other system calls.

ARMlock can perform **903,342** inter-module calls every second.