Evaluating Confidential Computing with Unikernels (Guided Research Project)

Roberto Castellotti
TU Munich

Masanori Misono
TU Munich

Abstract

We report a preliminary performance evaluation of AMD SEV (Secure Environment Virtualization) on a Linux system.

1 Environment

We run our experiments on ryan, we using a patched version of QEMU from AMD. Do we need additional info about the system?

Table 1 shows the detailed environment. We use QEMU/KVM as a hypervisor. We assign the guest the same amount of CPUs (16) and 16G of memory.

2 Micro Benchmarks

2.1 CPU benchmarks

We start by benchmarking the CPU by compling different programs (ffmpeg,gdb,linux kernel, llvm (ninja)) and then we ran the lz4 benchmark.

compilation [6] This measures how much time it takes to compile common programs. Figure 1 shows the results.

lz4 [1] We measure compression and decompression speed (MB/s). Figure 2 and Figure 3 show the results.

2.2 Memory overhead

We measure the memory overhead of TDX using the following benchmarks using phoronix-test-suite [3].

RAMSpeed [4] This measures the memory latency with several operations. **??** shows the results.

Tinymembench [7] This benchmark measures the memory latency of the system. **??** shows the results.

MBW [2] This measures the memory bandwidth of the system. **??** shows the results.

We observe the followings from the results.

- For the RAMSpeed benchmarks, we observe 3.3% overhead for "bare:tme" and 6.38% for "vm:tdx" in geometric mean.
- For the Tinymembench benchmarks, we observe 5.95% overhead for "bare:tme" and 4.42% for "vm:tdx" in geometric mean.
- For the MBW benchmarks, we observe 9.37% overhead for "bare:tme" and 10.52% for "vm:tdx" in geometric mean.
- The overhead of the memory bandwidth (MBW) is larger than the overhead of the memory latency (RAMSpeed, Tinymembench).

3 Application Benchmarks

We measure several application benchmarks using Phoronix Benchmark Suite [3]. We run and redis and SQLite benchmarks as memory-intensive applications.

Redis [5] This measures the times of several MPI parallel applications. **??** shows the results.

SQLite [5] This measures the time to perform a pre-defined number of insertions to a SQLite database. **??** shows the results.

We observe the followings from the results.

- As of compilation and NPB benchmarks, we observe around 10 to up to 60% overhead in the TDX VM ("vm:tdx"). However, vPCU over-commitment might affect these results, so we expect the actual performance will be better.
- As of LZ4 benchmaks, both "vm:notdx" and "vm:tdx" have similar performance. This is because LZ4 is a memory-intensive application, and the main overhead

Table 1: Experiment environment

Host CPU	AMD EPYC 7713P 64-Cores
Host Memory	HMAA8GR7AJR4N-XN (Hynix) 3200MHz 64 GB × 8 (512GB)
Host Config	Automatic numa balancing disabled; Side channel mitigation default (enabled)
Host Kernel	6.1.0-rc4 #1-NixOS SMP PREEMPT_DYNAMIC (NixOS 22.11)
QEMU	7.2.0 (patched)
OVMF	Stable 202211 (patched) ????
Guest vCPU	16
Guest Memory	16GB
Guest Kernel	5.19.0-41-generic #42-Ubuntu SMP PREEMPT_DYNAMIC (Ubuntu 22.10)
Guest Config	No vNUMA; Side channel mitigation default (enabled)

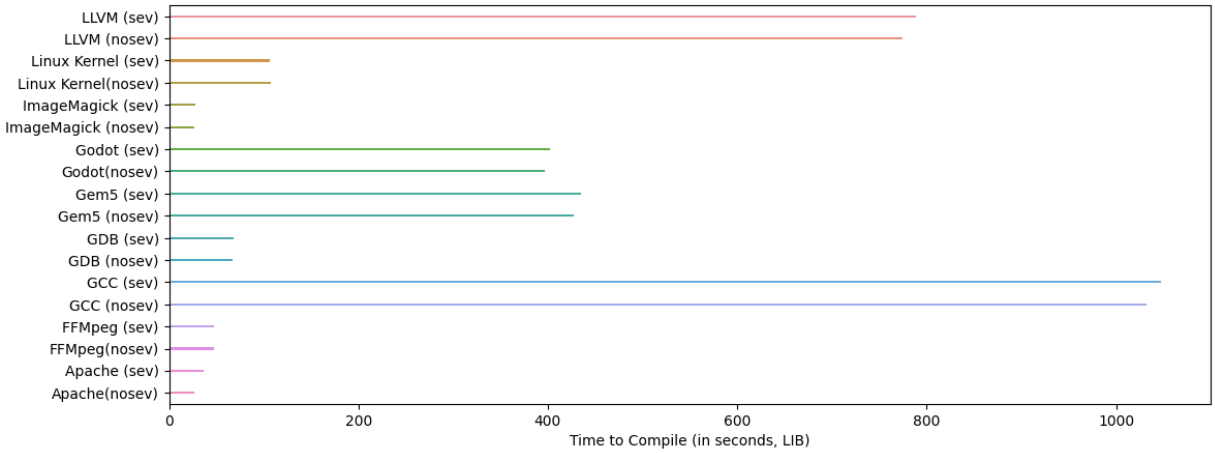

Figure 1: Compilation benchmark results

comes from memory encryption/decryption. NPB and these results also highlight the importance of TME bypass if we want to eliminate the memory encryption overhead in non-TDX VMs.

- As of SQLite benchmarks, we observe larger performance overhead in "vm:tdx" when copy size is larger than 32. This might be due to the vCPU overcommitment, but further investigation is needed.

References

[1] *LZ4 Compression Benchmark - OpenBenchmarking.org*. https://openbenchmarking.org/test/pts/compress-lz4. Accessed: 2023-04-01.

[2] *MBW Benchmark - OpenBenchmarking.org*. https://openbenchmarking.org/test/pts/mbw. Accessed: 2023-04-01.

[3] Phoronix Media. *Phoronix Test Suite - Linux Testing & Benchmarking Platform, Automated Testing, Open-Source Benchmarking*. https://www.phoronix-test-suite.com/. Accessed: 2023-04-01.

[4] *RAMspeed SMP Benchmark - OpenBenchmarking.org*. https://openbenchmarking.org/test/pts/ramspeed. Accessed: 2023-04-01.

[5] *SQLite Benchmark - OpenBenchmarking.org*. https://openbenchmarking.org/test/pts/sqlite. Accessed: 2023-04-01.

[6] *Timed Code Compilation Benchmark - OpenBenchmarking.org*. https://openbenchmarking.org/suite/pts/compilation. Accessed: 2023-04-01.

[7] *Tinymembench Benchmark - OpenBenchmarking.org*. https://openbenchmarking.org/test/pts/tinymembench. Accessed: 2023-04-01.

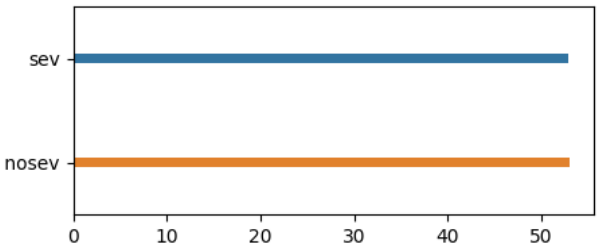

Figure 2: LZ4 compression benchmark results

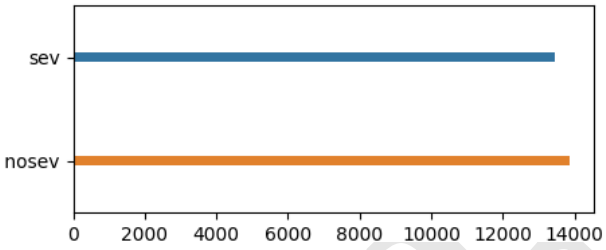

Figure 3: LZ4 decompression benchmark results

Appendix

Check MSR values We can check related MSR values with the following script.

```c
#include "cpuid.h"
#include <stdio.h>
int main()
{
    int eax, ebx, ecx, edx = 0;
    unsigned int leaf = 0x8000001f;
    __get_cpuid(leaf, &eax, &ebx, &ecx, &edx);
    printf("id: %x :: eax %x :: ebx %x\n", leaf,
    eax, ebx);
    if (eax && 0x00000001){
        printf("AMD SEV is supported.\n");
    }
    if (eax && 0x00001000){
        printf("AMD SEV ES is supported.\n");
    }
    if (eax && 0x00010000){
        printf("AMD SEV-SNP is supported.\n");
    }
    else{
        printf("No AMD SEV related technologies
    are enabled\n");
    }
}
```