

Ensure that the solutions for the **first** coursework assignment are stored in the **cw1** directory of your repository. If they are not, move them now, then commit and push the changes.

Place the *cw2.tar.gz* file in the **cw2** directory of your local repository and execute: *tar -xvf cw2.tar.gz*

After executing that command, delete the *cw2.tar.gz* file, then commit and push the extracted files and directories. Each exercise has a specific directory: *cw2/ex1*, *cw2/ex2*, and *cw2/ex3*. When you need to write answers in text, use **.md** (Markdown) or **.adoc** (Asciidoc) files. You may answer in English or Portuguese.

Before starting this assignment, install additional packages in your Ubuntu 24.04 environment, by executing:

```
sudo apt update
sudo apt install qemu-user qemu-user-static
sudo apt install gcc-aarch64-linux-gnu binutils-aarch64-linux-gnu binutils-aarch64-linux-gnu-dbg
```


1. Consider the System Programming Manual for the AMD64 architecture, and answer the questions below:
<https://www.amd.com/content/dam/amd/en/documents/processor-tech-docs/programmer-references/24593.pdf>
 - a. Consider Figures 5-18 through 5-23. What changes would need to be made to these figures if the architectural limit for **physical memory** were to be increased to 64 PB?
 - b. Suppose AMD wants to support **virtual address spaces** of 512 PB. What changes would need to be made to Figure 5.18? How many different “Page-Map Level-5” tables could then exist for a single process? Read sections 1.1.3 and 5.3.1 and indicate the ranges of canonical addresses for the proposed extension.
 - c. While processing the Present (P) bit in Page-Translation-Table Entry Fields (see Section 5.4.1), one of the entries in Table 8-1 is particularly relevant. Which entry is that, and why?
 - d. [OPTIONAL] Use an AI tool designed for assisting with programming and technical inquiries to help you read the code snippet between lines 378 and 397 of the following Linux kernel source code file: https://github.com/torvalds/linux/blob/7234e2ea0edd00bfb6bb2159e55878c19885ce68/arch/x86/kernel/head_64.S#L378 Find information in Section 5.4.1 of the AMD64 manual to explain what is going on between lines 387 and 390.

Tag these answers on the GitHub repository with: **CW2-1**

2. Observe the *x86-64* assembly code in *cw2/ex2/x86-64/prog.s*, and consult the documentation on Linux system calls to determine the purpose of the program. You can use the following resources for reference:
 - <https://man7.org/linux/man-pages/man2/syscalls.2.html>
 - https://chromium.googlesource.com/chromiumos/docs/+/master/constants/syscalls.md#x86_64-64_b

Additionally, you may build the program (*make*) and run it (*./prog*) to complement your analysis. Running the program under [*strace*](#) may also provide useful insights.

- a. In the text file *cw2/ex2/a.md*, list and explain the sequence of calls performed by the program.
- b. In the source file *cw2/ex2/arm64/prog.s*, replace ? and ?? with adequate numbers, such that it becomes a working ARM64 version of *cw2/ex2/x86-64/prog.s*.

 *Note that the system calls are the same, but the preparation instructions differ in order..*

Build the program (*make*) and run it with an emulator: *qemu-aarch64 ./prog*

Tag these answers on the GitHub repository with: **CW2-2**

3. Complete the source code in `cw2/ex3/prog.c` in the indicated places, in order to cause the following effects in the process virtual address space, which can be observed in `/proc/<pid>/smaps` :
- a. Increase the *resident set* (`Rss`) by about 3 MB in the `.bss` region.
 - b. Access 256 bytes of initialized data (`.data`) with maximum impact in `Private Clean` pages.
 - c. Reduce the `Pss` of non-initialized data (`.bss`) to around 1.5 MB for 30 seconds, while keeping `Rss`.
 - d. Execute a single operating system function that results in two new regions being added to the existing address space, one with code (~ 4 KB) and one with data (~ 512 KB).
 - e. Increase `Private dirty` pages by about 128 KB in the region for the data section created in d).

Solve this exercise using `cw2/ex3/prog.c` as the main file. Add other source and/or auxiliary files if needed.

Tag this solution on the GitHub repository with: **CW2-3**

Do not submit binaries and other unneeded files to the repository.

For the absolute final version, use the tag **CW2-DONE** on the GitHub repository.

ISEL, October 14th, 2023

Submission last date: October 26th, 2023