Operating Systems (CS:3620)

Assignment 4 (Paging Based Memory Virtualization) Total points: 100

Due: 9 days (by 11:59 pm of November 6 (Sunday), 2016)

This assignment will contribute 10% to your final grades.

This is an individual assignment.

Book Chapter 18 (Paging): `http://pages.cs.wisc.edu/~remzi/OSTEP/Homework/homework.html`

In this project, you will use a simple program, which is known as `paging-linear-translate.py`, to see if you understand how simple virtual-to-physical address translation works with linear page tables. See the `README` for details.

`paging-linear-translate.py` and the accompanying `README` file can be downloaded from `http://pages.cs.wisc.edu/~remzi/OSTEP/Homework/HW-Paging-LinearTranslate.tgz`. There is a YouTube video of this homework which can be found in `http://youtu.be/AhfSDqud3j4`.

Questions

1. Before doing any translations, let's use the simulator to study how linear page tables change size given different parameters. Compute the size of linear page tables as different parameters change. Some suggested inputs are below; by using the `-v` flag, you can see how many page-table entries are filled. First, to understand how linear page table size changes as the address space grows:

   ```
   paging-linear-translate.py -P 1k -a 1m -p 512m -v -n 0
   paging-linear-translate.py -P 1k -a 2m -p 512m -v -n 0
   paging-linear-translate.py -P 1k -a 4m -p 512m -v -n 0
   ```

   Then, to understand how linear page table size changes as page size grows:

   ```
   paging-linear-translate.py -P 1k -a 1m -p 512m -v -n 0
   paging-linear-translate.py -P 2k -a 1m -p 512m -v -n 0
   paging-linear-translate.py -P 4k -a 1m -p 512m -v -n 0
   ```

   Before running any of these, try to think about the expected trends. How should page-table size change as the address space grows? As the page size grows? Why shouldn't we just use really big pages in general?

2. Now let's do some translations. Start with some small examples, and change the number of pages that are allocated to the address space with the `-u` flag. For example:

   ```
   paging-linear-translate.py -P 1k -a 16k -p 32k -v -u 0
   paging-linear-translate.py -P 1k -a 16k -p 32k -v -u 25
   paging-linear-translate.py -P 1k -a 16k -p 32k -v -u 50
   paging-linear-translate.py -P 1k -a 16k -p 32k -v -u 75
   paging-linear-translate.py -P 1k -a 16k -p 32k -v -u 100
   ```

   What happens as you increase the percentage of pages that are allocated in each address space?

3. Now let's try some different random seeds, and some different (and sometimes quite crazy) address-space parameters, for variety:

   ```
   paging-linear-translate.py -P 8 -a 32 -p 1024 -v -s 1
   paging-linear-translate.py -P 8k -a 32k -p 1m -v -s 2
   paging-linear-translate.py -P 1m -a 256m -p 512m -v -s 3
   ```

   Which of these parameter combinations are unrealistic? Why?

4. Use the program to try out some other problems. Can you find the limits of where the program doesn't work anymore? For example, what happens if the address-space size is bigger than physical memory?