3address-translation-lab

CST 334 (Operating Systems)
Dr. Glenn Bruns

Lab: Address translation

Here are a few important warm-up questions:

1. If a virtual address is 2 bits long, how many different addresses are there?
2. If a virtual address is 4 bits long, what is the size of the virtual memory space (assuming each byte of memory has its own address).
3. If a virtual address is 8 bits in size, what is the size of the virtual memory space?

Now, copy the program â€˜relocation.pyâ€™ from this directory on mlc104

`/home/CLASSES/brunsglenn/OSTEP/HW-Relocation`Â

to a directory of your own, or download it from the textbook site:

pages.cs.wisc.edu/~remzi/OSTEP/Homework/homework.html

When you run the program, you get something like this:

```
$ ./relocation.py

ARG seed 0
ARG address space size 1k
ARG phys mem size 16k

Base-and-Bounds register information:

Â  Base Â  : 0x00003082 (decimal 12418)
Â  Limit Â : 472

Virtual Address Trace
Â  VA Â 0: 0x000001ae (decimal: Â 430) --> PA or segmentation violation?
Â  VA Â 1: 0x00000109 (decimal: Â 265) --> PA or segmentation violation?
Â  VA Â 2: 0x0000020b (decimal: Â 523) --> PA or segmentation violation?
Â  VA Â 3: 0x0000019e (decimal: Â 414) --> PA or segmentation violation?
Â  VA Â 4: 0x00000322 (decimal: Â 802) --> PA or segmentation violation?

For each virtual address, either write down the physical address it translates to OR write
down that it is an out-of-bounds address (a segmentation violation). For this problem, you
should assume a simple virtual address space of a given size.
```

This output lists some questions to answer using base-and-bounds address translation. Â You see the base, the bound (also known as â€œlimitâ€), and then some virtual addresses. Â For each of the five virtual addresses (VA) shown, either compute the corresponding physical address (PA), or say that a segmentation violation has occurred. Â Use decimal numbers.

For example, the first virtual address shown above is 430 decimal, which translates to physical address 12418 + 430. Â It is not a segmentation violation because 430 is less than 472.

Now do the following:

4. Answer question 1 at the end of OSTEP chapter 15. Â (To run with random seed 1, use -s 1 on the command line when you run the program.) Â Quoting the text:

Run with seeds 1, 2, and 3, and compute whether each virtual address generated by the process is in or out of bounds. If in bounds, compute the translation.

As usual, you can check your answers by adding the -c option.

5. Answer question 2 at the end of chapter 15. Â (Use flag -h to see what all the flags of the simulator mean.) Â Quoting the text:

Run with these flags:-s 0 -n 10. What value do you have set -l (the bounds register) to in order to ensure that all the generated virtual addresses are within bounds?

6. Answer question 3 at the end of chapter 15. Â Quoting the text:

Run with these flags: -s 1 -n 10 -l 100. What is the maximum value that base can be set to, such that the address space still fits into physical memory in its entirety?

7. If you still have time, answer question 5 at the end of chapter 15.

What fraction of randomly-generated virtual addresses are valid, as a function of the value of the bounds register? Make a graph from running with different random seeds, with limit values ranging from 0 up to the maximum size of the address space.