INDIAN INSTITUTE OF TECHNOLOGY MADRAS

Operating Systems : Lab 2

SATHVIK JOEL K

CS19B025

September 5, 2021

CONTENTS

1 Summary

In this lab, new system calls are added to xv6. They are

- echo_simple : called from *test_problem_1* file. It takes an argument and prints it from kernel
- echo_kernel : called from *test_problem_2* file. It an extention from above. It takes any number of argumrnts and prints it from kernel.
- trace : called from trace file. It traces any system call given to it as a mask. Prints all the processes forked from the main process. The trace system call enable tracing for the process that calls it and will not effect other processes

2 Problem 1

The following steps are followed

- Added a new user program called *test_problem_1* and called the newly added *echo_simple* from it with the string argument

```
/* in test_problem_1.c */
void main(int argc, char** argv)
{
/* something */
echo_simple(argv[1]);
}
```

- Retrived the argument using helper function argstr and printed from kernel

```
/* in sysproc.c */
argstr(0, str, MAXPATH)
printf("%s\n"str);
```

3 Problem 2

The steps followed are same except that instead of a simple char*, argv is passed and it is retrived using helperfunctions argaddr, fetchaddr and the strings are obtained using fetchstr

4 Problem 3

- trace user program is added using given file *trace.c*
- A syscall *trace* is also added to the list of existing system calls appropriately
- A new member called trace_arg is added to proc struct in *proc.h* file

- In trace sys_call this member of the struct is set to the argument
- And finally whenever a traced process is abt to exit it is printed in appropriate format

5 Problem 4

It is in continuation of the above problem.

In all the system call functions whenever that system call is traced. The arguments are printed. (find a comment saying " tracing functionality " in syscalls whereever this functinality is added)