PSET 1

Jonathan Lam
Prof. Hakner
ECE 357
Operating Systems
9/4/19

Problem 1: What is a system call and what is not?

name	syscall?	triggers?	triggers what?
read	Y	–	–
fputc	N	M – only on buffer flush	write
strcpy	N	N	–
sqrt	N	N	–
malloc	N	M – if program runs out of heap space	sbrk, sometimes brk/mmap
fopen	N	Y	open/openat
strerror	N	N	–
isalpha	N	N	–
atoi	N	N	–
scanf	N	Y	read
return	N	M – if returning from main (ending the process)	_exit

Problem 2: Error messages

A) close is called with a parameter of -1
<u>EBADF Bad file descriptor</u>

B) write is made to a file which resides on a disk that is completely full
<u>ENOSPC No space left on device</u>

C) open is called with its first parameter referring to a non-existent file and second parameter of O_RDONLY
<u>ENOENT No such file or directory</u>

D) write is made with the second parameter of 0, a first parameter which refers to a valid fd open for writing, and a third parameter >0
<u>EFAULT Bad address</u>

Problem 3: Use of system calls in a simple concatenation program

<u>kitty.c</u>

```
#include <fcntl.h>
#include <stdio.h>
#include <string.h>
#include <errno.h>
#include <unistd.h>
```

PSET 1 Jonathan Lam
Prof. Hakner
ECE 357
Operating Systems
9/4/19

```c
#define BUF_SIZE 4096
int main(int argc, char **argv) {
  int ofd, ifd, flen, rlen, wlen, is_bin, rwcnt;
  char buf[BUF_SIZE], *bufp, *fnames[argc], **fnamep, *out_file = NULL,
       *errop, *errctx, *errctm;
  memset(fnames, 0, argc*sizeof(char *));

  // parse args
  for(fnamep = fnames, ++argv; --argc; ++argv)
    if(!strcmp(*argv, "-o")) {
      if(!--argc) {
        errop = "parsing", errctx = "args", errctm = "No output file after -o";
        goto fail;
      }
      out_file = *++argv;
    } else
      *fnamep++ = *argv;

  // open out_file using creat (open w/ flags O_CREAT|O_WRONLY|O_TRUNC)
  if(out_file) {
    if((ofd = creat(out_file, 0666)) == -1) {
      errop = "creating (for writing)", errctx = out_file;
      goto fail;
    }
  } else
    ofd = 1;

  // if no input file specified (fnames empty), add std. input to input list
  if(!*fnames)
    *fnames = "-";

  // loop through and open input files, concatenate to output
  for(fnamep = fnames; *fnamep; fnamep++) {
    if(strcmp(*fnamep, "-")) {
      if((ifd = open(*fnamep, O_RDONLY)) == -1) {
        errop = "opening (for reading)", errctx = *fnamep;
        goto fail;
      }
    } else
      ifd = 0;

    // attempt reading file
    flen = is_bin = rwcnt = 0;
    while(rlen = read(ifd, buf, BUF_SIZE)) {
      if(rlen == -1) {
        errop = "reading of", errctx = *fnamep;
        goto fail;
      }

      // write to output file
      if((wlen = write(ofd, buf, rlen)) == -1) {
        errop = "writing to", errctx = out_file;
        goto fail;
      }

      // account for partial write scenario
      if(wlen != rlen) {
        errop = "writing to", errctx = out_file, errctm = "Partial write";
        goto fail;
      }

      // add to total length (in bytes) and read/write count
      flen += rlen;
      ++rwcnt;

      // check if file includes binary chars
      if(!is_bin)
        for(bufp = buf; bufp-buf < rlen; bufp++)
          if((*bufp < 32 || *bufp >= 127) && !(*bufp >= 9 && *bufp <= 13)) {
            is_bin = 1;
            break;
          }
    }

    // report bytes transferred for file
    fprintf(stdout, "%s%s: %d bytes transferred. %d read/write call(s).\n",
            ifd ? *fnamep : "<standard input>", is_bin ? " [BINARY]" : "",
            flen, rwcnt);
  }

  return 0;

fail:
  fprintf(stderr, "Error: %s %s: %s\n",
          errop, errctx, errno ? strerror(errno) : errctm);
  return -1;
}
```

kitty.c sample runs

```
(base) [jon@archijon programs]$ ./kitty -o file1
Hello, world!
```

PSET 1

Jonathan Lam
Prof. Hakner
ECE 357
Operating Systems
9/4/19

```
This is file1

<standard input>: 29 bytes transferred. 3 read/write call(s).
(base) [jon@archijon programs]$ echo 'This is file2' > file2
(base) [jon@archijon programs]$ echo -e 'file3\nfile3\nfile3' > file3
(base) [jon@archijon programs]$ ./kitty file1 file2 file3
Hello, world!
This is file1

file1: 29 bytes transferred. 1 read/write call(s).
This is file2
file2: 14 bytes transferred. 1 read/write call(s).
file3
file3
file3
file3: 18 bytes transferred. 1 read/write call(s).
(base) [jon@archijon programs]$ ./kitty file1 file2 file3 -o file4
file1: 29 bytes transferred. 1 read/write call(s).
file2: 14 bytes transferred. 1 read/write call(s).
file3: 18 bytes transferred. 1 read/write call(s).
(base) [jon@archijon programs]$ ./kitty file4
Hello, world!
This is file1

This is file2
file3
file3
file3
file4: 61 bytes transferred. 1 read/write call(s).
(base) [jon@archijon programs]$ dd if=/dev/urandom of=rand1 bs=1M count=50
50+0 records in
50+0 records out
52428800 bytes (52 MB, 50 MiB) copied, 0.310224 s, 169 MB/s
(base) [jon@archijon programs]$ ./kitty rand1 -o rand2
rand1 [BINARY]: 52428800 bytes transferred. 12800 read/write call(s).
(base) [jon@archijon programs]$ sha256sum rand1 rand2
f5c0c772512b1b177fa7144e92d637c0a7d608b75e22601646f9b7e15c5d9870  rand1
f5c0c772512b1b177fa7144e92d637c0a7d608b75e22601646f9b7e15c5d9870  rand2
(base) [jon@archijon programs]$ ./kitty -o file5 -
This is to go in file5
Hello, world!
<standard input>: 37 bytes transferred. 2 read/write call(s).
(base) [jon@archijon programs]$ ./kitty - - file5 - -o file6
This is to
go in
<standard input>: 17 bytes transferred. 2 read/write call(s).
file6:
```

Jonathan Lam
Prof. Hakner
ECE 357
Operating Systems
9/4/19

```
<standard input>: 7 bytes transferred. 1 read/write call(s).
file5: 37 bytes transferred. 1 read/write call(s).
End of file6.
<standard input>: 14 bytes transferred. 1 read/write call(s).
(base) [jon@archijon programs]$ ./kitty file6
This is to
go in
file6:
This is to go in file5
Hello, world!
End of file6.
file6: 75 bytes transferred. 1 read/write call(s).
(base) [jon@archijon programs]$ ./kitty kitty.c
/**
  * kitty - concatenate and copy files
  *
                                    [TRUNCATED]
          errop, errctx, errno ? strerror(errno) : errctm);
  return -1;
}
kitty.c: 2843 bytes transferred. 1 read/write call(s).
(base) [jon@archijon programs]$ ./kitty kitty
ELF>�@�;@8
          @@@@h���   �-�=�=���-�=�=����DDP�td� � � 44Q�tdR�td�-�=�=/lib64/ld-linux-
                                    [TRUNCATED]
�  �� � 4� ! !������=�-��?��@�h@h�@x00 �0x0�0�^��7��:kitty [BINARY]: 17112 bytes
transferred. 5 read/write call(s).
(base) [jon@archijon programs]$ ./kitty kitty kitty.c /usr/bin/cat -o kittykittycat
kitty [BINARY]: 17112 bytes transferred. 5 read/write call(s).
kitty.c: 2843 bytes transferred. 1 read/write call(s).
/usr/bin/cat [BINARY]: 38952 bytes transferred. 10 read/write call(s).
(base) [jon@archijon programs]$ ./kitty -o -

This is to go inside the file "-". This can be kittied using ./-
<standard input>: 66 bytes transferred. 2 read/write call(s).
(base) [jon@archijon programs]$ ./kitty ./-

This is to go inside the file "-". This can be kittied using ./-
./-: 66 bytes transferred. 1 read/write call(s).
(base) [jon@archijon programs]$ ./kitty kitty.c -o
Error: parsing args: No output file after -o
(base) [jon@archijon programs]$ ./kitty nonexistentfile.txt
Error: opening (for reading) nonexistentfile.txt: No such file or directory
(base) [jon@archijon programs]$ touch badpriv
(base) [jon@archijon programs]$ chmod 000 badpriv
(base) [jon@archijon programs]$ ./kitty badpriv
Error: opening (for reading) badpriv: Permission denied
```

PSET 1

Jonathan Lam
Prof. Hakner
ECE 357
Operating Systems
9/4/19

kitty.c

```
#include <fcntl.h>
#include <stdio.h>
#include <string.h>
#include <errno.h>
#include <unistd.h>

#define BUF_SIZE 4096
#define MAX_PAR_WRITE_RETRY 256

int main(int argc, char **argv) {
  int ofd = -1, ifd = -1, flen, rlen, wlen, is_bin, rcnt, wcnt, par_retry_count;
  char buf[BUF_SIZE], *bufp, *fnames[argc+1], **fnamep, *out_file = NULL,
       *errop, *errctx, *errctm;
  memset(fnames, 0, (argc+1)*sizeof(char *));

  // parse args; only handles -o argument (handles others as filenames)
  for(fnamep = fnames, ++argv; --argc; ++argv)
    if(!strcmp(*argv, "-o")) {
      if(!--argc) {
        errop = "parsing", errctx = "args", errctm = "No output file after -o";
        goto fail;
      }
      out_file = *++argv;
    } else
      *fnamep++ = *argv;

  // open out_file using creat (open w/ flags O_CREAT|O_WRONLY|O_TRUNC)
  if(out_file) {
    if((ofd = creat(out_file, 0666)) == -1) {
      errop = "creating (for writing)", errctx = out_file;
      goto fail;
    }
  } else
    ofd = 1;

  // if no input file specified (fnames empty), add std. input to input list
  if(!*fnames)
    *fnames = "-";

  // loop through and open input files, concatenate to output
  for(fnamep = fnames; *fnamep; fnamep++) {
    if(strcmp(*fnamep, "-")) {
      if((ifd = open(*fnamep, O_RDONLY)) == -1) {
        errop = "opening (for reading)", errctx = *fnamep;
        goto fail;
      }
    } else
      ifd = 0;

    // attempt reading file
    flen = is_bin = rcnt = wcnt = 0;
    while(rcnt++, rlen = read(ifd, buf, BUF_SIZE)) {
      if(rlen == -1) {
        errop = "reading of", errctx = *fnamep;
        goto fail;
      }

      // write to output file
      // account for partial write scenario; most likely due to a pipe/socket
      // with a small buffer; keep retrying until exceeded maximum tries or
      // write complete; while loop breaks when buffer successfully written
      wlen = 0, par_retry_count = 0;
      while(wcnt++, (wlen += write(ofd, buf+wlen, rlen-wlen)) != rlen) {
        if(++par_retry_count == MAX_PAR_WRITE_RETRY) {
          errop = "writing to", errctx = out_file, errctm = "Partial write";
          goto fail;
        }

        // write error
        if(wlen == -1) {
          errop = "writing to", errctx = out_file;
          goto fail;
        }
      }

      // add to total length (in bytes) and read/write count
      flen += rlen;

      // check if file includes binary chars
      if(!is_bin)
        for(bufp = buf; bufp-buf < rlen; bufp++)
          if((*bufp < 32 || *bufp >= 127) && !(*bufp >= 9 && *bufp <= 13)) {
            is_bin = 1;
            break;
          }
    }

    // close input file
    if(ifd > 2 && close(ifd) == -1) {
      errop = "closing", errctx = *fnamep;
      goto fail;
    }
```

Jonathan Lam
Prof. Hakner
ECE 357
Operating Systems
9/4/19

```
    // report bytes transferred for file to stderr
    fprintf(stderr, "%s%s: %d bytes transferred. %d read / %d write call(s).\n",
            ifd ? *fnamep : "<standard input>", is_bin ? " [BINARY]" : "",
            flen, rcnt, wcnt);
  }

  // close output file and exit
  if(ofd > 2 && close(ofd) == -1) {
    errop = "closing", errctx = out_file, ofd = -1;
    goto fail;
  }
  return 0;

fail:
  fprintf(stderr, "Error: %s %s: %s\n",
          errop, errctx, errno ? strerror(errno) : errctm);

  // attempt to close input/output files
  // silently fail here because files will automatically be closed anyway
  // and to avoid extra errors printed to screen
  if(ofd != -1)
    close(ofd);
  if(ifd != -1)
    close(ifd);
  return -1;
}
```

```
PSET 1                                                      Jonathan Lam
                                                            Prof. Hakner
                                                                 ECE 357
                                                       Operating Systems
                                                                  9/4/19
```

Example output

```
(base) [jon@archijon programs]$ echo -e 'Hello, world!\nThis is file1\n\ntesting' >
file1
(base) [jon@archijon programs]$ ./kitty -o file2
This is file2
<standard input>: 14 bytes transferred. 2 read / 1 write call(s).
(base) [jon@archijon programs]$ cat > file3
file3, file3, file3
(base) [jon@archijon programs]$ ./kitty file1 file2 file3 -o file4 && ./kitty file4
file1: 37 bytes transferred. 2 read / 1 write call(s).
file2: 14 bytes transferred. 2 read / 1 write call(s).
file3: 20 bytes transferred. 2 read / 1 write call(s).
Hello, world!
This is file1

testing
This is file2
file3, file3, file3
file4: 71 bytes transferred. 2 read / 1 write call(s).
(base) [jon@archijon programs]$ dd if=/dev/urandom of=rand bs=1M count=50
50+0 records in
50+0 records out
52428800 bytes (52 MB, 50 MiB) copied, 0.307862 s, 170 MB/s
(base) [jon@archijon programs]$ ./kitty rand -o rand2
rand [BINARY]: 52428800 bytes transferred. 12801 read / 12800 write call(s).
(base) [jon@archijon programs]$ cat rand > rand3
(base) [jon@archijon programs]$ sha256sum rand rand2 rand3
901f72e6755ab3186fa0f1c80dc9773c19ec44aa7d53ae8543fff03276da2e86  rand
901f72e6755ab3186fa0f1c80dc9773c19ec44aa7d53ae8543fff03276da2e86  rand2
901f72e6755ab3186fa0f1c80dc9773c19ec44aa7d53ae8543fff03276da2e86  rand3
(base) [jon@archijon programs]$ ./kitty -o file5 -
This is to go in file5
Hello, world!
<standard input>: 37 bytes transferred. 3 read / 2 write call(s).
(base) [jon@archijon programs]$ ./kitty - - file5 - -o file6
This is to
go in
<standard input>: 17 bytes transferred. 3 read / 2 write call(s).
file6
<standard input>: 6 bytes transferred. 2 read / 1 write call(s).
file5: 37 bytes transferred. 2 read / 1 write call(s).
End of file6
<standard input>: 13 bytes transferred. 2 read / 1 write call(s).
(base) [jon@archijon programs]$ ./kitty file6
This is to
go in
```

Jonathan Lam
Prof. Hakner
ECE 357
Operating Systems
9/4/19

```
file6
This is to go in file5
Hello, world!
End of file6
file6: 73 bytes transferred. 2 read / 1 write call(s).
(base) [jon@archijon programs]$ ./kitty -o file7 kitty.c kitty
kitty.c: 3511 bytes transferred. 2 read / 1 write call(s).
kitty [BINARY]: 17120 bytes transferred. 6 read / 5 write call(s).
(base) [jon@archijon programs]$ cat kitty.c kitty > file8
(base) [jon@archijon programs]$ sha256sum file7 file8
a8d8e491f40d9812b3d9f502e3a5e07f247d0ae51aacfde2a9c042d1389adf8e  file7
a8d8e491f40d9812b3d9f502e3a5e07f247d0ae51aacfde2a9c042d1389adf8e  file8
(base) [jon@archijon programs]$ ./kitty file7
#include <fcntl.h>
#include <stdio.h>
#include <string.h>
```

[TRUNCATED]

```
tag.gnu.hash.dynsym.dynstr.gnu.version.gnu.version_r.rela.dyn.rela.plt.init.text.fi
ni.rodata.eh_frame_hdr.eh_frame.init_array.fini_array.dynamic.got.got.plt.data.bss.
comment�#��$6�� D��No
�  �� � 4�(!(!������=���?��@�p@p�@�0 �0�0�0��o���7��:file7 [BINARY]: 20631 bytes
transferred. 7 read / 6 write call(s).
(base) [jon@archijon programs]$ ./kitty -o ./-
This is the file "-"
<standard input>: 21 bytes transferred. 2 read / 1 write call(s).
(base) [jon@archijon programs]$ ./kitty ./-
This is the file "-"
./-: 21 bytes transferred. 2 read / 1 write call(s).
(base) [jon@archijon programs]$ ./kitty -o
Error: parsing args: No output file after -o
(base) [jon@archijon programs]$ ./kitty nonexistentfile.txt
Error: opening (for reading) nonexistentfile.txt: No such file or directory
(base) [jon@archijon programs]$ touch badpriv
(base) [jon@archijon programs]$ chmod 000 badpriv
(base) [jon@archijon programs]$ ./kitty badpriv
Error: opening (for reading) badpriv: Permission denied
(base) [jon@archijon programs]$ ./kitty
hello
hello
world
world
!
!
<standard input>: 14 bytes transferred. 4 read / 3 write call(s).
```