Recitation 10: Tshlab + VM

Instructor: TAs
Outline

- Labs
- Signals
- IO
- Virtual Memory
TshLab and MallocLab

- TshLab due Tuesday

- MallocLab is released immediately after
  - Students overwhelming say, “Start early!”
  - For many, checkpoint will take half of the time
  - Expect a “bootcamp” and recitation next week
    - Working for several hours will improve their value significantly
Signals

- Parent process sends SIGINT to a child process. What is the behavior of the child?
- What is the default?
- What else could the child do?
More Signals

- Parent process sends SIGKILL to a child process. What is the behavior of the child?

- What is the default?

- What else could the child do?
Sending Signals

- Parent sends SIGKILL to a child process.

... pid_t pid = ...; // child pid
kill(pid, SIGKILL);
// At this point, what has happened
// to the child process?
Signals

- How many times is Hi printed?

```c
int main(int argc, char** argv)
{
    pid_t ppid = getpid(), cpid, tpid;
    cpid = fork();
    if (cpid == 0) tpid = ppid;
    else tpid = cpid;
    kill(tpid, SIGINT);
    write(STDOUT_FILENO, "Hi", strlen("Hi"));
    return 0;
}
```
Blocking Signals

- The shell is currently running its handler for SIGCHLD.

- What signals can it receive?
- What signals can it not receive (i.e., blocked)?
File descriptor management can be tricky.

How many file descriptors are open in the parent process at the indicated point?

How many does each child have open at the call to execve?

```c
int main(int argc, char** argv)
{
    int i;
    for (i = 0; i < 4; i++)
    {
        int fd = open("foo", O_RDONLY);
        pid_t pid = fork();
        if (pid == 0)
        {
            int ofd = open("bar", O_RDONLY);
            execve(...);
        }
    }
    // How many file descriptors are open in the parent?
```
Redirecting IO

- File descriptors can be directed to identify different open files.

```c
int main(int argc, char** argv)
{
    int i;
    for (i = 0; i < 4; i++)
    {
        int fd = open("foo", O_RDONLY);
        pid_t pid = fork();
        if (pid == 0)
        {
            int ofd = open("bar", O_WRONLY);
            dup2(fd, STDIN_FILENO);
            dup2(ofd, STDOUT_FILENO);
            execve(...);
        }
    }

    // How many file descriptors are open in the parent?
```
Redirecting IO

- At the two points (A and B) in main, how many file descriptors are open?

```c
int main(int argc, char** argv)
{
    int i, fd;
    fd = open("foo", O_WRONLY);
    dup2(fd, STDOUT_FILENO);
    // Point A
    close(fd);
    // Point B
    ...
```
Memory Access

- The processor tries to write to a memory address.
- List different steps that are required to complete this operation.
Memory Access

- The processor tries to write to a memory address.
- List different steps that are required to complete this operation. (non exhaustive list)

  - Virtual to physical address conversion (TLB lookup)
  - TLB miss
  - Page fault, page loaded from disk
  - TLB updated, check permissions
  - L1 Cache miss (and L2 ... and)
  - Request sent to memory
  - Memory sends data to processor
  - Cache updated
Address Translation with TLB

- Translate 0x15213, given the contents of the TLB and the first 32 entries of the page table below.

- **1MB Virtual Memory**
  - 256KB Physical Memory
  - 4KB page size

<table>
<thead>
<tr>
<th>Index</th>
<th>Tag</th>
<th>PPN</th>
<th>Valid</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>05</td>
<td>13</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>3F</td>
<td>15</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>10</td>
<td>0F</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>0F</td>
<td>1E</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>1F</td>
<td>01</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>11</td>
<td>1F</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>03</td>
<td>2B</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>1D</td>
<td>23</td>
<td>0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>VPN</th>
<th>PPN</th>
<th>Valid</th>
<th>VPN</th>
<th>PPN</th>
<th>Valid</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>17</td>
<td>1</td>
<td>10</td>
<td>26</td>
<td>0</td>
</tr>
<tr>
<td>01</td>
<td>28</td>
<td>1</td>
<td>11</td>
<td>17</td>
<td>0</td>
</tr>
<tr>
<td>02</td>
<td>14</td>
<td>1</td>
<td>12</td>
<td>0E</td>
<td>1</td>
</tr>
<tr>
<td>03</td>
<td>0B</td>
<td>0</td>
<td>13</td>
<td>10</td>
<td>1</td>
</tr>
<tr>
<td>04</td>
<td>26</td>
<td>0</td>
<td>14</td>
<td>13</td>
<td>1</td>
</tr>
<tr>
<td>05</td>
<td>13</td>
<td>0</td>
<td>15</td>
<td>18</td>
<td>1</td>
</tr>
<tr>
<td>06</td>
<td>0F</td>
<td>1</td>
<td>16</td>
<td>31</td>
<td>1</td>
</tr>
<tr>
<td>07</td>
<td>10</td>
<td>1</td>
<td>17</td>
<td>12</td>
<td>0</td>
</tr>
<tr>
<td>08</td>
<td>1C</td>
<td>0</td>
<td>18</td>
<td>23</td>
<td>1</td>
</tr>
<tr>
<td>09</td>
<td>25</td>
<td>1</td>
<td>19</td>
<td>04</td>
<td>0</td>
</tr>
<tr>
<td>0A</td>
<td>31</td>
<td>0</td>
<td>1A</td>
<td>0C</td>
<td>1</td>
</tr>
<tr>
<td>0B</td>
<td>16</td>
<td>1</td>
<td>1B</td>
<td>2B</td>
<td>0</td>
</tr>
<tr>
<td>0C</td>
<td>01</td>
<td>0</td>
<td>1C</td>
<td>1E</td>
<td>0</td>
</tr>
<tr>
<td>0D</td>
<td>15</td>
<td>0</td>
<td>1D</td>
<td>3E</td>
<td>1</td>
</tr>
<tr>
<td>0E</td>
<td>0C</td>
<td>0</td>
<td>1E</td>
<td>27</td>
<td>1</td>
</tr>
<tr>
<td>0F</td>
<td>2B</td>
<td>1</td>
<td>1F</td>
<td>15</td>
<td>1</td>
</tr>
</tbody>
</table>
If you get stuck on TshLab

- Read the writeup!
- Do manual unit testing before runtrace and sdriver!

- Read the writeup!
- Post private questions on piazza!

- Read the man pages on the syscalls.
  - Especially the error conditions
  - What errors should terminate the shell?
  - What errors should be reported?