09. Processes

CAOS, MIPT 2019

What is a process?

- An instance of the program in one of execution states
- Isolated virtual address space in UNIX

Process attributes

- Memory
  - Registries, mmaps, stacks
- Filesystem
  - Fds, pathes, umask
- Other
  - Envvars, limits, uids, …

Processes info

- ps -A - all process
- top – list all process by resources consumption
- /proc filesystem

Tree of processes

- Pid == 1 – systemd (init)
- Every process has a parent
- If parent dies before child, systemd becomes parent
- Parent needs to check if child dies

Processes states

- Running
- Suspended – waiting for event, can be finished;
- sTopped – waiting for explicit wake up
- Zombie

Schedulers

- FIFO
- Priority scheduling (man nice)
- Round-robin
- Multilevel FIFO
- etc.

sched_yield()

```
while (1) {
    // do nothing – just waste CPU
}

while (1) {
    shed_yield(); // OK
}
```

Creating process

pid_t result = fork();

- Creates almost exact (except for ids, signals, timers, locks) copy of current process

- -1 – error
- 0 – parent for child process
- >0 – child for parent process

Limits

- /proc/sys/kernel/pid_max – max number of simultaneously launched processes
- /proc/sys/kernel/threads_max – max number of simultaneously executing threads

Fork bomb

```
void fork_bomb() {
    pid_t p;
    do {
        p = fork();
    } while (p != -1);
    while (1) sched_yield();
}
```

Finishing process

- _exit(int) syscall
- exit(int) function (calls atexit handlers; flushes stdio; deletes tmpfiles)
- return int in main

- kill -<signum> <pid>

Checking child

- man wait / waitpid
- man wait3 / wait4

- WIFEXITED(wstatus) – if was ended by _exit
- WIFSIGNALED(wstatus) – if was forced to die
- WEXITSTATUS(wstatus) – get exit code from 0 to 255
- WTERMSIG(wstatus) – get signal if was killed

```
int status;
waitpid(child, &status, 0);
if (WIFEXITED(status)) {
   printf("Exit code: %d",
WEXITSTATUS(status));
} else if (WIFSIGNALED(status)) {
   printf("Killed by %d signal",
   WTERMSIG(status));
}
```

Replace child’s program

man 3 exec

- l – variadic arguments count; v – array of arguments
- e – pass envvars
- p – search programs in $PATH

```
int main() {
    pid_t pid = fork();
    if (pid == -1) { perror("fork"); exit(1);
}
    if (pid == 0) {
        execlp("ls", "ls", "-l", NULL);
        perror("exec"); exit(2);
    } else {
        waitpid(pid, NULL, 0);
    }
}
```

Attributes of process that exec saves

- File descriptors
- Curdir
- Limits
- UID, GID