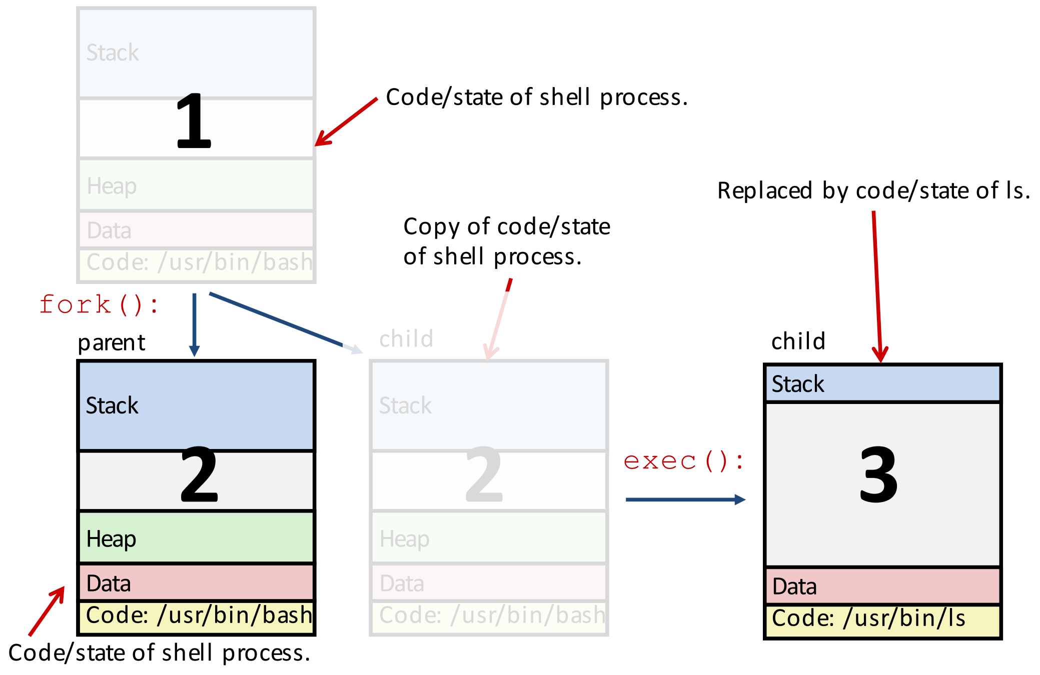
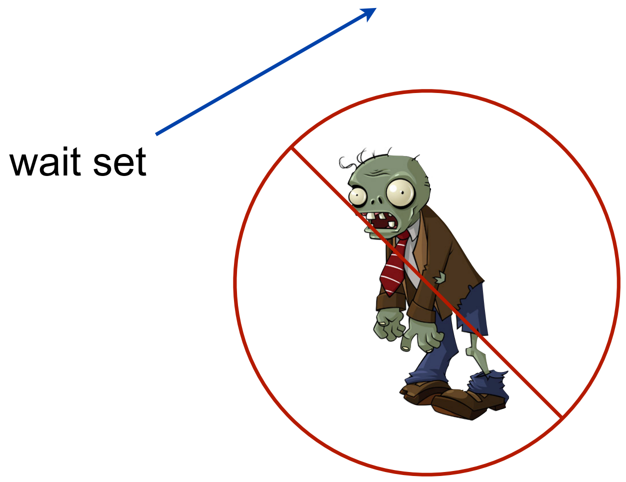
Fork/Exec



Reaping: waitpid

```
pid_t waitpid(pid_t pid, int* stat, int ops)
```



Status Macros

```
pid_t waitpid(pid_t pid, int* stat, int ops)
```

`WEXITSTATUS(stat)`	child exit code
`WIFEXITED(stat)`	true if terminated normally (called exit or returned from main)
`WIFSIGNALED(stat)`	true if terminated by signal
`WIFSTOPPED(stat)`	true if stopped (not terminated) by signal

Option Macros

```
pid_t waitpid(pid_t pid, int* stat, int ops)
```

`WNOHANG`	return immediately if child not already terminated
`WUNTRACED`	also wait for stopped children
`WCONTINUED`	also wait for resumed children

System Call Error Handling

Always check return values!

```
    if ((pid = fork()) < 0) {
        fprintf(stderr, "fork error: %s\n", strerror(errno));
        exit(0);
    }
```

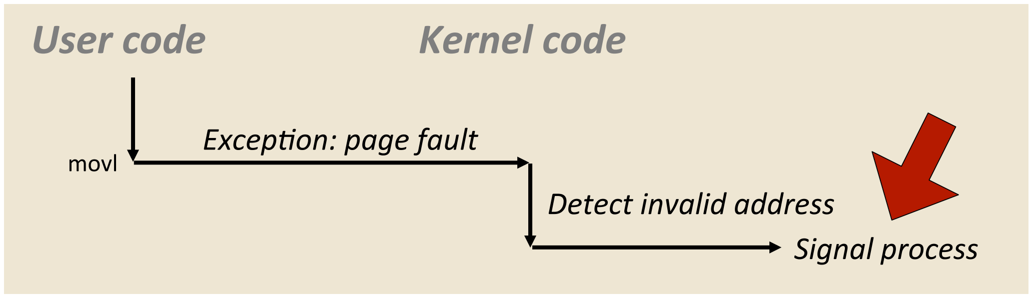
Shell Design

```
while (true) {
    Print command prompt.
    Read command line from user.
    Parse command line.
    If command is built-in, do it.
    Else fork process to execute command.
        in child:
            Execute requested command with execv.
                                     (never returns)
        in parent:
            Wait for child to complete.
}
```

Recap: Segmentation Fault

```
int a[1000];
main ()
{
    a[5000] = 13;
}
```

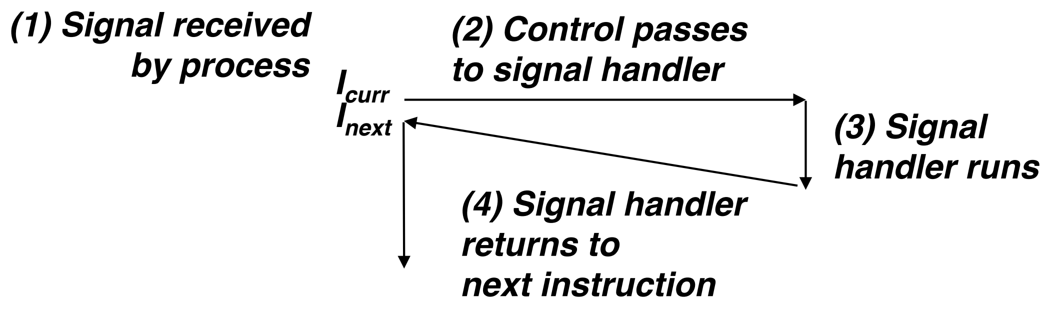
```
 80483b7:       c7 05 60 e3 04 08 0d   movl   $0xd,0x804e360
```



Signals

ID	Name	Corresponding Event	Default Action	Can Override?
2	SIGINT	Interrupt (Ctrl-C)	Terminate	Yes
9	SIGKILL	Kill process (immediately)	Terminate	**No**
11	SIGSEGV	Segmentation violation	Terminate & Dump	Yes
14	SIGALRM	Timer signal	Terminate	Yes
15	SIGTERM	Kill process (politely)	Terminate	Yes
17	SIGCHLD	Child stopped or terminated	Ignore	Yes
18	SIGCONT	Continue stopped process	Continue (Resume)	No
19	SIGSTOP	Stop process (immediately)	Stop (Suspend)	**No**
20	SIGTSTP	Stop process (politely)	Stop (Suspend)	Yes

Signal Control Flow


(1) Signal received by process
I_{curr}
I_{next}
(2) Control passes to signal handler
(3) Signal handler runs
(4) Signal handler returns to next instruction