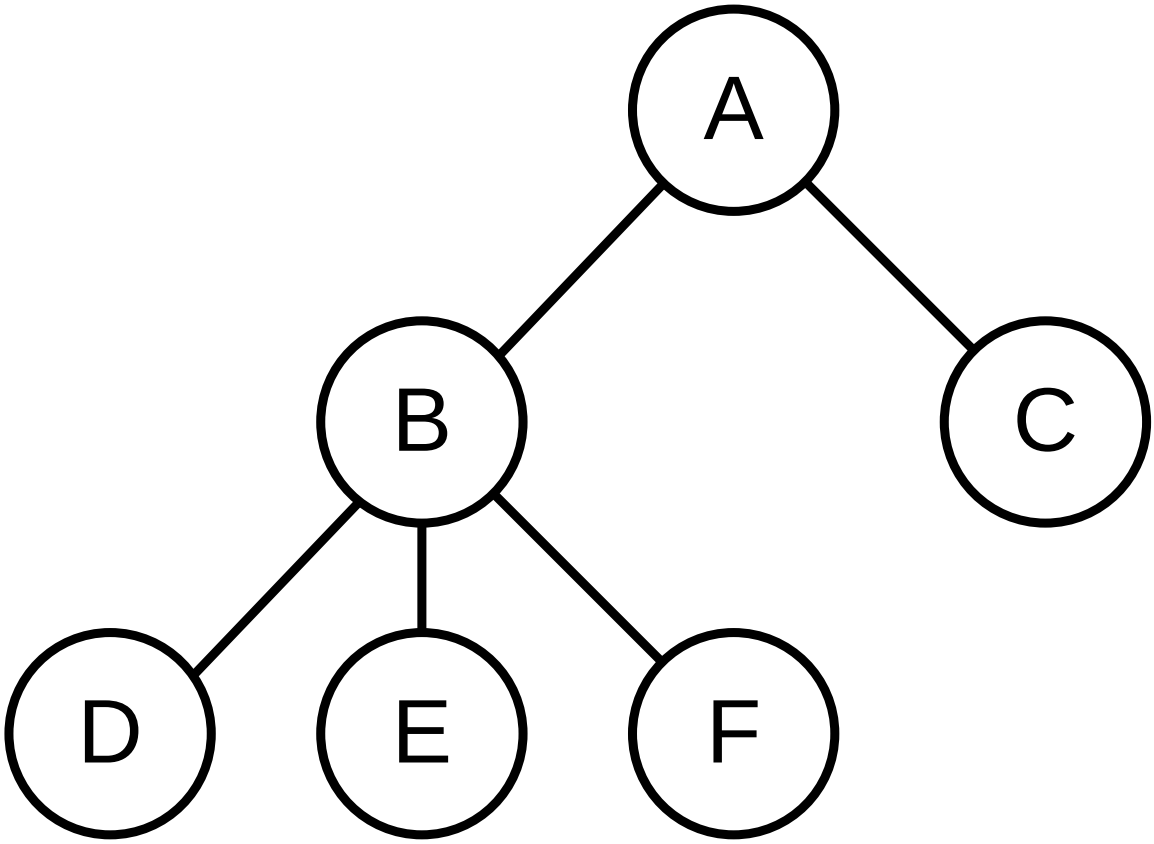

Figure 1-5. A process tree. Process *A* created two child processes, *B* and *C*. Process *B* created three child processes, *D*, *E*, and *F*.

```
while (TRUE) {                          /* repeat forever */
  read_command(command, parameters);/* read input from terminal */

  if (fork() != 0) {                    /* fork off child process */
       /* Parent code. */
       waitpid(–1, &status, 0);         /* wait for child to exit */
  } else {
       /* Child code. */
       execve(command, parameters, 0);/* execute command */
  }
}
```

Figure 1-10. A stripped-down shell. Throughout this book, *TRUE* is assumed to be defined as 1.

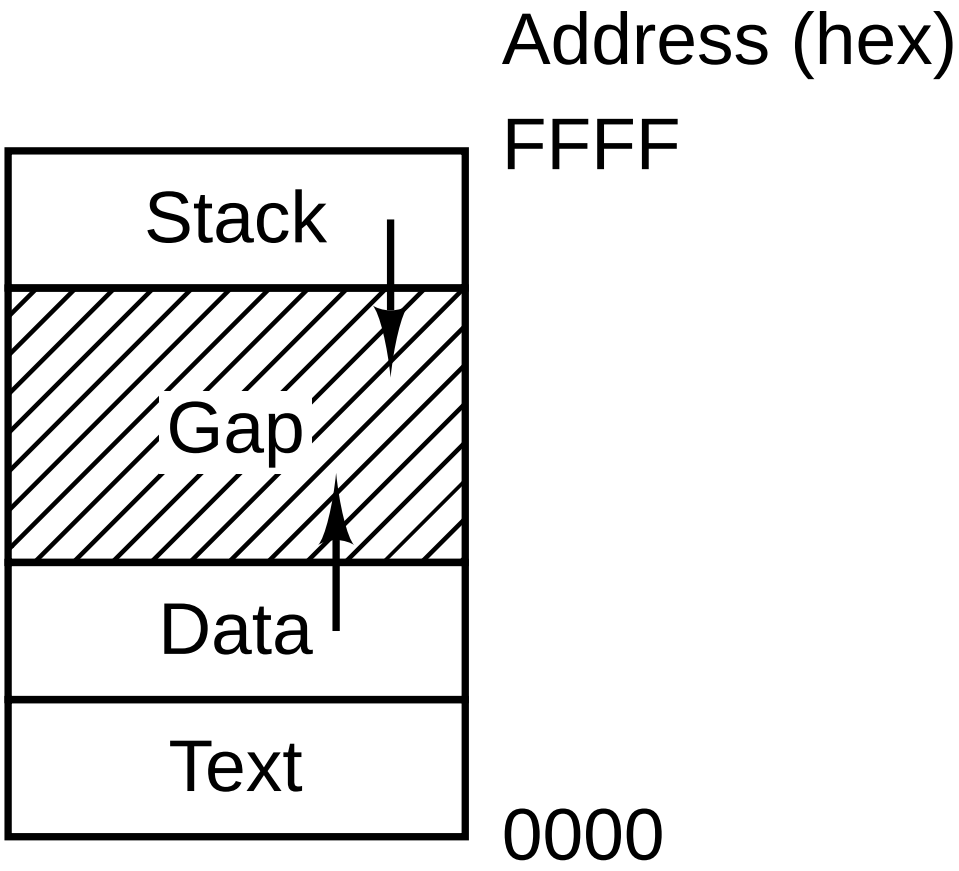

Figure 1-11. Processes have three segments: text, data, and stack. In this example, all three are in one address space, but separate instruction and data space is also supported.

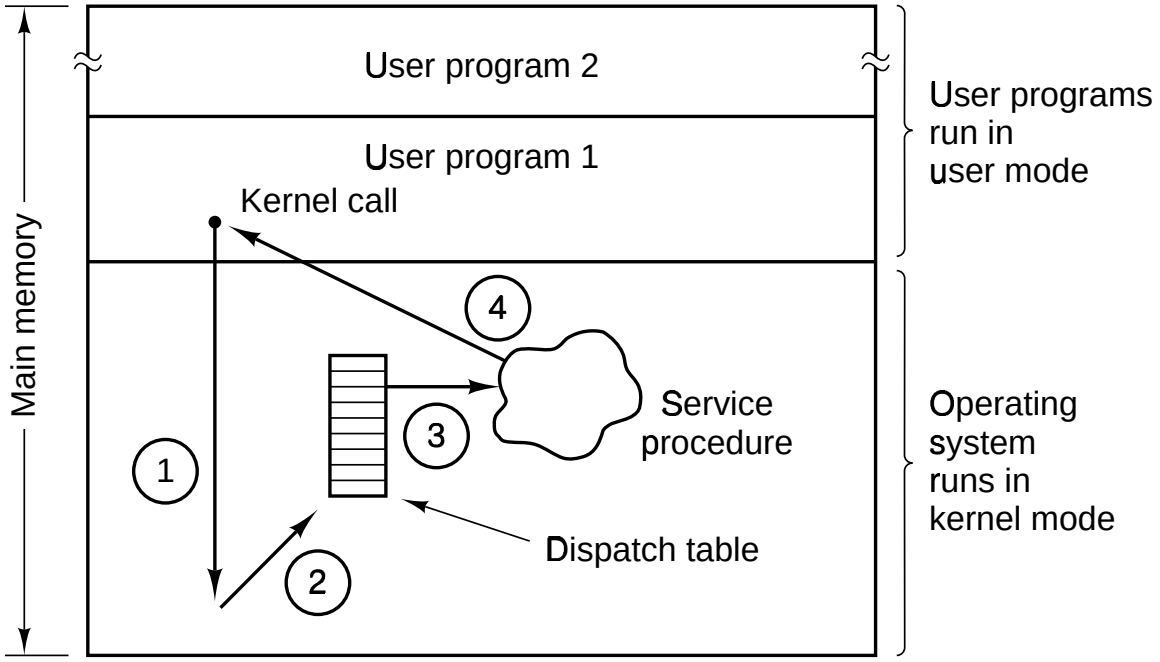

Figure 1-16. How a system call can be made: (1) User program traps to the kernel. (2) Operating system determines service number required. (3) Operating system calls service procedure. (4) Control is returned to user program.