System Programming Lab

2021315385

이건 / Gun Daniel Lee

Report 5

The commands *fork(), exit(), wait(),* and *execv()* are essential commands in controlling the process flow. This report will summarize the Linux manual of these commands given using the *'man'* command.

fork() – The fork command creates a child process. It duplicates the calling process and refers to this new process as the child process. The calling process is then referred to as the parent process. Although duplicated, both processes are run in separate memory spaces. At the moment of fork(), both memory spaces have the same content. However, specified data and information such as the process ID, memory locks, CPU time counters, etc. are different.

exit() – The exit command causes normal process termination. The value of the status, which is the parameter of the exit command, is returned to the parent. All the functions registered with *atexist()* and *on_exit()* are called in the reverse order of their registration. During the exiting process, if a function does not return, then none of the remaining functions will be called or exited. The exit command will flush and close all open stdio streams and remove all created tmp-files. The C standard specifies two constants, EXIT_SUCCESS and EXIT_FAILURE, which indicate a successful or unsuccessful termination, respectively, when passed to exit().

wait() – The wait command is used to wait for a certain process to change state. It waits for state changes occurring in a child of a calling process, and obtains information about this child. Certain state changes include a child being terminated, stopped by a single, and resumed by a signal. Performing a wait will allow the system to release resources associated with the child. If it is not performed, the terminated child remains in a "zombie" state.

execv() – The execv command executes a file. The entire exec() family of functions replaces the current process image with a new process image. For the execv() command, it is provided with an array of pointers to null-terminated strings that represent the argument list available to the new program. The first argument, by convention, should point to the filename associated with the file being executed. The array of pointers must end with a null pointer.

In conclusion, the following commands are great in manipulating process within a certain program. It will be of great use in future coding exercises.