Unix Fork

The Unix “Fork” is a command in the UNIX-based operating system that creates a new process by duplicating an existing one. The created process is named the child process while the copied process becomes the parent process. Both of these have their own distinctive IDs so they can be distinguished one from the other. After calling the fork command, both processes execute independently and have their own memory space dedicated to each. The fork command calls on the operating system to create a copy of the entire process, including all of its stack, heap and global variables. Along with that it also inherits all of the parent process's resources, such as open files and signal handlers. However, as they operate within their own memory spaces, the child process can make changes to those resources without affecting the parent’s process resources. As an example, if the child process inherits a file, the child can open and close the file without the parent process’s file being affected.

Both child and parent processes starts executing at the same point as the partn process. After calling fork, the parent process will continue in the running state while the child process will be placed into the new state. Once the child process is in the ready state the child process will begin executing from the point where “fork” was called but, as stated previously, it will be running in a different memory location.

The Unix fork command has many use cases and it is one of the most essential function in the Unix system. It is used by a shell in order to invoke programs called form the command line. Web servers utilize it to create multiple server processes, each of which handles requests in its own address space. Google Chrome utilizes it to prevent client-side code on one page form bringing your whole browser down. Another use of this is to create concurrent and parallel programs that can take full advantage of the hardware available to it.

Overall, the Unix fork command is a critical aspect of network programming as it allows multiple processes to execute simultaneously, improving performance and efficiency. The ability to create child processes is essential in order for a computer and an operating system to take advantage of all of the available hardware resources it has in order to fulfill tasks.

Resources:

https://stackoverflow.com/questions/985051/what-is-the-purpose-of-fork

https://en.wikipedia.org/wiki/Fork_(system_call)

https://www.geeksforgeeks.org/fork-system-call/

https://www.csl.mtu.edu/cs4411.ck/www/NOTES/process/fork/create.html