Fork function

Aishi De

The fork() function in C is a system call used in Unix/Linux to create a new process by duplicating the current process. When fork() is called, it creates a child process that is an exact copy of the parent process, including the same code, variables, and open file descriptors. After the fork, both the parent and child processes continue execution independently from the point where fork() was called. The function returns a value that helps distinguish between the two: it returns 0 to the child process, the child's process ID (PID) to the parent, and -1 if the fork fails. This mechanism is fundamental for process creation and management in operating systems and is often used to demonstrate parent-child process interaction.

```
aishi@Aishi:~$ vi fork_example.c
aishi@Aishi:~$ touch output
aishi@Aishi:~$ cat fork_example.c

#include <stdio.h>
#include <unistd.h>

int main() {
    pid_t pid;

    pid = fork(); // create a child process

    if (pid < 0) {
        // Fork failed
        printf("Fork failed.\n");
        return 1;
    } else if (pid == 0) {
        // Child process
        printf("Hello from the Child process! PID: %d\n", getpid());
    } else {
        // Parent process
        printf("Hello from the Parent process! PID: %d, Child PID: %d\n", getpid(), pid);
    }

    return 0;
}

aishi@Aishi:~$ gcc fork_example.c -o output
aishi@Aishi:~$ ./output
Hello from the Parent process! PID: 3526, Child PID: 3527
Hello from the Child process! PID: 3527
aishi@Aishi:~$
```