execve()

The `execve()` (Execute Vector Environment) function replaces the current process with a new program, loading it into memory and executing it from the entry point.

It’s defined in the “unistd.h” system header.

```
#include <unistd.h>

// "pathname" is the full path to the executable
// "argv" is a NULL-terminated array of command-line arguments
// "envp" is a NULL-terminated array of environment variables
int execve(const char *pathname, char *const argv[], char *const envp[]);
```

On success, does not return because the current process is replaced.

In case of failure, returns -1 and sets “errno”.

- Overwrites the current process’s code, data, heap, and stack. The PID remains the same (no new process is created)
- “argv[0]” is the program name by convention. Example for “ls -l /tmp”:
    - char *argv[] = {”ls”, “-l”, “/tmp”, NULL}
- “envp” is an array of strings like “PATH=/bin”. If “envp” is NULL, the new program inherits the current environment

Common “errno” values:

- “EACCES” → No execute permission
- “ENOENT” → File not found
- “ENOEXEC” → Not a valid executable

Example Usage

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

int main(void)
{
    char *argv[] = {"ls", "-l", "/tmp", NULL};
    char *envp[] = {"PATH=/bin", "USER=root", NULL};

    execve("/bin/ls", argv, envp);
    // if we get here, execve() failed
    perror("execve");
    return (1);
}
```