Execute commands, exec system call family and parser basics

VE482 pre-group 11

September 29, 2021

exec system call family

exec system call family

- executing exec system calls will start a new process on the original process.
- exec system calls with "l", i.e., execl, execle and execlp: takes VA_ARGS as argument, which should be ended with a NULL; exec system calls with "v": takes an array as argument, which should also be ended with a NULL.
- exec system calls with "p": include $PATH in environment, i.e., could directly access the program in stored under $PATH, including all the basic shell commands.
- exec system calls with "e": inject custom environment variables (if we need to implement export command, this function is needed).
- exec system call family is a wrapping of execve system call.

exec and fork

- exec system calls will replace the original process if the new process is successfully launched.
- fork could create a subprocess for exec to replace in execution of shell commands (discussed in topic 2).
- exec will not change the pid of the process.
- exec will not return if it has launched the process successfully.

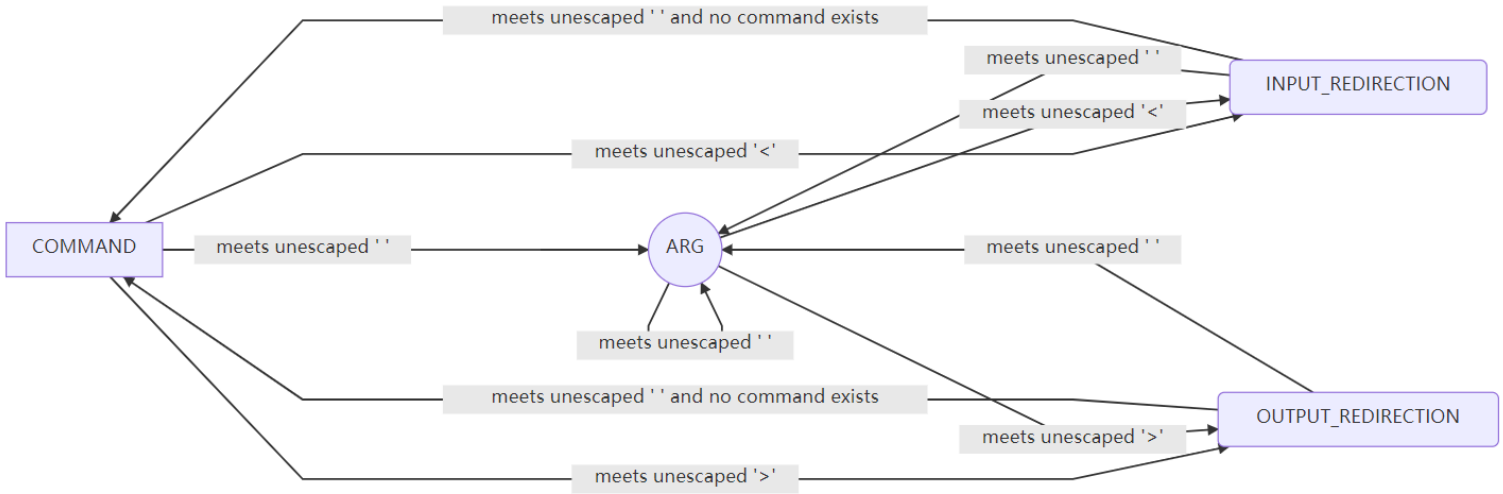
exec and errno

- if `exec` could not launch the process, it will return a `-1`.
- `exec` will set `errno` to record error message.

Parser

Basic idea: state machine

- a strong parser could be implemented by state machine.



How to store parsed command

- notice that the data structure should include “command”, “args”, “redirection” and pipeline execution.
- a single linked list could represent chain pipelined commands.

```
struct command {
    char *command;
    char **args;
    char *input_redirection;
    char *output_redirection;
    struct command* next;
}
```

Thanks for your atthention!