Unix System Programming

Process Control

Chandravva Hebbi
Department of Computer Science and Engineering
chandravvahebbi@pes.edu

Unix System Programming

Process Control

Chandravva Hebbi

Department of Computer Science and Engineering

Topics to be Covered

❖fork(),exec()

❖wait() and waitpid()

❖Programming examples

vfork Functions

- The vfork function is intended to create a new process when the purpose of the new process is to exec a new program.
- Does not copy the address space into the child
- Fork followed by exec
- The child executes in the address space of the parent
- vfork guarantees that the child runs first, parent waits until the child calls exec or exit.

exec Functions

- When a process calls one of the exec functions, that process is completely replaced by the new program.
- The new program starts executing at its main function.
- The process ID does not change across an exec, because a new process is not created.
- exec merely replaces the current process its text, data, heap, and stack segments with a brand new program from disk.

UNIX SYSTEM PROGRAMMING

exec Functions

```
#include <unistd.h>

int execl(const char *pathname, const char *arg0, ... /* (char *)0 */ );

int execv(const char *pathname, char *const argv []);

int execle(const char *pathname, const char *arg0, .../* (char *)0, char

*const envp[] */ );

int execve(const char *pathname, char *const argv[], char *const envp []);

int execlp(const char *filename, const char *arg0, ... /* (char *)0 */ );

int execvp(const char *filename, char *const argv []);
```

All six return: -1 on error, no return on success

UNIX SYSTEM PROGRAMMING

Exec functions

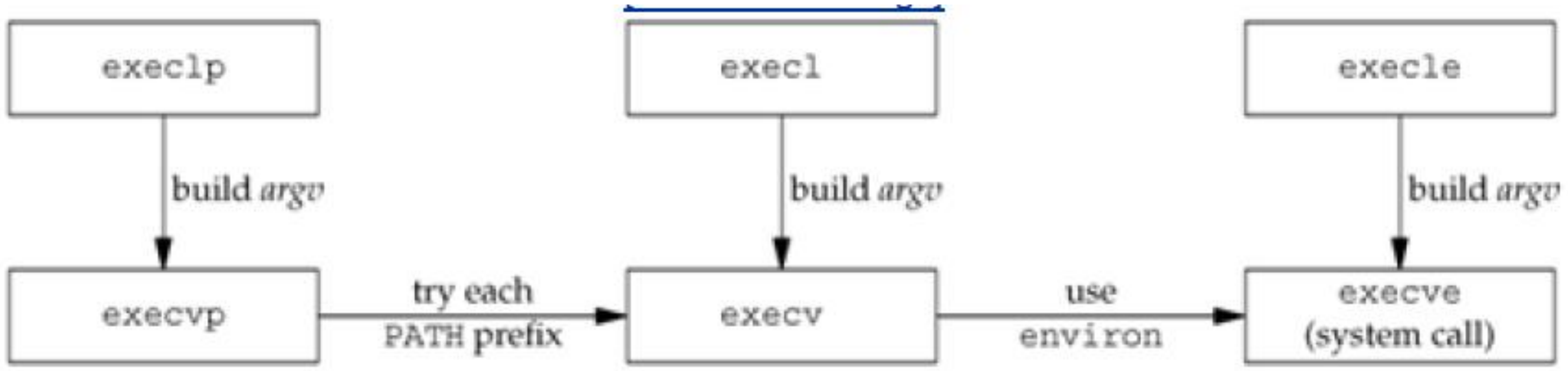
char **pathname*, char **arg0*, ..., char **argn*, (char *)0, char **envp*[]

Difference Among Exec functions

Function	*pathname*	*filename*	Arg list	*argv*[]	environ	*envp*[]
execl	●		●		●	
execlp		●	●		●	
execle	●		●			●
execv	●			●	●	
execvp		●		●	●	
execve	●			●		●
(letter in name)		p	l	v		e

```
PATH=/bin:/usr/bin:/usr/local/bin/:.
```

Relationship of the exec functions



only one of these six functions, execve is a system call within the kernel.

UNIX SYSTEM PROGRAMMING

exit Functions

- Executing a return from the main function.
- Calling the exit function.
- Calling the _exit or _Exit function.
- Executing a return from the start routine of the last thread in the process.
- Calling the pthread_exit function from the last thread in the process.
- Calling abort.
- When the process receives certain signals.
- The last thread responds to a cancellation request.

exit Functions

- Executing a return from the main function.
- Calling the exit function.
- Calling the _exit or _Exit function.
- Executing a return from the start routine of the last thread in the process.
- Calling the pthread_exit function from the last thread in the process.
- Calling abort.
- When the process receives certain signals.
- The last thread responds to a cancellation request.

UNIX SYSTEM PROGRAMMING

exit Functions

- The exit status is converted into a termination status by the kernel when _exit is finally called.
- If the child terminated normally, the parent can obtain the exit status of the child.

Macros to examine the termination status returned by wait and waitpid

exit Functions

- WIFEXITED(*status*)

True if status was returned for a child that terminated normally.

WEXITSTATUS (*status*) is executed to fetch the low-order 8 bits of the argument that the child passed to exit, _exit,or _Exit.

- WIFSIGNALED (*status*) True if status was returned for a child that terminated abnormally, by receipt of a signal that it didn't catch.

WTERMSIG (*status*) is executed to fetch the signal number that caused the termination.

WCOREDUMP (*status*) that returns true if a core file of the terminated process was generated

exit Functions

- WIFSTOPPED (*status*) True if status was returned for a child that is currently stopped

  WSTOPSIG (*status*) is executed to to fetch the signal number that caused the child to stop.

- WIFCONTINUED (*status*) True if status was returned for a child that has been continued after a job control stop.

wait and waitpid Functions

When a process terminates, either normally or abnormally, the kernel notifies the parent by sending the SIGCHLD signal to the parent.

```
#include <sys/wait.h>
pid_t wait(int *statloc);
pid_t waitpid(pid_t pid, int *statloc, int options);
```

Both return: process ID if OK, 0 (see later), or -1 on error

wait and waitpid Functions

The interpretation of the *pid* argument for waitpid depends on its value:

pid == 1 Waits for any child process. In this respect, waitpid is equivalent to wait.

pid > 0 Waits for the child whose process ID equals *pid*.

pid == 0 Waits for any child whose process group ID equals that of the calling process.

pid < 1 Waits for any child whose process group ID equals the absolute value of *pid*.

The waitpid function returns the process ID of the child that terminated and stores the child’s termination status in the memory location pointed to by *statloc*.

wait and waitpid Functions

The *options* constants for waitpid

WCONTINUED: If the implementation supports job control, the status of any child specified by *pid* that has been continued after being stopped, but whose status has not yet been reported, is returned.

WNOHANG: The waitpid function will not block if a child specified by *pid* is not immediately available. In this case, the return value is 0.

WUNTRACED: If the implementation supports job control, the status of any child specified by *pid* that has stopped, and whose status has not been reported since it has stopped, is returned

wait and waitpid Functions

The waitpid function provides three features that aren't provided by the wait function.

1. The waitpid function lets us wait for one particular process, whereas the wait function returns the status of any terminated child. We'll return to this feature when we discuss the popen function.

2. The waitpid function provides a nonblocking version of wait. There are times when we want to fetch a child's status, but we don't want to block.

3. The waitpid function provides support for job control with the WUNTRACED and WCONTINUED options

THANK YOU

Chandravva Hebbi
Department of Computer Science and Engineering

chandravvahebbi@pes.edu