Процеси; Системни извиквания за работа с процеси

Христо Иванов[1]
hivanov@elsys-bg.org

[1]Технологично училище "Електронни системи"
Технически университет, София

18 ноември 2020 г.

Съдържание

Системно извикване fork()

Описание

```
#include <sys/types.h>
#include <unistd.h>
pid_t fork(void);
```

- Системното извикване **fork()** създава нов процес като дублира процеса, който го е извикал. Новият процес е процес-дете на процеса извикал **fork()**.
- Системното извикване **fork()** няма аргументи и връща стойност от тип pid_t, който е целочислен. PID е process ID (идентификатор на процеса).
- При успешно извикване **fork()** връща 0 към процеса-дете, PID на процеса-дете към процеса-родител или -1 при грешка. Номерът на грешката се записва в **errno**.

Системно извикване fork()

Особености

- При създаването на процеса-дете той получава копия на следните атрибути на процеса-родител:
  - Данни и стек –> всички променливи декларирани преди **fork()**.
  - Таблица на отворените файлове, текущата директория.
- Двата процеса, родител и дете, продължават изпълнението си от инструкцията след **fork()**.
- Двата процеса, родител и дете, имат различни идентификатори на процеса.

Системни извиквания execl() и execv()

Общо описание

```
#include <unistd.h>
int execl(const char *path, const char *arg, ...);
int execv(const char *path, char *const argv[]);
```

- Системните извиквания **execl()** и **execv()** заменят текущо изпълняваната програма с друга, указана чрез аргументите.
- Това не предизвиква създаване на нов процес, само се преминава към изпълнението на другата програма(изпълним двоичен файл).
- Връщането в оригиналната програма не е възможно, освен ако извикването на **execl()** и **execv()** не пропадне.
- Системните извиквания **execl()** и **execv()** връщат стойност само при наличие на грешка. Върнатата стойност е -1 и номерът на грешката се записва в **errno**.

Системни извиквания execl() и execv()

Описание на аргументите

```
#include <unistd.h>
int execl(const char *path, const char *arg, ...);
int execv(const char *path, char *const argv[]);
```

- Системните извиквания **execl()** и **execv()** приемат като цяло едни и същи аргументи, но групирани по различен начин.
- Типът данни на тези аргументи е null terminated string.
- Първият аргумент и на двете извиквания, *path*, указва пътя към изпълнимия файл, който искаме да стартираме.

Системни извиквания execl() и execv()

Описание на аргументите на execl()

```
Пример: execl("/bin/cat", "cat", "-A", "test.txt", NULL);
```

- След пътя към изпълнимия файл, **execl()** приема списък от аргументи нужни за извикването му.
- Първият е името на файла, след което следват останалите нужни аргументи и накрая NULL, с което с указва че няма повече аргументи.
- Списъкът с аргументи след пътя може да се приеме като "раздробена версия"на параметъра *argv* на функцията **main()**.
- Обикновено **execl()** се използва, когато броят на параметрите е известен.

Системни извиквания execl() и execv()

Описание на аргументите на execv()

Пример:

```
char* myargv[] = {"mycat", "-A", "test.txt", (char*)NULL};
execv("/home/haxor/mycat", myargv);
```

- След пътя към изпълнимия файл, **execv()** приема вектор(масив) от указатели към нулево-терминирани стрингове, подобен на *argv*.
- Първият е името на файла, след което следват останалите нужни аргументи и накрая указател сочещ към NULL, с което с указва че няма повече аргументи.
- Параметърът *argv* на функцията **main()** може да се предава директно на **execv()**.
- Обикновено **execv()** се използва, когато броят на параметрите е неизвестен.

Системно извикване waitpid()

Общо описание

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

- Системното извикване **waitpid()** блокира процеса, който го е извикал, докато не приключи някое от неговите процеси-деца.
- Параметърът *pid* съдържа идентификатора на процеса, чието приключване се очаква. Ако се подаде -1, то се чака приключването на кой да е процес-дете, вместо конкретен.
- Параметърът *wstatus* е указател към променлива, в която ще се запише код на състоянието на процеса-дете.
- Параметърът *options* съдържа различни настройки за системното извикване, но може просто да се подаде 0.

Системни извиквания getpid() и getppid()

Общо описание

```
#include <sys/types.h>
#include <unistd.h>
pid_t getpid(void);
pid_t getppid(void);
```

- Системното извикване **getpid()** връща идентификатора на извикващия го процес.
- Системното извикване **getppid()** връща идентификатора на родителя на извикващия го процес.
- И двете системни извиквания нямат параметри и връщат стойност с тип *pid_t*.