Inter-process communication in Linux

spawn()

La funzione **spawn()** in C crea un nuovo processo figlio, eseguito in background. Ha la seguente sintassi:

```
spawn(programma, arg);
```

dove:

- *programma* è il path del programma (eseguibile) da lanciare;
- *arg* è un puntatore ad un array di stringhe che contiene gli argomenti da passare al processo figlio. Il primo elemento dell'array deve essere il nome stesso del programma.

A cosa serve?

- Creazione di un nuovo processo: Viene creato un nuovo processo che esegue il programma specificato. Il processo padre prosegue la sua esecuzione in parallelo.
- Esecuzione in background: Il nuovo processo viene eseguito in background, senza bloccare il processo padre. I due processi sono completamente indipendenti.
- Condivisione delle risorse: I due processi condividono le stesse risorse del sistema (file, variabili globali, heap, stack, etc.), essendo parte dello stesso spazio di indirizzamento.
- Comunicazione inter-processo: Per comunicare, i processi devono utilizzare meccanismi IPC (Inter-Process Communication) come pipe(), socket(), messaggi, etc.
- PID del figlio: La funzione restituisce il PID (Process ID) del nuovo processo creato. Il padre può utilizzarlo per monitorarlo, terminarlo, etc.
- Gestione degli errori: Se la funzione non riesce a creare il nuovo processo, restituisce -1. In questo caso, il valore di ritorno di errno indica l'errore specifico.

Limitazioni

La funzione spawn() è limitata ai sistemi operativi Unix-like (Unix, Linux, etc). Non è presente su Windows.

O_WRONLY

O_WRONLY è una costante definita nelle header file sys/types.h e fcntl.h, utilizzata nelle chiamate di sistema per l'apertura di file (*open()*).
Si tratta di un flag che specifica il modo in cui verrà aperto il file: O_WRONLY indica che il file verrà aperto in sola scrittura. Questo significa che una volta aperto il file:

- Si potranno scrivere dati sul file, utilizzando le funzioni write(), printf(), etc.

- Non sarà possibile leggere dati dal file. Ogni tentativo di lettura comporterà un errore.
- Ogni nuova scrittura sul file sovrascrive i dati precedentemente salvati.
- Il file verrà troncato. Se il file esisteva precedentemente, verrà eliminato il contenuto azzerando la dimensione del file.

La sintassi per aprire un file in sola scrittura è:

```
fd = open("nomefile", O_WRONLY | O_CREAT | O_TRUNC, 0666);
```

dove:
- "nomefile" è il percorso e nome del file da aprire;
- O_WRONLY specifica l'apertura in sola scrittura;
- O_CREAT e O_TRUNC sono altri flag che specificano la creazione del file (se non esiste) e la troncatura;
- 0666 sono i permessi di accesso al file (lettura, scrittura, esecuzione per l'utente, il gruppo e gli altri utenti).

Altre modalità possibili sono:
- O_APPEND per concatenare,
- O_RDWR per lettura e scrittura.

Le altre modalità di apertura (lettura, lettura/scrittura, etc.) hanno flag e comportamenti analoghi.

A cosa serve?

Lo scopo principale dell'uso di O_WRONLY è evitare l'apertura accidentale di un file in modalità di lettura, con conseguente rallentamento delle operazioni di scrittura.

Alcuni esempi

➔ fd = open("file.txt", O_CREAT | O_TRUNC | O_WRONLY);
Questo apre/crea "file.txt" in sola scrittura e lo tronca immediatamente.

➔ fd = open("file.txt", O_WRONLY);
Questo apre "file.txt" in sola scrittura. Ogni successiva chiamata a write() sovrascriverà e troncherà parte del file.

➔ fd = open("file.txt", O_APPEND | O_WRONLY);
Questo apre "file.txt" in modalità di append (aggiunta), pertanto non verrà troncato. Le nuove scritture verranno sempre aggiunte alla fine del file.