execl

venerdì 31 marzo 2023 12:53

execl

int execl(char *file_name, [char *arg0, ... ,char *argN,] 0)

Descrizione invoca l'esecuzione di un programma

Parametri 1) *file_name: nome del programma
2) [*arg0, ... ,*argN,] sono i parametri della funzione main()

Restituzione -1 in caso di fallimento

Quando chiamiamo una execl, chiamiamo il kernel per cercare di lanciare un programma associato al nome che è il primo parametro della funzione, ma il programma passato, a che punto del file system viene ad essere considerato?
Nel mio file system posso avere più programmi con lo stesso nome in cartelle differenti, ma ci sono. Supponiamo ho due programmi che si chiamano X e dico al kernel di sostituire il mio address space con un programma X, quale dei due è quello che voglio sostituire?
QUINDI QUANDO L'API RICEVE IL NOME PER CAPIRE QUALE RAMIFICAZIONE SCEGLIERE IN UN FILE SYSTEM, SI CERCA DI LANCIARE IL PROGRAMMA CON IL NOME PASSATO CHE EVENTUALMENTE È PRESENTE NELLA DIRECTORY CORRENTE.

- l'esecuzione avviene per sostituzione del codice (valido per tutte le chiamate della famiglia)
- 'cerca' il programma da eseguire solo nel direttorio corrente

Per capire il direttorio corrente questa execl, quando prende il controllo, va nelle variabili d'ambiente e va nella variabile che indica qual'è il direttorio corrente di lavoro dell'applicazione che sta chiamando la exec, prende quel direttorio, associa il nome del programma a quel direttorio e genera il percorso che deve essere indicato al kernel per il set-up.

Una variante è execlp(), in cui quando noi andiamo a cercare di lanciare un programma, essa prende una stringa che identifica il programma da lanciare, però per capire qual'è il punto dell'archivio dove eventualmente il programma dovrà essere lanciato viene utilizzata un'altra variabile d'ambiente che si chiama "Path". Mentre invece nella Execl() il direttorio corrente è Process Working Directory (PWD).

PWD è la directory di lavoro corrente dell'applicazione attiva.

```
OCESSES-AND-THREADS/UNIX> env
```

Questo è un comando per visualizzare tutte le variabili d'ambiente.
ENV!

```
XSESSION_IS_UP=yes
LOGNAME=francesco
DBUS_SESSION_BUS_ADDRESS=unix:path=/run/user/1000/bus
XDG_RUNTIME_DIR=/run/user/1000
XAUTHORITY=/home/francesco/.local/share/sddm/.Xauthority
INPUT_METHOD=ibus
JRE_HOME=/usr/lib64/jvm/java-10-openjdk-10/jre
XDG_SESSION_PATH=/org/freedesktop/DisplayManager/Session1
QT_AUTO_SCREEN_SCALE_FACTOR=0
XDG_CONFIG_DIRS=/etc/xdg
PATH=/home/francesco/bin:/usr/local/bin:/usr/bin:/bin:/usr/lib/mit/sbin
JAVA_BINDIR=/usr/lib64/jvm/jre/bin
KDE_SESSION_UID=1000
SDL_AUDIODRIVER=pulse
KDE_SESSION_VERSION=5
QT_IM_SWITCHER=imsw-multi
G_BROKEN_FILENAMES=1
HISTSIZE=1000
SESSION_MANAGER=local/linux-mxb5:@/tmp/.ICE-unix/1984,unix/linux-mxb5:/tmp/.ICE-unix/1984
CPU=x86_64
CVS_RSH=ssh
LESSOPEN=lessopen.sh %s
GTK_IM_MODULE=ibus
_=/usr/bin/env
OLDPWD=/home/francesco/git-web-site/FrancescoQuaglia.github.io/TEACHING/SISTEMI-OPERATIVI/CURRENT/SOFTWARE-EXAMPLES/C-
BASICS
```

```
PATH=/home/francesco/bin:/usr/local/bin:/usr/bin:/bin:/usr/lib/mit/sbin
PWD=/home/francesco/git-web-site/FrancescoQuaglia.github.io/TEACHING/SISTEMI-OPERATIVI/CURRENT/SOFTWARE-EXAMPLES/PROCE
SSES-AND-THREADS/UNIX
```

In execlp() se noi vogliamo passare a.out come nome del programma da eseguire, essa interpreta la variabile d'ambiente PATH e non PWD, e va a lanciare il programma se presente nei path di quella variabile.

- la funzione `execlp()` cerca in tutto il path valido per l'applicazione che la invoca, secondo uno schema 'fail-retry'

secondo uno schema "fail retry"

Mantenuto nella variabile d'ambiente PATH

Questo è il senso per il quale noi andiamo delle volte a lavorare con dei path e con dei file name.
Il file_name è la PWD

Poi nella execl abbiamo da specificare tutta una serie di parametri fino alla stringa nulla. Queste sono le stringhe che vogliamo che vengano passate al main dell'applicazione che va a sostituire l'applicazione corrente.

Quindi stiamo chiedendo di lanciare quell'applicazione e stiamo anche passando i parametri per il main.
Quando un kernel lancia un'applicazione tipicamente può scrivere all'interno dell'address space le stringhe per il main, ma anche le stringhe ambientali, ma qui non le possiamo specificare le stringhe ambientali.
Il che implica dire che quando questa API prende il controllo e cerca di fare delle attività e prima o poi prenderà il controllo verso un API vera di sistema che è EXECVE, per le stringhe ambientali viene fatto un lavoro di DEFAULT: execve va ad indicare quali sono le stringhe ambientali da inserire sul programma che deve essere lanciato, e queste vengono ad essere acquisite da EXECL secondo uno schema di DEFAULT.
Prima non specifichiamo le stringhe ambientali nella execl, ma poi è esattamente quest'ultima che chiama execve, dove gli passa sicuramente I parametri del main ma anche i parametri delle stringhe ambientali.
Così poi quando verrà lanciata la nuova applicazione troveremo tutte le informazioni che ci servono nel nuovo address space. ☆