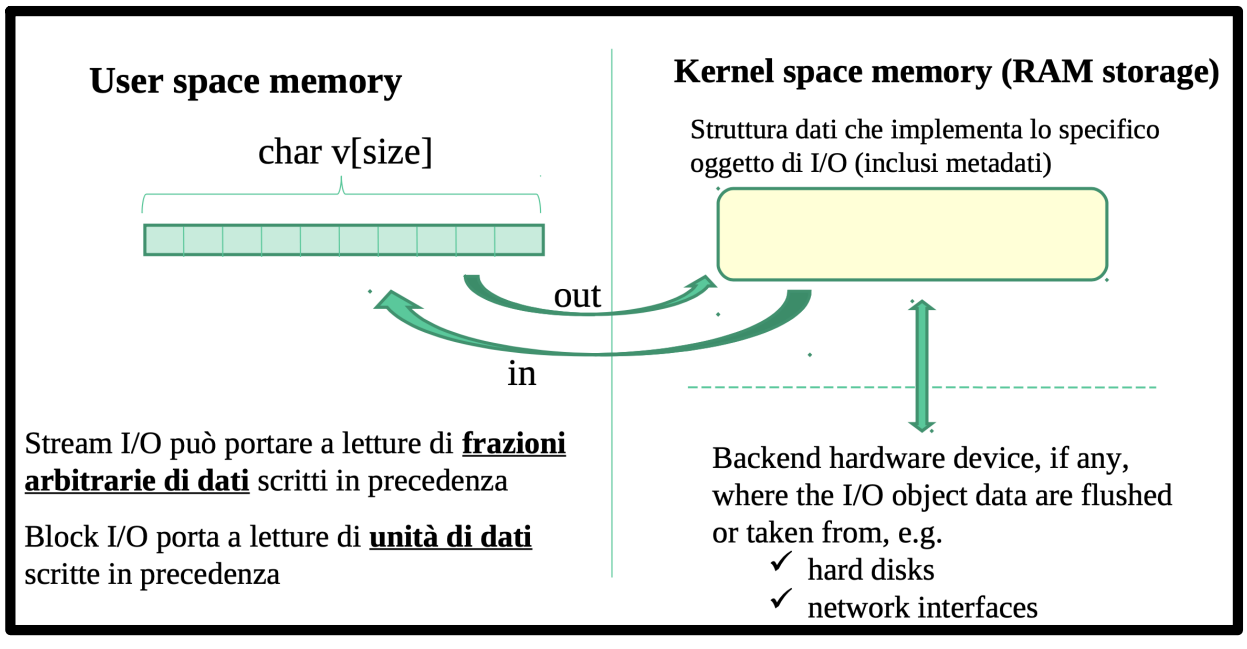
Stream/block I/O model

venerdì 14 aprile 2023 12:31

Stream/block I/O model

Questi sono gli unici due modi che noi abbiamo all'interno di un sistema operativo moderno per andare ad eseguire delle operazioni di I/O su degli oggetti. A seconda del tipo di oggetto di I/O su cui andiamo a lavorare utilizzeremo l'uno o l'altro dei modelli.
Ora vediamo come un'operazione di I/O viene realmente ad essere eseguita all'interno di un sistema operativo moderno e poi cerchiamo di capire quali sono le peculiarità di un'operazione "stream" rispetto ad un'operazione "block".



Tipicamente per eseguire un'operazione di I/O da parte di un'applicazione che è attiva all'interno di un sistema operativo, noi dobbiamo avere all'interno dello USER-SPACE, ossia lo spazio di memoria che l'applicazione sta utilizzando, un'array di caratteri (area di memoria) che possiamo utilizzare esattamente come "buffer" per servire l'operazione di I/O.

↓

Se l'operazione di I/O è un'operazione di output, quindi dobbiamo produrre dei dati che devono essere emessi verso un oggetto di I/O, i dati li scriveremo prima all'interno dell'area verde in figura, e questi dati sono rappresentati esattamente come una sequenza di byte. Una volta scritti all'interno di questa area, noi possiamo chiamare un'operazione di OUT per andare ad attivare ovviamente un servizio del kernel che possa andare nel nostro address space a prelevare questi dati e copiarli verso una struttura dati, che tipicamente è gestita a livello sistema, che implementa lo specifico oggetto di I/O su cui stiamo lavorando.

↓

Ovviamente la struttura dati che rappresenta questo oggetto di I/O CONTIENE ANCHE i metadati: magari i permessi d'accesso.

↓

Quando noi andiamo a cercare una lettura di informazioni da un oggetto di I/O, le informazioni da qualche parte devono essere prese per essere copiate e rese a noi disponibili all'interno della USER SPACE MEMORY. I dati per essere copiati in questa area di memoria, inizialmente, sono presenti dento la Kernel Space Memory, ma potrebbero a loro volta essere stati presi da un hard drive o da una scheda di rete.

↓

Se noi abbiamo uno STREAM I/O possiamo eseguire letture di frazioni di dati che sono stati scritti sull'oggetto di I/O in precedenza. Questo ci permette di leggere le informazioni che sono presenti nell'oggetto di I/O nella maniera più disparata possibile: magari nell'operazione di OUT avevamo scritto N byte e magari decidiamo in una lettura di prelevare M byte, dove M<N. Gli altri M-N byte li possiamo leggere successivamente.

↓

Se noi abbiamo un BLOCK I/O gli N byte rappresentano un unico contenuto informativo. Quando verranno lette informazioni dalla kernel area, leggeremo tutto il blocco di N byte.

↓

Non possiamo leggere questi N byte in maniera frazionata.

↓

Stream I/O può portare a letture di **frazioni arbitrarie di dati** scritti in precedenza

Block I/O porta a letture di **unità di dati** scritte in precedenza