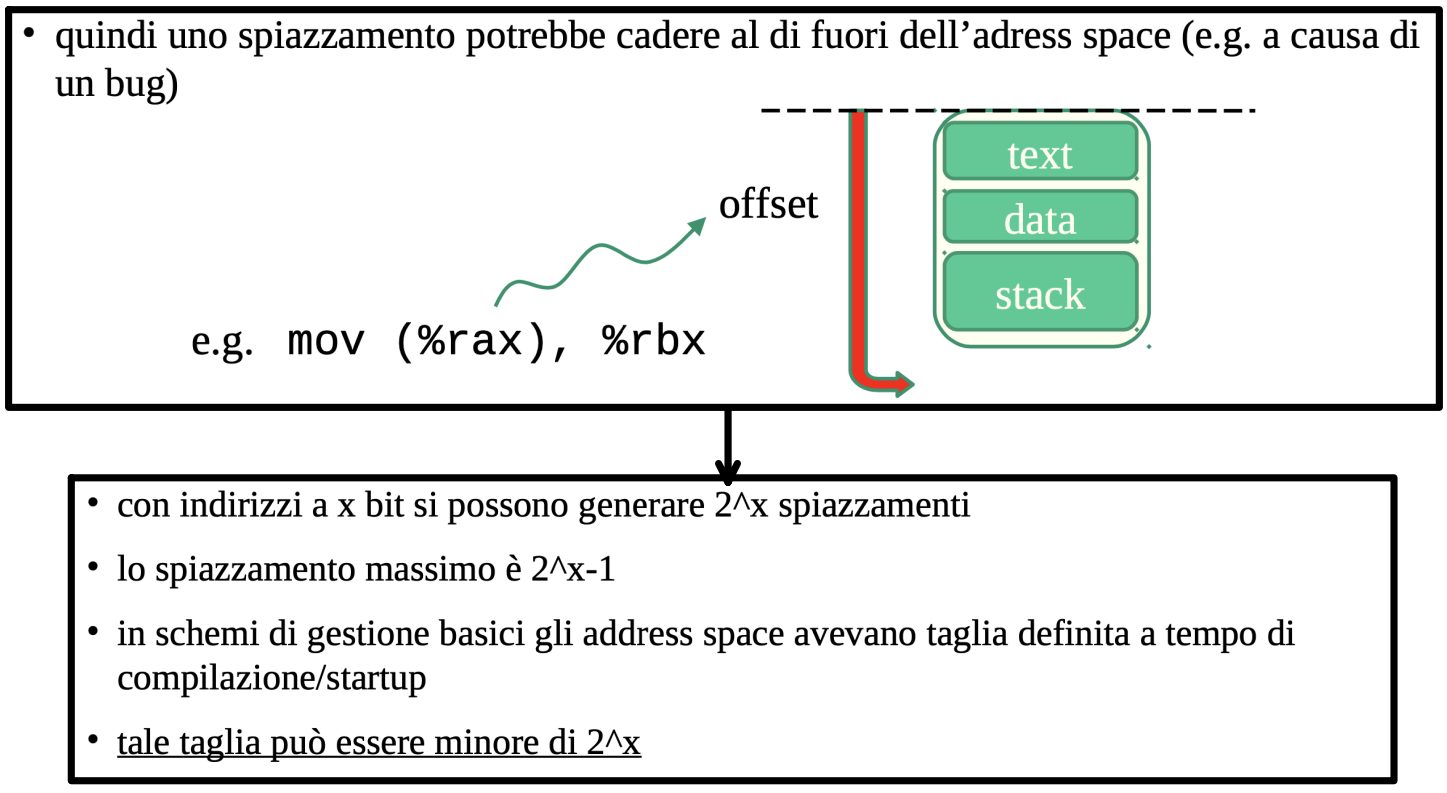
Taglie degli indirizzi e taglie degli address space

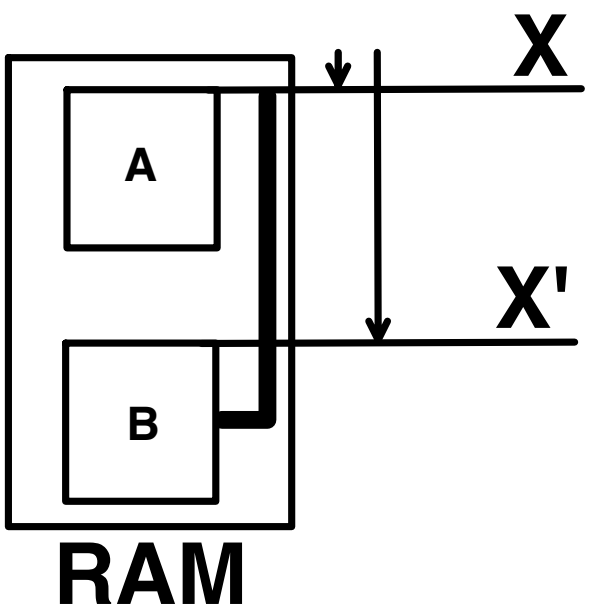
mercoledì 10 maggio 2023 16:18

Taglie degli indirizzi e taglie degli address space

Vediamo come si può lavorare con address space di taglia differente. Potevamo avere applicazioni con un address space piccolo, e applicazioni molto più grandi.
Come potevamo gestire questa situazione?
La taglia massima di un address space che possiamo gestire dipende dal numero dei bit che il processore utilizza per esprimere un indirizzo LOGICO (che è uno spiazzamento a partiredall'address space). Se abbiamo N bit per esprimere un indirizzo logico, abbiamo 2^N byte possibili per la taglia massima dell'address space.
Non è assolutamente detto che l'applicazione ha necessità di utilizzarli tutti, anche perché se così fosse, avremo la necessità anche di dover posizionare quantità di informazioni ampie caratterizzanti un address space all'interno della memoria fisica quando questo non serve. Quindi se l'applicazione è più piccola, di questi 2^N, ne utilizziamo un sottoinsieme.
Questo induce la necessità di avere una MMU anche un po' più articolata per la gestione di taglie differenti degli address space, nel momento in cui - per quanto riguarda gli indirizzi logici - possiamo avere una certa taglia dell'indirizzo logico che ci permette di avere lo spiazzamento massimo pari a 2^x-1 a partire da 0, quindi 2^x possibilità se abbiamo X bit possibili per esprimere un indirizzo.
Quindi la taglia di un address space può essere minore di 2^x possibili indirizzi, e quindi uno spiazzamento potrebbe cadere al di fuori dell'address space:



Voglio muovere ciò che in memoria è mantenuto all'indirizzo registrato in RAX, all'interno del registro RBX. RAX può ospitare questi X bit, quindi con x BIT POSSIAMO esprimere questo indirizzo. Idealmente con questi X bit potremmo esprimere un offset a partire dall'inizio del contenitore che ci può portare anche fuori. Posso anche colpire altre applicazioni che sono all'interno della RAM. Quindi bisogna proteggere gli accessi. Perché si possono creare interferenze di altri processi.



Mentre eseguo A esprimo un riferimento che è fuori dalla sua zona.

Questa è un'interferenza inaccettabile nel momento in cui noi vogliamo avere una reale concorrenza, quindi mantenere più applicazioni attive in modo tale che comunque la gestione che il S.O fa di queste applicazioni garantisce che queste siano comunque isolate tra di loro.
Se noi andiamo fuori dal contenitore e per accedere alla memoria fisica abbiamo semplicemente sommato la base di dove è presente questa applicazione per determinare la posizione in RAM di questa applicazione, stiamo uscendo dalla zona riservata per questa applicazione.
Vado a riferire informazioni che sono all'interno della RAM che caratterizzano lo stato di un'altra applicazione.