

(2) אך במקרה Trap או Exception מתרחש בmodeusu (user mode) ובמקרה זה exception יתבצע על ידי syscall ו-Trap תחילה (user mode) ו-exception יתבצע לאחר מכן (kernel mode). Trap יתבצע על ידי מעבדה הולכת (kernel handler) ש-Trap יתבצע בmode kernel (kernel mode). במקרה הזה יתבצע exception על ידי מעבדה הולכת (kernel handler) ש-Trap יתבצע בmode kernel (kernel mode). Trap יתבצע בmode kernel (kernel mode) ו-exception יתבצע על ידי מעבדה הולכת (kernel handler) ש-Trap יתבצע בmode kernel (kernel mode).

(3) במקרה קוריאט פונקציית write ל.library (library) כמפורט לעיל.

(4) במקרה של write syscall ב-Trap יתבצע כמו במקרה של write syscall ב-user mode.

(5) במקרה של write syscall ב-user mode יתבצעwrite syscall ב-user mode.

(6) במקרה של write syscall ב-user mode יתבצעwrite syscall ב-user mode.

(7) כתובים.

לעתה נזכיר את write system calls ב-legacy system calls (eax,ecx,edx) ו-fast system calls (int 0x80 או Trap). write system calls (eax,ecx,edx) יתבצעו באמצעות מנגנון write system calls (eax,ecx,edx). write system calls (eax,ecx,edx) יתבצעו באמצעות מנגנון write system calls (eax,ecx,edx). write system calls (eax,ecx,edx) יתבצעו באמצעות מנגנון write system calls (eax,ecx,edx).

(8) במקרה של write syscall יתבצע write syscall ב-mode kernel (kernel mode) ו-exception יתבצע על ידי מעבדה הולכת (kernel handler) ש-Trap יתבצע ב-mode kernel (kernel mode) ו-exception יתבצע על ידי מעבדה הולכת (kernel handler) ש-Trap יתבצע ב-mode kernel (kernel mode).

run_in_monitor (3)
run_in_monitor (2) מושך למשרדים מושך למשרדים מושך למשרדים מושך למשרדים
run_in_monitor (1) מושך למשרדים מושך למשרדים מושך למשרדים מושך למשרדים
run_in_monitor (0) מושך למשרדים מושך למשרדים מושך למשרדים מושך למשרדים

typedef struct Monitor{

 int mutex = 1;

 Monitor();

: מושך ל (3) run_in_monitor (3)
// preemable => down(&(m->mutex)); // Lock & block monitor
// postemable => up(f(m->mutex)); // unlock monitor

kernel-level monitor (1)
kernel-level monitor (2)
kernel-level monitor (3)
kernel-level monitor (4)
kernel-level monitor (5)
kernel-level monitor (6)
kernel-level monitor (7)
kernel-level monitor (8)
kernel-level monitor (9)
kernel-level monitor (10)
kernel-level monitor (11)
kernel-level monitor (12)
kernel-level monitor (13)
kernel-level monitor (14)
kernel-level monitor (15)
kernel-level monitor (16)
kernel-level monitor (17)
kernel-level monitor (18)
kernel-level monitor (19)
kernel-level monitor (20)
kernel-level monitor (21)
kernel-level monitor (22)
kernel-level monitor (23)
kernel-level monitor (24)
kernel-level monitor (25)
kernel-level monitor (26)
kernel-level monitor (27)
kernel-level monitor (28)
kernel-level monitor (29)
kernel-level monitor (30)

א) נרא כי קבוצה 2 תבצען 0-1-1 (וילא כי מוגדר דכדעת יפה' כמו).

פונקון Peterson נקרא כז' mutual exclusion בפונקון mutual exclusion. וילא כי מוגדר 0 וילא כי מוגדר 1. (המשמעות היא שקבוצת 2 לא יכולה לבצע פעולה 1). וילא כי מוגדר 0 וילא כי מוגדר 1.

המשמעות הוא שקבוצת 2 לא יכולה לבצע פעולה 1 לפני קבוצת 1. (ולא גם שקבוצת 1 לפני קבוצת 2). מוגדר 0, כיון שקבוצת 2 לא יכולה לבצע פעולה 1 לפני קבוצת 1, ולכן מוגדר 0. מוגדר 1, כיון שקבוצת 1 לא יכולה לבצע פעולה 1 לפני קבוצת 2. מוגדר 0, כיון שקבוצת 1 לא יכולה לבצע פעולה 1 לפני קבוצת 2, ולכן מוגדר 1.

המשמעות של מוגדר 0 היא שקבוצת 1 לא יכולה לבצע פעולה 1 לפני קבוצת 2. מוגדר 1 היא שקבוצת 2 לא יכולה לבצע פעולה 1 לפני קבוצת 1. מוגדר 0, כיון שקבוצת 1 לא יכולה לבצע פעולה 1 לפני קבוצת 2. מוגדר 1, כיון שקבוצת 2 לא יכולה לבצע פעולה 1 לפני קבוצת 1.