

作業 11: 了解 context-switch 的 overhead

學習目標:

- Context-switch 分為二種，一種是自願性的，例如：使用 I/O。另一種是非自願性的，例如：用完 time slice。了解這二種情況對系統效能的影響
- 不管是自願性或者非自願性，其 overhead 有一部分來自於「執行排程器和 task switch 的程式碼」，前者是 Linux 的 CFS 排成演算法和 `_switch_to` 函數。後者則是「將 CPU 中的資料更新為新的 task 的資料」，例如：cache、TLB 等等

前言:

- 請先將 `/proc/sys/kernel/sched_min_granularity_ns` 盡可能的設定為最小值，例如：`echo 10000 > sched_min_granularity_ns`
 - 在我的電腦上預設值是 2250000
 - 在我的電腦上最小可以設置為 100000
- 設定 `/proc/sys/kernel/sched_latency_ns` 為任意數值，例如：
10000~10000000000
 - 在我的電腦上預設值是 18000000
 - 在我的電腦上可設置為 100000~10000000000
- 執行應用程式

- `time ./reportChildStat ./cpu& ./cpu`
 - ◆ 當 `sched_latency_ns` 為 10000000000 時
 - 迴圈執行次數為: `iteration = 3.14e+05`
 - Context-switch 的數量為: 64+1
 - ◆ 當 `sched_latency_ns` 為 100000 時
 - 迴圈執行次數為: `iteration = 3.14e+05`
 - Context-switch 的數量為: 2404+1
- `time ./reportChildStat ./cpu& ./cpu& ./cpu& ./cpu&`
 - ◆ 當 `sched_latency_ns` 為 10000000000 時
 - `iteration = 3.145e+05`
 - Context-switch 的數量為: 71+1
 - ◆ 當 `sched_latency_ns` 為 100000 時
 - `iteration = 3.105e+05`
 - Context-switch 的數量為: 2502+1

題目：

- 解釋: `sched_latency_ns` 和 `sched_min_granularity_ns`
- 設計實驗, 說明 context-switch 的次數與效能的關係
- 每組實驗最少進行 10 次
- Linux 為了避免「腦殘用戶」因此限定了 `sched_latency_ns` 和

sched_min_granularity_ns 的「可設定的範圍」，你要如何設計實

驗才可以得到更多的「context-switch 與效能的數據」呢？

- 請將你的數據，想辦法用 excel 或其他製圖工具，用曲線圖或者 bar chart，顯示出 context-switch 與效能的關係，建議：x-y 散佈圖
- 提示：
 - ◆ 修改 Linux kernel
 - ◆ 試試看在 user mode app 中加入 yield 或者是 sleep，讓應用程式「自願切換」

作業繳交：

- 繳交報告：
 - 姓名、學號（請隱匿個人資訊）
 - 實驗設計：以條列的方式說明，約 50 個字
 - 說明數據：附上圖表，並解釋圖表，約 50 個字

繳交：

1. 繳交期限：參考網站上的時間
2. 如果真的不會寫，記得去請教朋友。在你的報告上寫你請教了誰即可。