Project #4. Synchronization

計算機科學中，同步（synchronization）是指兩個不同但有聯繫的概念：進程同步與數據同步。進程同步指多個進程在特定點會合（join up）或者握手使得達成協議或者使得操作序列有序。數據同步指一個數據集的多份拷貝一致以維護完整性。常用進程同步原語實現數據同步。

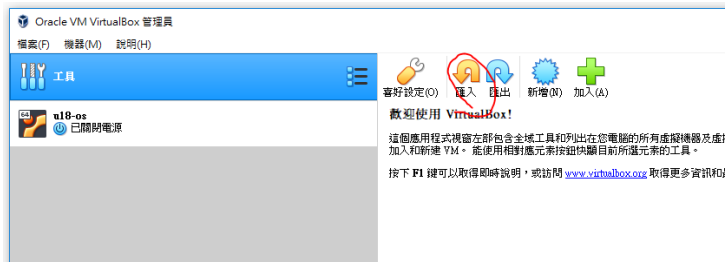
Preparation

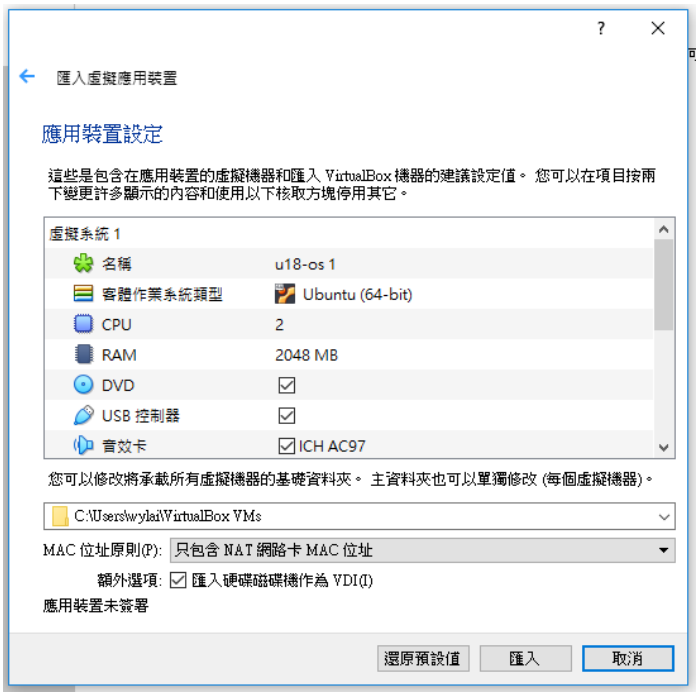
這次作業助教分別提供 Docker Imag 以及 Virtualbox 兩種環境供大家使用，作業也放在環境的家目錄底下

Docker 環境下載 (Ubuntu 16.04 or Ubuntu 18.04)

```
$ sudo apt-get update
$ sudo apt-get install docker-compose
$ sudo docker run -it --cpus=2 pandaft/os_project /bin/bash
```

VirtualBox 虛擬機映像檔 連結 https://bit.ly/2DQXdEd
VirtualBox 官網 https://www.virtualbox.org/
VM_User: os_project Password: 0
**使用虛擬機若無法正常啟動，需要先進入 UEFI/BIOS 介面 enable Virtualization Technology / VT-X support.
下載完成後選取檔案直接匯入至 VirtualBox 內執行





Note: 機器最少要有兩個 CPU Core.

A. Spin-lock Implementation

Spin lock 是一種 busy waiting 的 synchronization 機制，它可以防止兩個線程同時進入 critical section。在這個部分，你必須使用 x64 指令集組合語言來實作 spinlock，並且當"***context-swich***"和"***out-of-order execution***"發生時，必須確保你的程式碼正確執行。

In this part, we supply the following files.

[Makefile]: Compile the project by `make` command, create ***spinlock*** executable file.
[main.c]: Create two threads and a counter, and call **spin_lock(&mutex) and spin_unlock(&mutex)**.
[check.sh]: Will execute ***spinlock*** for 100 times, and grade your code.
[spinlock.s]: Implement your spinlock in here.

Note: 當程式 compile 成功可執行 check.sh 來驗證是否結果正確。

SingleTest:

```
$ ./spinlock 1000
```

B. Producer-Consumer Problem

當我們從網路上下載資料，大致會經過兩次寫入，這些資料必定首先經由 socket 讀取資料寫入在記憶體中，再將資料從記憶體中寫入硬碟，在這部分，我們可以使用 multi-process 的方式將這兩個步驟切開來加速下載的流程，你必須使用 ***semaphore*** 來同步這兩支程式運行，使得最後下載得到的 output file 跟原始資料的 sha256 比對相同。圖 1 為本次作業的 Server/Client 架構圖，需要先執行起 server.py 才能接收檔案下載 request。圖 2 為編譯完成後的 client 程式內部流程圖。

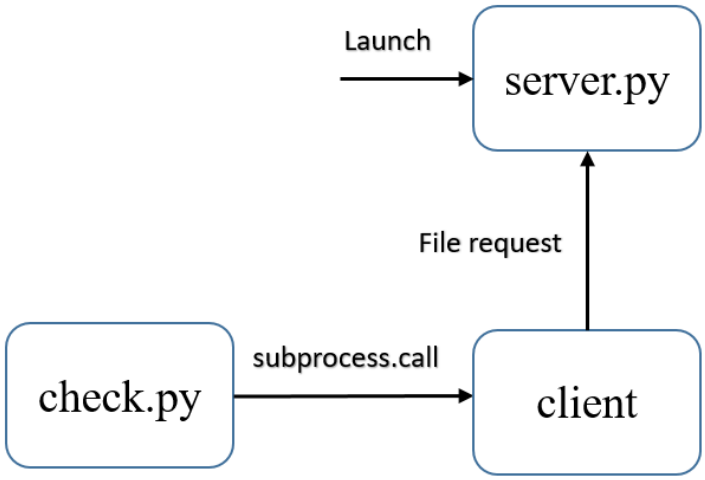


圖 1. Server/Client Architecture

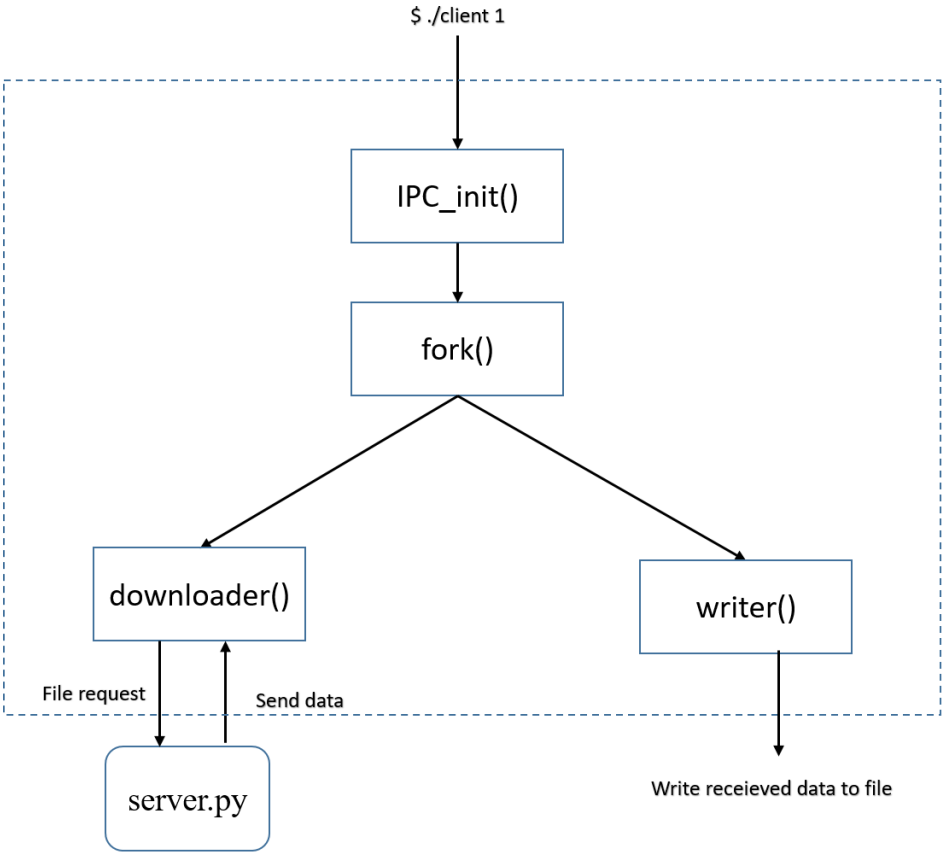


圖 2. Client Control Flow Graph

In this part, we supply the following files.

[client.c]: Create IPCs inat IPC_init function, fork two process as downloader and writer, and implement writer and downloader function by using semaphore.

[Makefile]:Compile the project by `make` command, and create ***client*** executable file.

[server.py]: A simple file server.

[check.py]: Will Execute ***client*** for 10 times, and grade your code.

Note: 當 Compile 成功，請先使用 python3 依序執行 ***[server.py]***, ***[client.py]*** 來驗證是否正確.

Tasks

A. Implement [spinlock.s].

1. Implement two functions, ***spin_lock*** and ***spin_unlock*** by using x64 assembly

and make sure that running ***check.sh*** script successfully**. Submit [spinlock.s] only!!!** [30%]

2. Please simply describe how to prevent "***context switch"*** and ***"out-of-order execution"*** issues. Write down your answer in spinlock.txt. **Submit [spinlock.pdf]** [20%]

B. Implement [client.c].

1. Fill up ***"TODO"*** and using **semaphore** to synchronization ***downloader*** and ***writer***. **Submit [client.c] only!!!** [50%]

Note: Put **[spinlock.s], [client.c], [spinlock.pdf]** in the folder named your student_id and compress as zip file. Submit Student_ID.zip