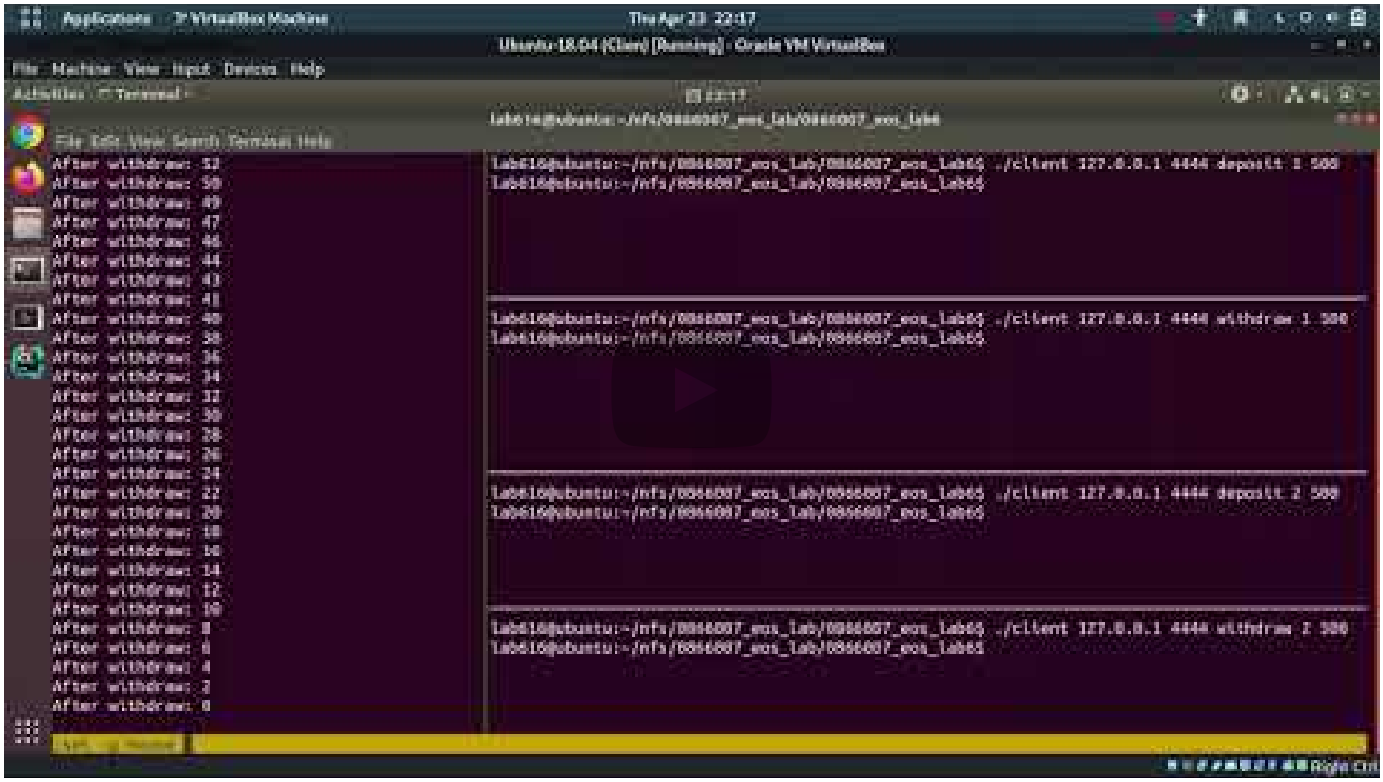
Lab 6 Web ATM



I. Introduction

請在 VM 上撰寫一隻 socket 程式。多個客戶端同時連線進來對同一個帳戶進行存款與提款的動作，而伺服器端要能正確處理同時存取所導致的 race condition 問題，才不至於造成客戶金錢上的損失。

II. Specification

- Server (**自行撰寫**)
  - 建立 socket server，等待 client 來建立連線。
  - server 端要能夠支援同時多人連線的情境。
  - 連線建立後，要根據客戶的要求做**存款**或是**提款**的動作。
  - 伺服器端只會有一個帳戶，由多人共用。
  - 輸出每次動作完成後，輸出帳戶的餘額。

```
After deposit: 2
After withdraw: 1
```

- Client (**自行撰寫**)
  - 撰寫 socket client，指定要求(存款或提款、多少錢)，然後將該資訊傳給 server 處裡。

```
<deposit | withdraw> <amount>
```

- 該動作將使用迴圈重複做 n 次，方便助教檢查。

III. Illustration

- Server

```
./server <port>
```

  - <port> 為 server listen 的端口
- Client

```
./client <ip> <port> <deposit/withdraw> <amount> <times>
```

  - <ip> 為 server 所在的 ip
  - <port> 為 server listen 的端口
  - <deposit/withdraw> 存款或提款
  - <amount> 多少錢
  - <times> 該動作將會用迴圈做多少次

IV. Note

- socket 與 thread 的撰寫請參考 laball.pdf 之 lab 3, 6-2 的範例程式碼。
- semaphore 的撰寫請參考 laball.pdf 之 lab 4 的範例程式碼。
- 請透過 semaphore 來處理同時存取造成的 race condition 問題。
- 請在 server 程式結束時 (Ctrl + C)，請將建立出來的 semaphore 給清除。
- 先ctrl-b，再用指令來讓視窗滾動

```
:set mouse on
```

V. Demo & Submission

- 助教有提供 demo.sh (http://demo.sh) 來協助同學測試與 Demo。
- 助教會透過下列指令檢查同學是否有在系統上留下 semaphore。

```
ipcs -s
```

- 請將程式碼以下列的格式擺放與命名，以方便助教評分。

```
0866007_eos_lab6
|-- Makefile
|-- server.cpp
|-- client.cpp
|-- demo.sh
```

- 請將上述之資料夾壓縮為單一 zip 檔案，並上傳到 E3 上。