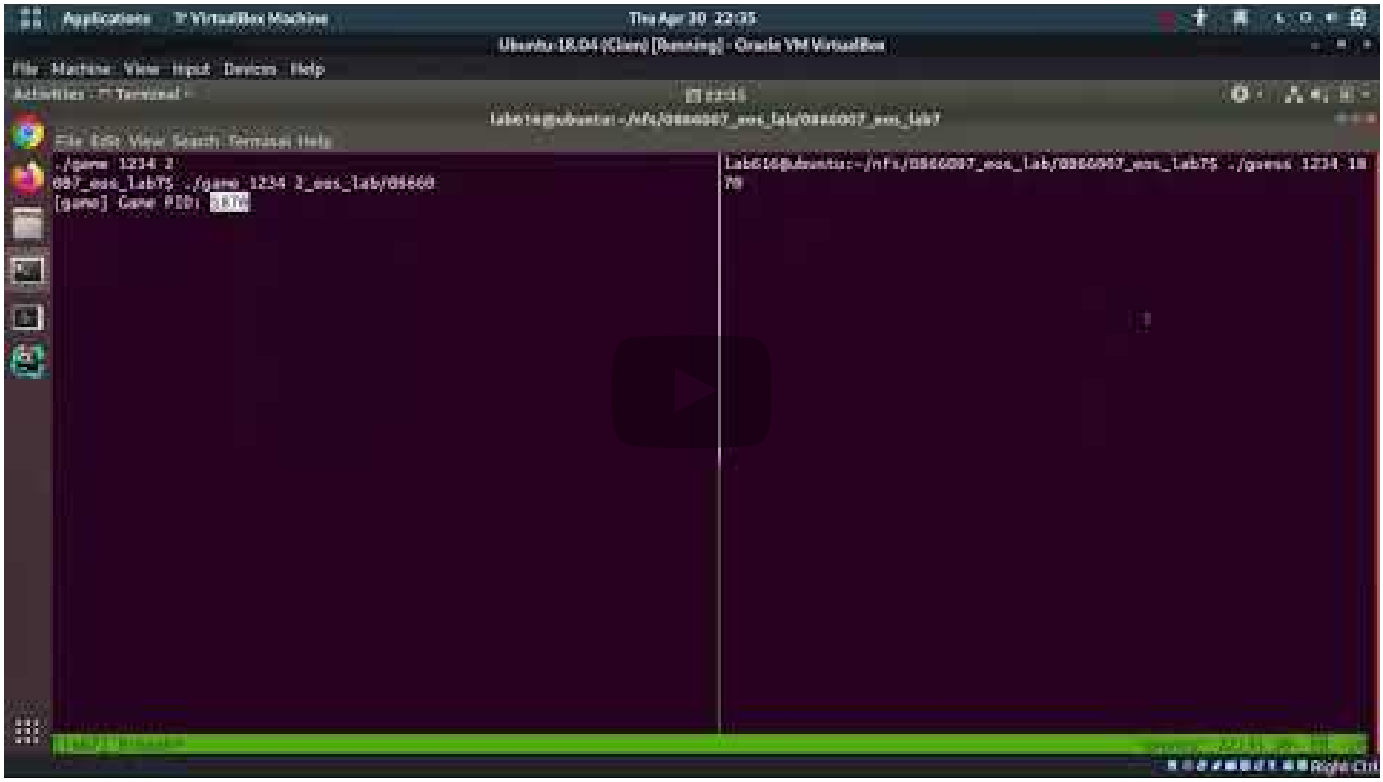
Lab 7 終極密碼



I. Introduction

請在 VM 上撰寫終極密碼遊戲。該遊戲由兩份程式組成，第一份程式利用 timer，每格一秒做猜數字的動作，用 signal 通知第二份程式，其會將被猜的數字讀進來，並給出猜中了、太大、太小這三種回應。兩隻程式透過 shared memory 做資料傳出的動作。

II. Specification

- Game 程式 (**自行撰寫**)
  - 取得 shared memory 的定址空間。
  - shared memory 除存的資料格式如下：

```
typedef struct {
    int guess;
    char result[8];
}data;
```

  - Guess 程式使用 **SIGUSR1** 信號來通知 Game 程式做數值上的判斷（太大、大小、猜中）。
  - 撰寫 **SIGUSR1** handler function，將 shared memory 中的 guess 變數與被猜的數字做比較，並將結果寫回 result 變數當中，最後發送 **SIGUSR1** 信號給 Guess 程式表示判斷

完成。

- Guess 程式 (**自行撰寫**)
  - 取得 shared memory 的定址空間。
  - 本程式使用 **對半猜(二分搜)** 的方式來做猜測的動作。
  - Game 程式使用 **SIGUSR1** 信號來通知 Game 程式去接收判斷結果。
  - 撰寫 **SIGUSR1** handler function，讀取 result 變數的結果，並更新猜測的數值範圍。
  - 撰寫 timer，每秒會將該回合要猜的數字寫入 guess 變數當中，並用 **SIGUSR1** 通知 Game 程式處理。

III. Illustration

- Game 程式

```
./game <key> <guess>
```

  - <key> 為 shared memory 的識別 key，同一個 key 將會指向相同的 shared memroy。
  - <guess> 為被猜的數值。
- Guess 程式

```
./guess <key> <upper_bound> <pid>
```

  - <key> 為 shared memory 的識別 key
  - <upper_bound> 為猜測數值之最大值，以 100 為例，猜測的範圍即為 1 - 100
  - <pid> 為 game 程式的 process id

IV. Note

- shared memory 的撰寫請參考 pdf 上 lab 5 的範例程式碼。
- timer, signal 的撰寫請參考 pdf 上 lab 8 的範例程式碼。
- 為了避免程式直接退出，請使用 **nanosleep()** 來做 blocking 的動作，寫法請參考 pdf 上 lab 8 的範例程式碼。
- 請在 Game 程式結束時 (Ctrl + C)，請將建立出來的 shared memory 給清除。

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 shared memroy。

```
ipcs -m
```

- 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_lab7
|-- Makefile
|-- game.cpp
|-- guess.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。