ĐỒ ÁN THỰC HÀNH 1 – LẬP TRÌNH SOCKET

MÔN MẠNG MÁY TÍNH

1. Quy định chung

- Đồ án được làm theo nhóm: mỗi nhóm tối đa **3** sinh viên, tối thiểu **2** sinh viên (trong trường hợp sĩ số lớp lẻ), sinh viên tự chọn nhóm (sử dụng nhóm thực hành đã đăng ký nếu có).

- **Các bài làm giống nhau sẽ đều bị điểm 0 toàn bộ phần thực hành tất cả các nhóm liên quan (dù có điểm các bài tập, đồ án thực hành khác)**.

- Môi trường lập trình: Tự do lựa chọn ngôn ngữ lập trình, tự do lựa chọn môi trường hệ điều hành: Windows, Unix/Linux, macOS

- Ngôn ngữ lập trình GV có thể hỗ trợ: C/C++, C#, Java, Python

- Thư viện hỗ trợ lập trình socket cho phép sử dụng như: Socket, CSocket, winsock. Tức là chỉ sử dụng các thư viện Socket do ngôn ngữ lập trình cung cấp, không dùng các thư viện bên ngoài liên quan đến Socket

2. Cách thức nộp bài

- **Nộp bài trực tiếp trên Website môn học, không chấp nhận nộp bài qua email hay hình thức khác.**
- Tên file: **MSSV1_MSSV2_MSSV3.zip** (Với MSSV1 < MSSV2 < MSSV3)

Ví dụ: Nhóm gồm 3 sinh viên: 2012001, 2012002, và 2012003, tên file nộp: **2212001_2212002_2212003.zip**

Cấu trúc file nộp gồm thư mục MSSV1_MSSV2_MSSV3 chứa các file:

1. **Report.pdf:** chứa báo cáo về bài làm
2. **Release**: thư mục chứa file thực thi của chương trình, **nếu có** (*.exe, nếu Python thì không cần)
3. **Source**: thư mục chứa source code của chương trình , yêu cầu nộp cả project đã xóa bỏ thư mục Debug và các file không cần thiết khác.. *Nhóm nào chỉ nộp file *.cpp và *.h và không biên dịch được thì bị 0 điểm.*

Lưu ý: Cần thực hiện đúng các yêu cầu trên, nếu không, bài làm sẽ không được chấm.

3. Hình thức chấm bài

Chấm vấn đáp vào thời điểm kết thúc phần thực hành.

4. Tiêu chí đánh giá

Về chương trình:

- Mục tiêu của đồ án này tập trung chủ yếu vào 2 vấn đề: lập trình socket, tự đưa ra được giao thức truyền file giữa client và server. Do đó các tiêu chí đánh giá dựa vào các chức năng chính được liệt kê trong yêu cầu của chương trình (có ghi chú thang điểm cho từng chức năng)

Về báo cáo:

- Thông tin của nhóm.
- Đánh giá mức độ hoàn thành từ 0 – 100% (Chú thích rõ những mục làm được,chưa làm được và còn bị lỗi)
- Kịch bản giao tiếp của chương trình: Giao thức trao đổi giữa client và server, cấu trúc thông điệp, kiểu dữ liệu của thông điệp, cách tổ chức cơ sở dữ liệu (nếu có).
- Môi trường lập trình và các framework hỗ trợ để thực thi ứng dụng.
- Hướng dẫn sử dụng các tính năng chương trình.
- Bảng phân công công việc và cho biết rõ ràng ai làm việc gì cách rõ ràng. Không ghi chung chung như chia đều công việc hay cùng làm mọi việc.
- Các nguồn tài liệu tham khảo.

Lưu ý: Trong báo cáo không dán các đoạn source code của chương trình. Mã chương trình chỉ trình bày nếu thật sự cần thiết và nếu cần minh họa cho các mô hình cài đặt hay các cơ chế đồng bộ (minh họa dạng mã giả, prototype hàm).

Về vấn đáp:

- Chuẩn bị thiết bị để chạy Client và Server trên 2 máy khác nhau (có thể dùng một máy thật và 1 máy ảo, ...), chương trình, báo cáo đầy đủ (không cần in).
- Trả lời các câu hỏi từ GV
- Trường hợp trả lời sai hoặc không trả lời được sẽ trừ trực tiếp điểm vào tổng điểm đồ án.

Lưu ý: Tất cả thành viên của nhóm phải tham gia buổi vấn đáp. Thành viên vắng mặt sẽ xử lý theo quy định sau:

- Có phép (gửi email xin phép trước buổi vấn đáp): trừ điểm vấn đáp trực tiếp
- Không phép: 0 điểm đồ án.

5. Nội dung:

Lưu ý nội dung đồ án gồm có 2 phần cần được thực hiện.

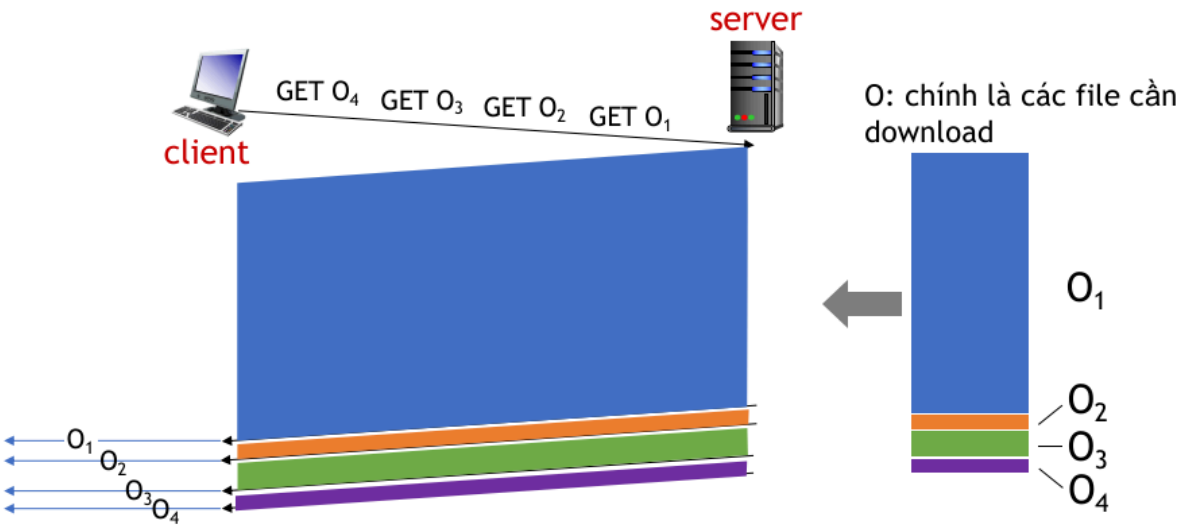
Phần I (3đ):

- Cập nhật ngày 09.07.2024:

+ Server sẽ phục vụ tuần tự từng Client, nghĩa là sao khi một Client đóng kết nối thì mới accept và phục vu Client tiếp theo.

+ Vì Client download tuần tự từng file, nên không cần phải quét 2s 1 lần, mà download xong file thì quét file input.txt để biết những file cần download mới.

Viết chương trình Client/Server cho phép **nhiều** Client download file từ 1 Server. Server sẽ phục vụ tuần tự từng Client.



Server: Sử dụng 1 file Text hoặc JSON (hoặc sử dụng file có cấu trúc khác) để lưu danh sách các file cho phép client download gồm *tên file, và dung lượng*. Ví dụ file text (.txt):

```
File1.zip 5MB
File2.zip 10MB
File3.zip 20MB
File4.zip 50MB
File5.zip 100MB
File6.zip 200MB
File7.zip 512MB
File8.zip 1GB
```

Client:

- Client kết nối đến Server, Client sẽ nhận được thông tin danh sách các file từ server và hiển thị trên màn hình.
- Để thực hiện việc download file từ server thì client có một file để ghi nhận danh sách các tên file sẽ được download, file này là input.txt. Người dùng sẽ thêm vào cuối danh sách file cần download trong tập tin input.txt (không xóa) tên những file cần download bất cứ khi nào muốn mà không làm ảnh hưởng việc thực thi của chương trình (Lưu ý: có thể thêm một hoặc nhiều tên file mỗi lần, mở file bằng chương trình nào đó và thêm vào bằng tay).
- Client chịu trách nhiệm duyệt file input.txt ~~**mỗi 2s một lần**~~, để lấy được danh sách file vừa được thêm mới và thực hiện download từ server. (không thực hiện download những file đã duyệt trước đó).

Ví dụ file input.txt

```
File1.zip
File2.zip (Ví dụ input sau khoảng 20 giây và lưu file này lại, client sẽ phát hiện so với lần đọc trước đó, thì File2.zip là file mới cần được download)
```

- Mỗi Client chỉ dùng duy nhất một kết nối để ***download tuần tự*** từng file một thông qua kết nối đó và lưu mặc định vào `output` folder của Client. Trên màn hình client cho phép hiển thị tỉ lệ % (từ 0-100%) dựa trên tiến độ download của file đang thực hiện download (Lưu ý: Một client chỉ mở duy nhất một kết nối tới Server để download tất cả những file từ người dùng).
- Khi Client bấm "Ctrl + C" thì chương trình client đóng kết nối đến Server và kết thúc chương trình. (https://www.tutorialspoint.com/how-do-i-catch-a-ctrlplusc-event-in-cplusplus)

Ví dụ màn hình console của Client:

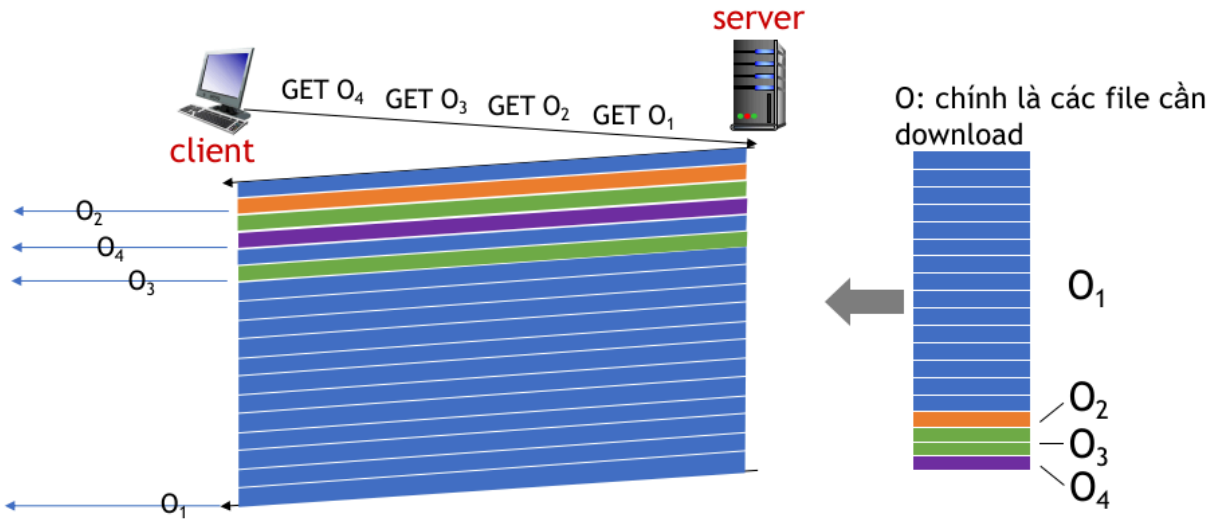
```
Downloading File5.zip ....  45%
```

Phần II (6.5đ):

- Cập nhật ngày 09.07.2024:

+ Server phục vụ đồng thời nhiều Client cùng lúc.

Viết chương trình Client/Server cho phép **nhiều** Client download file từ Server theo độ ưu tiên. Lưu ý Server phục vụ nhiều Client cùng lúc.



Server: sử dụng 1 file Text hoặc JSON (hoặc sử dụng file có cấu trúc khác) để lưu danh sách các file cho phép client download gồm *tên file, và dung lượng*. Ví dụ file text (.txt):

```
File1.zip 5MB
File2.zip 10MB
File3.zip 20MB
File4.zip 50MB
File5.zip 100MB
File6.zip 200MB
File7.zip 512MB
File8.zip 1GB
```

Client:

- Client kết nối đến Server, Client sẽ nhận được thông tin danh sách các file từ server và hiển thị trên màn hình.
- Client download các file có tên trong tập tin input.txt tương tự như yêu cầu ở phần 1. Người dùng sẽ thêm vào cuối danh sách file cần download trong tập tin input.txt (không xóa) tên những files cần download và *độ ưu tiên download*

(CRITICAL: Ưu tiên download nhanh gấp 10 lần NORMAL, HIGH: Ưu tiên download nhanh gấp 4 lần NORMAL, NORMAL) bất cứ khi nào muốn mà không làm ảnh hưởng việc thực thi của chương trình (Lưu ý: có thể thêm một hoặc nhiều tên file mỗi lần)..

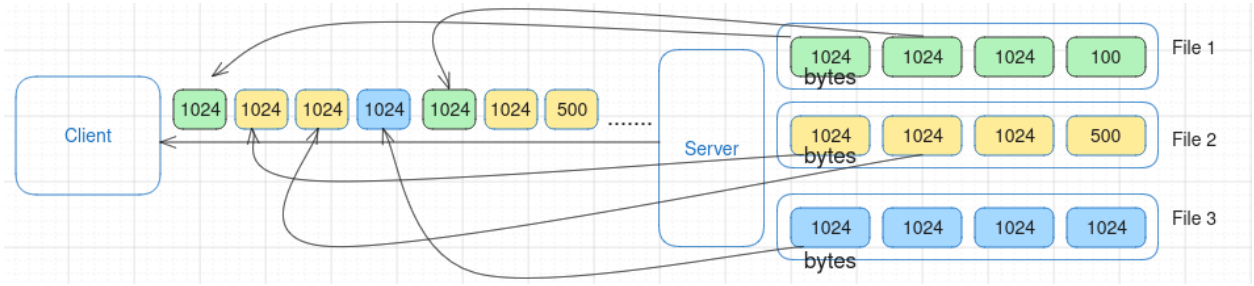
- Client chịu trách nhiệm duyệt file input.txt mỗi 2s một lần, để lấy được danh sách file vừa được thêm mới và thực hiện download từ server. (không thực hiện download những file đã duyệt trước đó)

Ví dụ file input.txt, thì File4.zip sẽ được download xong trước, sau đó đến File3.zip và File2.zip

File2.zip NORMAL File3.zip HIGH File4.zip CRITICAL

- Mỗi Client chỉ dùng duy nhất một kết nối để ***download tuần tự*** các phần nhỏ xen kẽ của từng file từ server thông qua kết nối đó và lưu mặc định vào `output` folder của Client. Trên màn hình client cho phép hiển thị tỉ lệ % (từ 0-100%) dựa trên tiến độ download của các files đang thực hiện download.
- Khi Client bấm "Ctrl + C" thì chương trình client đóng kết nối đến Server và kết thúc chương trình.
(https://www.tutorialspoint.com/how-do-i-catch-a-ctrlplusc-event-in-cplusplus)

Giải thích cách download của client:



- Ví dụ theo hình trên, client cần download 3 tập tin File1.txt, File2.txt và File3.txt. Để hiển thị được tỉ lệ download và tăng độ thân thiện của ứng dụng, tránh sự chờ đợi cho người dùng, các tập tin được đọc từng phần nhỏ (chunk) có kích thước nhỏ hơn trong quá trình download. Sinh viên cần xử lý để client có thể download tuần tự các chunk của từng tập tin kèm theo độ ưu tiên như ví dụ trong hình:
    - File1.txt và File3.txt có độ ưu tiên download thấp hơn file2.txt nên mỗi lần chỉ download một chunk.
    - File2.txt có độ ưu tiên cao hơn nên mỗi lần được download 2 chunks
- Sinh viên hãy tùy chỉnh số lượng chunk cần tải của mỗi tập tin tùy vào độ ưu tiên được thiết lập trong file input.txt

- **Gợi ý:** Sinh viên cần tạo protocol cho chương trình của mình (Application protocol) để đánh dấu chunk bắt đầu-đang-kết thúc download thuộc tập tin nào và khi nhận thì ghi các chunk đúng và đủ vào file để nhận đủ 100% dung lượng tập tin cần download.

Ví dụ màn hình console của Client:

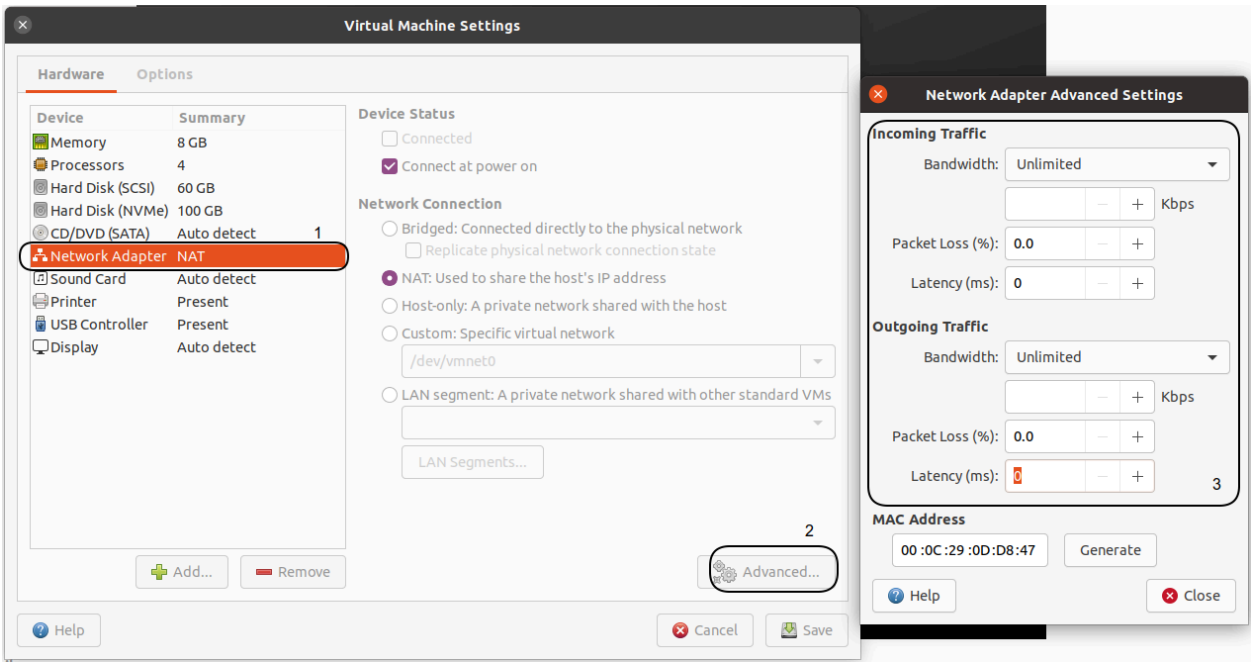
```
Downloading File1.zip ....  20%
Downloading File2.zip ....  40%
Downloading File3.zip ....  20%
```

6. Tài nguyên

Các bạn có thể lấy các file test có dung lượng khác nhau ở đây (5MB, 10MB, 20 MB, 50 MB, ...):

- https://www.thinkbroadband.com/download
- https://ash-speed.hetzner.com/
- http://xcal1.vodafone.co.uk/

Trong trường hợp bạn muốn test bandwidth chậm với máy ảo để test (chạy Server hoặc Client), bạn có thể chỉnh bandwidth ở đây để giới hạn tốc độ gửi chậm lại



Hoặc bạn có thể dùng tools bên dưới để giới hạn tốc độ bandwidth giữa Client-Server:

- NetLimiter
- TMeter
- Clumsy

Hoặc có thể dùng sleep để test chương trình của mình.

7. Thang điểm

STT	Yêu cầu	Điểm
Phần I	Client có thể nhận được danh sách các file từ Server và ctrl-c	0,5
Phần I	Client có thể nhận lần lượt từng file thành công từ Server. Server có thể gửi file thành công tới Client	2
Phần I	Hiển thị percent download file và phát hiện những file cần download tiếp theo ~~2s quét file input.txt 1 lần~~	0,5
Phần II	Client có thể nhận được danh sách các file từ Server và ctrl-c	0,5
Phần II	2s quét file input.txt 1 lần	0,5
Phần II	Hiển thị percent download files	1
Phần II	Client có thể nhận files thành công từ Server. Tập tin sau khi download phải đúng và đủ dung lượng	3
Phần II	Độ ưu tiên CRITICAL, HIGH, NORMAL	1,5
	Báo cáo	0,5