

VIETNAM NATIONAL UNIVERSITY HO CHI MINH CITY
HO CHI MINH CITY UNIVERSITY OF TECHNOLOGY
FACULTY OF COMPUTER SCIENCE AND ENGINEERING



COMPUTER NETWORKS

Delevop a network application

FILE-SHARING APPLICATION

Advisor: Bùi Xuân Giang
Students: Kim Gia Bảo - 2152417
Lê Nhật Anh - 2153158
Nguyễn Thế Cường - 2153240
Đỗ Nhật Thái - 2152964

HO CHI MINH CITY, NOVEMBER 2023

Contents

1	Mở đầu	3
1.1	Đặt vấn đề	3
1.2	Mục tiêu	3
1.3	Yêu cầu	4
1.4	Hiện thực	5
2	Cơ sở lí thuyết	6
2.1	Giao thức	6
2.2	Giao thức TCP/IP	6
3	Các chức năng của ứng dụng	7
3.1	Server	7
3.1.1	Chức năng quản lý	7
3.1.2	Lưu trữ danh sách các client đang kết nối với server và danh sách file của các client đây . .	7
3.1.3	Trả lời yêu cầu của các client	7
3.1.4	Trình thông dịch dòng lệnh: 'discover', 'ping' .	8
3.2	Client	8
3.2.1	Thông báo cho server biết những file đang có .	8
3.2.2	Gửi thông tin để truy hồi file	9
3.2.3	Trình thông dịch dòng lệnh: 'publish', 'fetch' .	9
4	Các giao thức cho ứng dụng	10
4.1	Giữa các tính năng khi giao tiếp giữa client và server	10
4.1.1	Ping	10
4.1.2	Discover	10
4.1.3	Publish	10
4.1.4	Fetch	10
4.2	Giữa các tính năng khi giao tiếp giữa client và client	11
4.2.1	Fetch	11



5	Diagram	12
5.1	Use-case diagram	12
5.2	Class diagram	13
6	Kết quả hiện thực	14
6.1	Server	14
6.1.1	Nhận kết nối từ các Client	14
6.1.2	Nhận và xử lý yêu cầu từ các Client	14
6.1.3	Thực hiện các lệnh trên Server	15
6.2	Client	16
6.2.1	Thực hiện publish và fetch text file cơ bản . .	16
6.2.2	Thực hiện publish và fetch các file type đa dạng và các file có kích thước lớn	17
6.3	Phía cơ sở dữ liệu	18
7	Kết quả đầu ra	20
8	Đánh giá hiệu năng	21
9	Hướng dẫn sử dụng	22
10	Kết luận	22

1 Mở đầu

1.1 Đặt vấn đề

Trong thời đại công nghệ thông tin phát triển như hiện nay, nhu cầu trao đổi dữ liệu qua lại giữa các máy tính với nhau là một nhu cầu vô cùng thiết yếu. Có rất nhiều phương pháp để thực hiện, trong đó, nổi bật có phương pháp truyền tệp ngang hàng **peer-to-peer**.

Phương pháp truyền tệp ngang hàng **peer-to-peer** là phương pháp truyền tệp trực tiếp giữa hai máy tính mà không cần phải truyền qua một máy chủ trung gian. Một số ưu điểm của phương pháp này là:

- Tốc độ truyền tệp nhanh hơn so với việc truyền qua một máy chủ trung gian.
- Không phụ thuộc vào một máy chủ trung gian, qua đó tăng độ tin cậy và bảo mật.

Với các ưu điểm đó, nhóm đã sử dụng phương pháp truyền tệp ngang hàng **peer-to-peer** để xây dựng ứng dụng truyền tệp giữa các máy tính cá nhân với nhau.

1.2 Mục tiêu

Mục tiêu của nhóm khi xây dựng ứng dụng là:

- Xây dựng một ứng dụng truyền tệp ngang hàng đơn giản, dễ sử dụng
- Hỗ trợ truyền tệp giữa các máy tính với nhau
- Hỗ trợ truyền tệp có kích thước tệp tin lớn

Để thực hiện hóa mục tiêu này, nhóm đã thực hiện:

- Xây dựng giao diện **Client CLI** trực quan, dễ sử dụng

- Xây dựng giao diện **Server CLI** trực quan, hỗ trợ cho việc theo dõi giữa các client với nhau
- Xây dựng hệ **Cơ sở dữ liệu** để lưu các thông tin Client, danh sách file ở từng client
- **Hiện thực** và **Kiểm thử** các hàm thực hiện truyền tải thông tin giữa các client với nhau

1.3 Yêu cầu

Các yêu cầu chức năng cho ứng dụng:

- Server có tính năng quản lý các thông tin đăng nhập, danh sách file của từng máy client
- Server có thể thực hiện được lệnh *ping* để kiểm tra trạng thái online/offline của client
- Server có thể thực hiện được lệnh *discover* để kiểm tra danh sách file của client
- Các client phải thực hiện được lệnh *publish* để server lưu lại thông tin về tên file.
- Các client phải thực hiện được lệnh *fetch* để có thể truyền nhận dữ liệu qua lại với nhau

Các yêu cầu phi chức năng cho ứng dụng:

- Các thao tác trên giao diện dễ hiểu, dễ sử dụng, các tính năng có thể được thực hiện **dưới 3 thao tác**
- Đảm bảo được đường truyền giữa server với các client, và các client đối với nhau
- Gửi được các file có kích thước lớn (trên **100MB**)



1.4 Hiện thực

Để hiện thực ứng dụng đáp ứng được các yêu cầu kể trên, nhóm lựa chọn ngôn ngữ lập trình **Python** đi kèm với các thư viện **threading**, **socket**, **os**, **json** và **psycopg2** để kết nối với hệ quản trị cơ sở dữ liệu **PostgreSQL**.

2 Cơ sở lý thuyết

2.1 Giao thức

Theo slide bài giảng của giảng viên môn Mạng máy tính, **giao thức** định nghĩa **định dạng, thứ tự** của các **thông điệp** được **gửi và nhận** giữa các thực thể mạng, cũng như các **hành động** khi thực hiện truyền và nhận thông điệp.

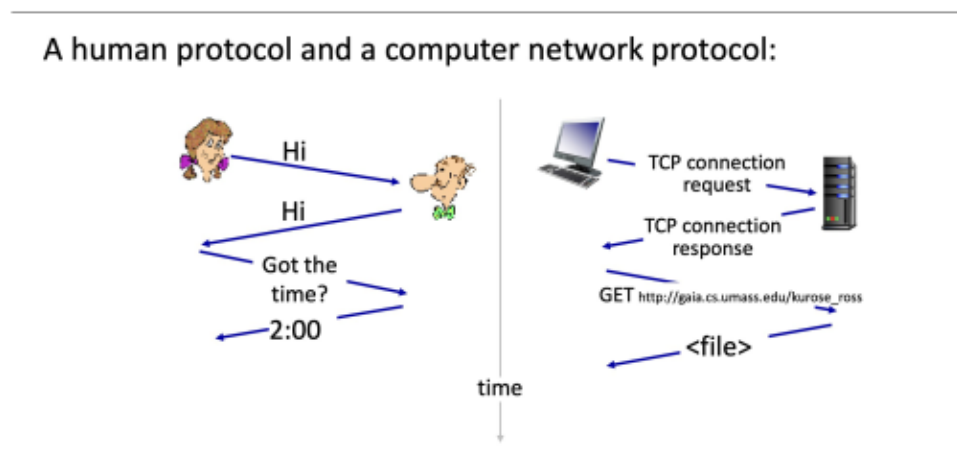


Figure 1: Minh họa về Protocol

2.2 Giao thức TCP/IP

Xét ở tầng ứng dụng, TCP/IP là một giao thức hướng kết nối, **connection-oriented protocol**. Nghĩa là trước khi có thể truyền và nhận dữ liệu, hai máy phải "bắt tay" và thiết lập kết nối TCP. Khi khởi tạo kết nối TCP, ta liên kết hai socket address lại với nhau. Khi hoàn thành khởi tạo kết nối TCP, nếu một phía muốn truyền dữ liệu qua phía còn lại, chỉ cần đưa data vào đường truyền thông qua socket của máy đấy.

3 Các chức năng của ứng dụng

3.1 Server

3.1.1 Chức năng quản lý

Máy chủ trở thành **Quản lý** của hệ thống truyền file, nhằm **theo dõi** client đang kết nối, file mà client đang giữ.

3.1.2 Lưu trữ danh sách các client đang kết nối với server và danh sách file của các client đây

Tính năng nhằm quản lý các file nằm ở máy **hostname**. Server tạo danh sách các client đã kết nối và các file của client đã kết nối. Khi client kết nối với server, server có thể theo dõi thông tin như địa chỉ IP, Port, trạng thái của client. Thông tin này giúp server xác định và quản lý các client một cách hiệu quả nhằm tránh, ngăn chặn các cuộc tấn công mạng hoặc hoặc giả mạo IP Address.

Ngoài ra server còn tạo một danh sách để quản lý và tìm kiếm các file mà client đã đưa lên thông qua lệnh **publish**. Danh sách này có thể bao gồm thông tin của file như tên file **fname**, tên file ở máy client **lname**, tên máy client chứa file **hostname**, địa chỉ IP đang chứa file này **ip**. Bằng cách theo dõi thông tin này, server có thể dễ dàng biết và trả lời được thông tin về máy đang giữ file mà các máy khác yêu cầu đang cần.

3.1.3 Trả lời yêu cầu của các client

Server và các client được kiến trúc theo mô hình Server-Client:

- **Client gửi yêu cầu:** Yêu cầu này là yêu cầu **fetch** hoặc yêu cầu **publish**.
- **Server nhận yêu cầu:** Server xử lý yêu cầu đến. Nó quyết định những việc cần làm dựa trên loại yêu cầu, dữ liệu được cung cấp, thông tin xác thực của client.

- **Server xử lý yêu cầu:** Server thực hiện các yêu cầu mà client yêu cầu xử lý phù hợp.
- **Server tạo và gửi phản hồi:** Sau khi xử lý yêu cầu, Server sẽ tổng hợp mọi thứ cần thiết cho phản hồi. Thông tin cụ thể sẽ tùy thuộc vào yêu cầu nhưng thông tin này thường bao gồm siêu dữ liệu có thể có và dữ liệu. Sau đó, server sẽ gửi phản hồi này lại cho client bằng giao thức thích hợp.
- **Client nhận phản hồi:** Client nhận phản hồi của server, lưu file đã tải xuống, v.v

3.1.4 Trình thông dịch dòng lệnh: 'discover', 'ping'

- **Discover hostname:** Khi nhận được yêu cầu discover, server sẽ thực hiện gửi yêu cầu đến Client ở máy hostname để kiểm các file đang có. Lệnh sẽ trả về danh sách các file đang có ở máy hostname.
- **Ping hostname:** Khi nhận được yêu cầu ping, server sẽ thực hiện gửi tín hiệu đến máy hostname. Nếu không nhận được phản hồi, lệnh sẽ phản hồi rằng máy hostname đang không hoạt động. Nếu nhận được phản hồi, lệnh sẽ phản hồi rằng máy hostname đang hoạt động.

3.2 Client

3.2.1 Thông báo cho server biết những file đang có

Tính năng này sẽ gửi thông tin những file đang có ở máy Client lên server, sẽ có 2 trường hợp xảy ra:

- Nếu kết nối được đến Server, Client sẽ gửi danh sách tên file lên Server.
- Nếu không, Client sẽ báo lỗi.

3.2.2 Gửi thông tin để truy hồi file

Tính năng này sẽ gửi thông tin (tên file) lên Server, và sau đó Server sẽ thực hiện tìm kiếm file này ở các máy Client đang kết nối với Server và ở cơ sở dữ liệu. Sẽ có 2 trường hợp xảy ra:

- Nếu tìm ra, Server sẽ gửi về Client đang thực hiện yêu cầu các thông tin về file (tên của file ở local, file đang ở client nào). Sau đó, nếu người dùng ở máy Client đang thực hiện yêu cầu muốn tải về file đấy, file sẽ được chuyển thẳng giữa 2 máy Client, không thông qua Server.
- Nếu không tìm ra, hệ thống sẽ báo lỗi và kết thúc quá trình tìm kiếm. Khi này, nếu người dùng muốn, có thể nhập lại tên file và nhấn nút để tiếp tục thực hiện tìm kiếm mới.

3.2.3 Trình thông dịch dòng lệnh: 'publish', 'fetch'

Hệ thống sẽ có một phần hỗ trợ 2 lệnh là publish và fetch cụ thể như sau:

- **publish lname fname:** Client có file tên lname sẽ được thêm vào cơ sở dữ liệu của server với tên fname.
- **fetch fname:** Gửi yêu cầu tìm kiếm file fname lên server, nếu tìm thấy thì client sẽ nhận được thông tin kết nối của client đang chứa tập tin đấy và tiến hành truyền tải dữ liệu.

4 Các giao thức cho ứng dụng

4.1 Giữa các tính năng khi giao tiếp giữa client và server

Các tính năng có sự giao tiếp giữa client và server sẽ sử dụng chung giao thức kết nối là **TCP** vì các thao tác giữa clients server có thể chấp nhận độ trễ nhất định tuy nhiên yêu cầu ở server như là một trung gian nhằm quản lí cơ sở dữ liệu và kết nối giữa các client với nhau nên cần đảm bảo thông tin chính xác và tránh bị mất gói.

4.1.1 Ping

Khi Server sử dụng lệnh ping, Server sẽ gửi một yêu cầu đến Client với hostname tương ứng để kiểm tra tình trạng đang alive hay không. Nếu quá thời gian không có phản hồi, Server sẽ thông báo lại rằng Client đó đã offline và ngược lại.

4.1.2 Discover

Khi Server sử dụng lệnh discover, Server sẽ gửi một yêu cầu đến Client với hostname tương ứng để xin thông tin về các file đang có. Client sau khi nhận yêu cầu từ Server sẽ tiến hành truy vấn tên các file hiện có trong thư mục của mình sau đó gửi về lại cho Server

4.1.3 Publish

Khi một Client sử dụng lệnh publish, Client sẽ gửi một yêu cầu tới Server muốn lưu trữ một file trên máy có tên **lname** và đặt tên file này trên repository của mình là **fname**. Sau đó từ Server sẽ tiến hành cập nhật cơ sở dữ liệu và gửi một xác nhận về lại cho Client.

4.1.4 Fetch

Khi một Client sử dụng lệnh publish, Client sẽ gửi một yêu cầu tới Server muốn truy xuất một file có tên **fname** về máy mình. Server sẽ tiến hành truy vấn cơ sở dữ liệu để tìm ra các Client khác hiện đã publish file **fname** lên repository của mình và tiến hành gửi lại cho

Client đã gửi yêu cầu để Client đó tự tạo kết nối với Client đang giữ file để tiến hành chia sẻ file.

4.2 Giữa các tính năng khi giao tiếp giữa client và client

Các tính năng có sự giao tiếp giữa client và client sẽ sử dụng chung giao thức kết nối là **TCP** vì các thao tác giữa clients trong các chức năng này có thể chấp nhận độ trễ nhất định tuy nhiên yêu cầu việc dữ liệu khi nhận về cần đầy đủ, không bị mất gói.

4.2.1 Fetch

Khi Client đã kết nối đến peer, tức là một Client khác, để gửi yêu cầu được chia sẻ file **fname** trên máy của peer (hiện đang có tên **lname** đã được lưu trong cơ sở dữ liệu của Server). Peer sẽ tiến hành gửi lại nội dung trong file được yêu cầu về cho Client đã kết nối.

5 Diagram

5.1 Use-case diagram

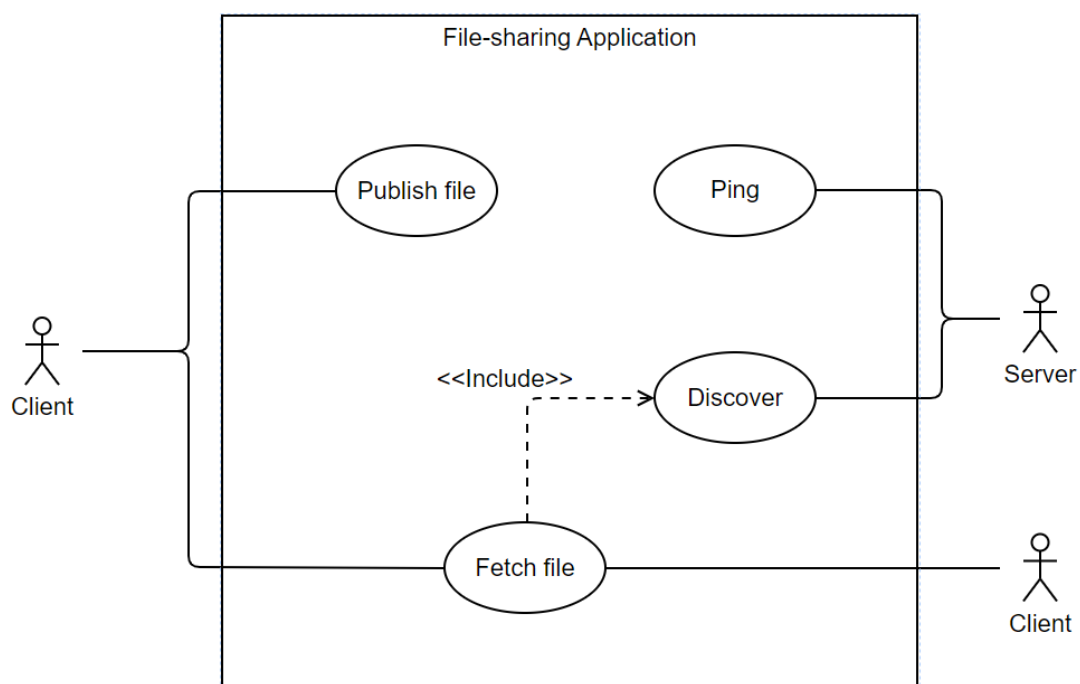


Figure 2: Use-case diagram tổng

5.2 Class diagram

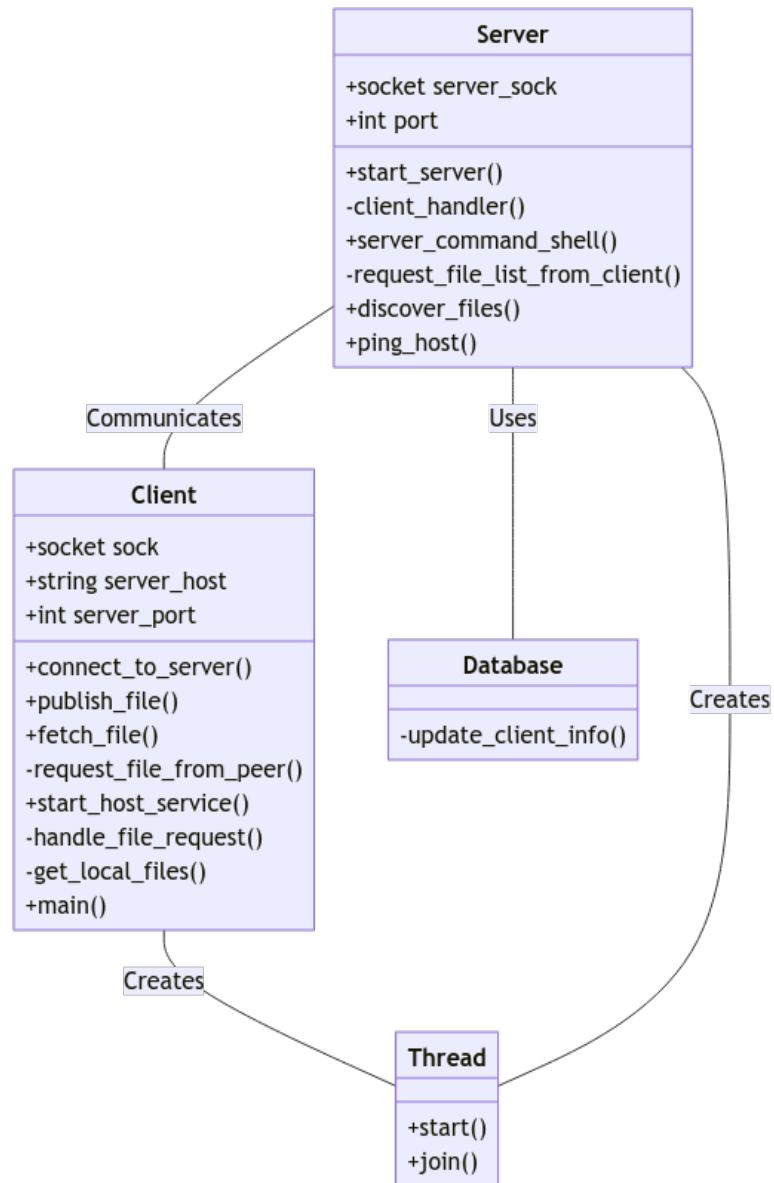


Figure 3: Class diagram tổng

6 Kết quả hiện thực

6.1 Server

6.1.1 Nhận kết nối từ các Client

Khi một Client kết nối đến Server, nó sẽ thông báo cho Server biết hostname cùng địa chỉ IP và port của mình. Server sẽ ghi nhận điều này và log ra ngoài terminal.

```
PS C:\Users\kimgi\MMT\As1> python3 server.py
Server command: 2023-11-26 16:04:35,403 - INFO - Server started and is listening for connections.
2023-11-26 16:04:46,298 - INFO - Received data from ('192.168.1.17', 54679): {"action": "introduce", "hostname": "Lio-PC"}

2023-11-26 16:04:46,299 - INFO - Active connections: 2
2023-11-26 16:04:46,299 - INFO - Connection established with 192.168.1.17 (Lio-PC)
2023-11-26 16:04:49,988 - INFO - Active connections: 3
2023-11-26 16:04:49,988 - INFO - Received data from ('192.168.1.15', 9668): {"action": "introduce", "hostname": "DESKTOP-8H0BART"}

2023-11-26 16:04:49,989 - INFO - Connection established with 192.168.1.15 (DESKTOP-8H0BART)
```

Figure 4: Server ghi nhận kết nối từ các Client

6.1.2 Nhận và xử lý yêu cầu từ các Client

Khi một Client thực hiện các lệnh **publish** hoặc **fetch** sẽ gửi yêu cầu tới Server. Lúc này phía Server sẽ ghi nhận các yêu cầu này và tiến hành xử lý tương ứng.

- Nếu là yêu cầu **publish** thì sẽ cập nhật dữ liệu tương ứng vào cơ sở dữ liệu.
- Nếu là yêu cầu **fetch** thì Server sẽ tiến hành truy vấn cơ sở dữ liệu để tìm ra các Client sở hữu các file tương ứng để trả thông tin lại về cho Client gửi yêu cầu để bên phía Client đó tự lựa chọn Client tương ứng để yêu cầu gửi file về.

```
2023-11-26 15:39:26,018 - INFO - Received data from ('192.168.1.15', 9060): {"action": "publish", "fname": "a.txt", "lname": "BaoBip.txt", "hostname": "DESKTOP-8H0BART"}
2023-11-26 15:39:26,018 - INFO - Updating client info in database for hostname: DESKTOP-8H0BART
2023-11-26 15:39:26,023 - INFO - Database update complete for hostname: DESKTOP-8H0BART
2023-11-26 15:42:50,357 - INFO - Received data from ('192.168.1.17', 54336): {"action": "fetch", "fname": "a.txt"}
2023-11-26 15:48:48,712 - INFO - Received data from ('192.168.1.17', 54336): {"action": "publish", "fname": "vlan.pdf", "lname": "vlan_tutorial.pdf", "hostname": "Lio-PC"}
2023-11-26 15:48:48,712 - INFO - Updating client info in database for hostname: Lio-PC
2023-11-26 15:48:48,716 - INFO - Database update complete for hostname: Lio-PC
2023-11-26 15:49:26,401 - INFO - Received data from ('192.168.1.17', 54336): {"action": "publish", "fname": "video.mp4", "lname": "2023-11-20-17-05-39.mp4", "hostname": "Lio-PC"}
2023-11-26 15:49:26,402 - INFO - Updating client info in database for hostname: Lio-PC
2023-11-26 15:49:26,403 - INFO - Database update complete for hostname: Lio-PC
2023-11-26 15:51:36,133 - INFO - Received data from ('192.168.1.15', 9060): {"action": "fetch", "fname": "vlan.pdf"}
2023-11-26 15:53:12,627 - INFO - Received data from ('192.168.1.15', 9060): {"action": "fetch", "fname": "video.mp4"}
```

Figure 5: Server ghi nhận yêu cầu từ các Client

6.1.3 Thực hiện các lệnh trên Server

Khi thực hiện **ping hostname**, Server sẽ tiến hành chạy lệnh và trả về kết quả máy có hostname đó có đang online hay không. Tương tự nếu thực hiện lệnh **discover hostname**, Server sẽ gửi yêu cầu tới Client có hostname tương ứng và trả về các file mà hostname đó đang có trong thư mục hiện tại của mình.

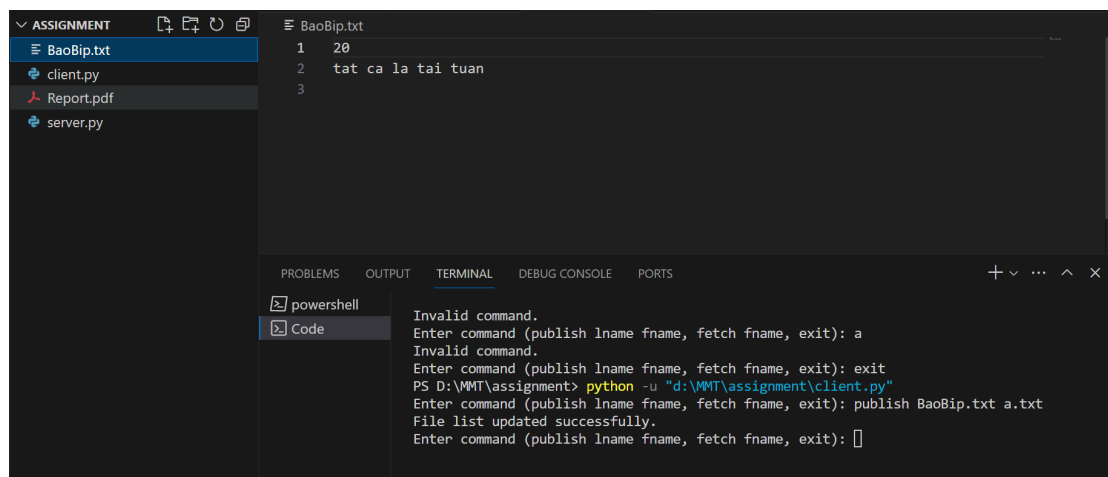
```
Server command: ping Lio-PC
Host Lio-PC is online.
Server command: ping NutPC
Host NutPC is offline.
Server command: discover Lio-PC
Files for Lio-PC: [('vlan.pdf',), ('video.mp4',)]
```

Figure 6: Server thực hiện các lệnh ping và discover

6.2 Client

6.2.1 Thực hiện publish và fetch text file cơ bản

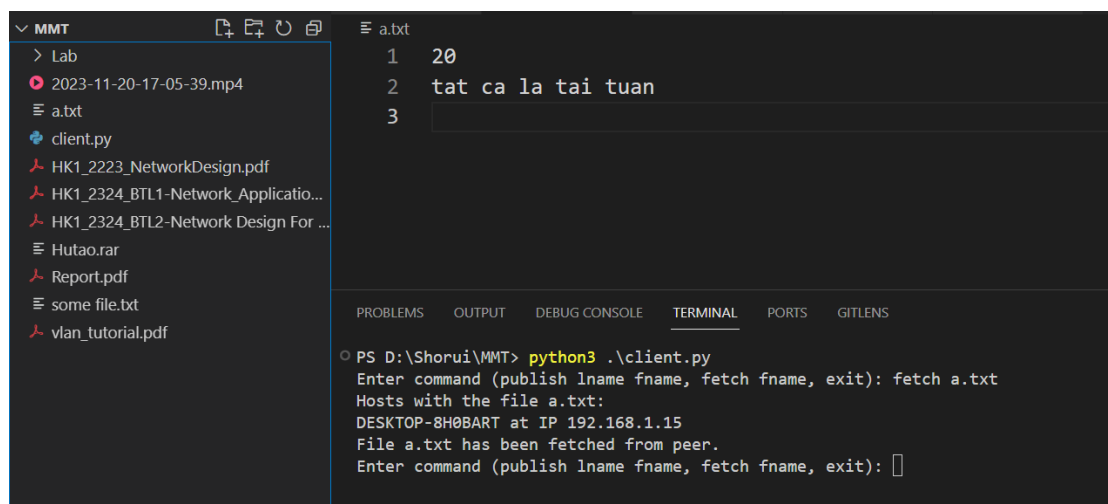
Các Client có thể thực hiện **publish** và **fetch** các file chỉ chứa plain text một cách nhanh chóng mà không xảy ra lỗi.



The screenshot shows a VS Code editor with a file explorer on the left containing 'BaoBip.txt', 'client.py', 'Report.pdf', and 'server.py'. The main editor displays 'BaoBip.txt' with three lines of text: '20', 'tat ca la tai tuan', and an empty line. The terminal at the bottom shows the execution of 'python -u "d:\MMT\assignment\client.py"', which results in a successful publish of 'BaoBip.txt a.txt'.

```
Invalid command.  
Enter command (publish lname fname, fetch fname, exit): a  
Invalid command.  
Enter command (publish lname fname, fetch fname, exit): exit  
PS D:\MMT\assignment> python -u "d:\MMT\assignment\client.py"  
Enter command (publish lname fname, fetch fname, exit): publish BaoBip.txt a.txt  
File list updated successfully.  
Enter command (publish lname fname, fetch fname, exit):
```

Figure 7: Client publish file .txt lên repository của mình



The screenshot shows a VS Code editor with a file explorer on the left containing various files, including 'a.txt'. The main editor displays 'a.txt' with three lines of text: '20', 'tat ca la tai tuan', and an empty line. The terminal at the bottom shows the execution of 'python3 .\client.py', which results in a successful fetch of 'a.txt' from a peer.

```
PS D:\Shorui\MMT> python3 .\client.py  
Enter command (publish lname fname, fetch fname, exit): fetch a.txt  
Hosts with the file a.txt:  
DESKTOP-8H0BART at IP 192.168.1.15  
File a.txt has been fetched from peer.  
Enter command (publish lname fname, fetch fname, exit):
```

Figure 8: Client fetch file .txt từ repository của Client khác về máy mình

6.2.2 Thực hiện publish và fetch các file type đa dạng và các file có kích thước lớn

Ngoài file chỉ chứa plain text, các Client còn có thể **publish** và **fetch** các file có kiểu phức tạp hơn như *.pdf*, *.mp4*,... cũng như các file có kích thước lớn (>100MB).

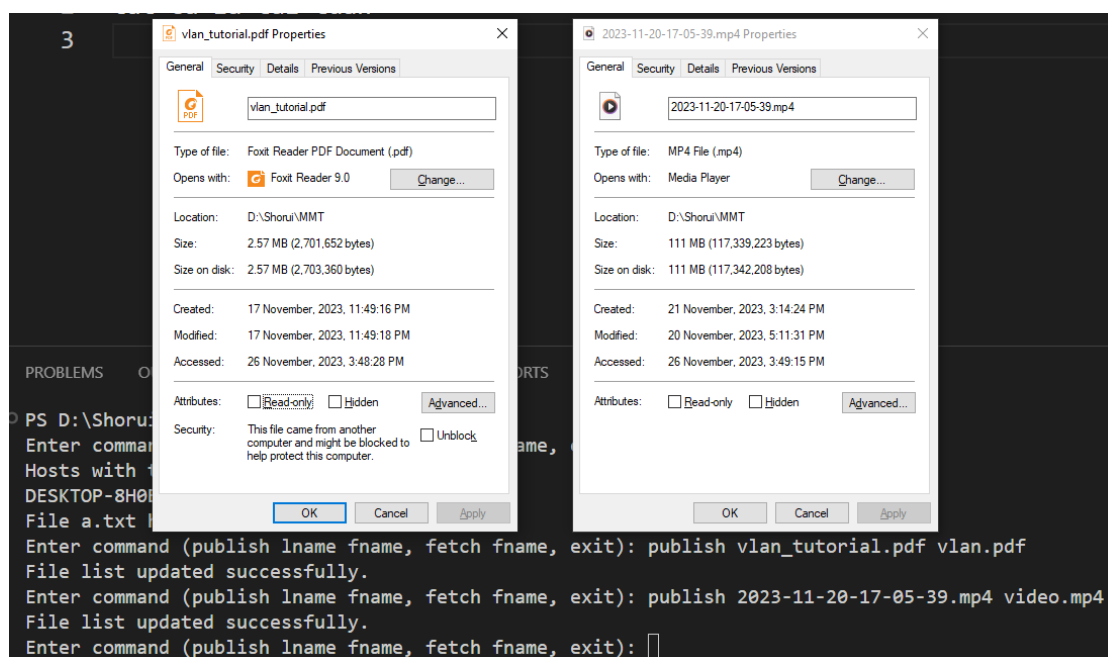


Figure 9: Client publish file .pdf và .mp4 lên repository của mình

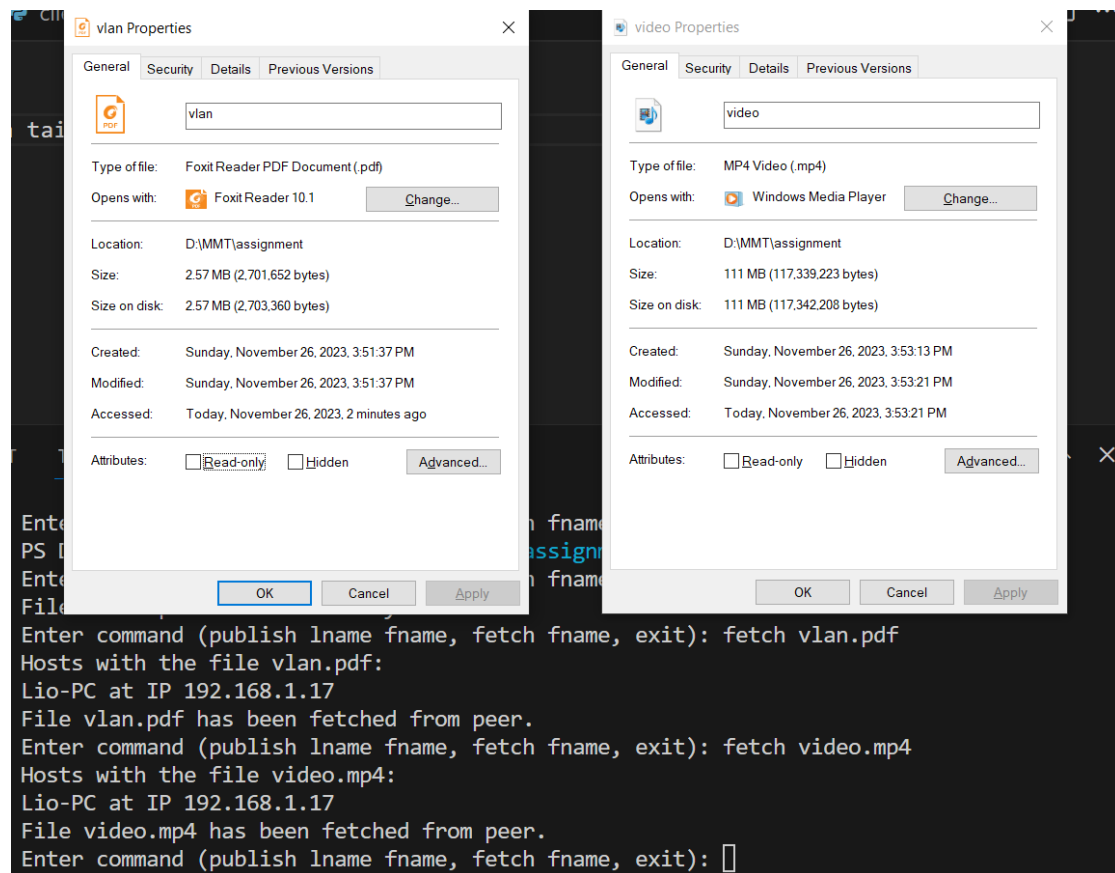


Figure 10: Client fetch file .pdf và .mp4 từ repository của Client khác về máy mình

6.3 Phía cơ sở dữ liệu

Tạo cơ sở dữ liệu MMT và schema **client_files** có bảng để lưu giữ các thông tin liên quan đến **tập tin** (lname, fname, hostname, address) và thực hiện kết nối đến cơ sở dữ liệu. Hệ quản lý cơ sở dữ liệu mà nhóm sử dụng là **PostgreSQL**

```

# Establish a connection to the PostgreSQL database
conn = psycopg2.connect(dbname="MMT", user="postgres", password="061203", host="localhost", port="4000")
cur = conn.cursor()

```

Figure 11: Kết nối đến csdl

```
def update_client_info(hostname, addr, fname, lname):  
    # Update the client's file list in the database  
    cur.execute(  
        f"""INSERT INTO client_files (lname, fname, hostname, address) VALUES (%s, %s, %s, %s)  
        ON CONFLICT (address, fname, hostname) DO UPDATE SET lname = EXCLUDED.lname""",  
        (lname, fname, hostname, addr)  
    )  
    conn.commit()  
  
    # Query the database for the IP addresses of the clients that have the file  
    cur.execute(f"""SELECT DISTINCT ON (address, hostname, fname)  
    address, hostname, lname FROM client_files WHERE fname = %s""", (fname,))  
  
    # Query the database for the file name on database of a specified client  
    cur.execute("SELECT fname FROM client_files WHERE hostname = %s", (hostname,))  
    files = cur.fetchall()
```

Figure 12: Lệnh truy vấn

7 Kết quả đầu ra

Ứng dụng chạy được trên hai hệ điều hành máy tính phổ biến là Window, MacOS. Về phía Server:

- Thực hiện được việc truy vấn, tương tác với cơ sở dữ liệu
- Lưu giữ được thông tin của các client để tiện cho việc xác thực thông tin
- Thực hiện được các lệnh **ping hostname**, **discover hostname**
- Thực hiện được các thao tác truyền, gửi, nhận dữ liệu

Về phía Client:

- Thực hiện được việc truyền, nhận dữ liệu từ client lên máy chủ và ngược lại
- Thực hiện được các lệnh **fetch fname**, **publish lname fname**

8 Đánh giá hiệu năng

Tất cả các chương trình ở 2 phía đều có thể chạy ổn trên cả hai hệ điều hành Window và MacOS.

Về phía server, các lệnh **ping** và **discover** thực hiện và trả về kết quả nhanh chóng, trừ trường hợp lệnh **ping** sử dụng lên một client đang offline thì kết quả trả về sẽ chậm hơn.

Về phía client, lệnh **publish** được hiện thực rất nhanh, tuy nhiên lệnh **fetch** sẽ rất phụ thuộc vào tình trạng mạng và kích thước file. Nhóm đã thử sử dụng Wireshark để kiểm tra thử tình trạng chuyển nhận file, tốc độ đối với các file có dung lượng bé sẽ được chuyển nhận rất nhanh, tuy nhiên với các file dung lượng từ 100Mb đổ lên thì tốc độ còn phụ thuộc vào đường truyền mạng.

Về mặt truy vấn cơ sở dữ liệu, các thao tác CRUD trên cơ sở dữ liệu có tốc độ thao tác nhanh và không xảy ra bất cứ tình trạng chậm trễ nào.

9 Hướng dẫn sử dụng

Người dùng có thể xem Hướng dẫn sử dụng tại [Github repository](#) của nhóm.

10 Kết luận

Thông qua việc hiện thực ứng dụng truyền tải dữ liệu **peer-to-peer file transferring**, nhóm đã học được cách để **lập trình socket** bằng ngôn ngữ Python, cách để truy vấn, lưu trữ các thông tin truyền và nhận vào cơ sở dữ liệu, cách sử dụng **Git** để hợp tác xây dựng phần mềm giữa mọi người trong nhóm.

Sau này, nếu có thời gian, nhóm sẽ phát triển thêm về mặt bảo mật, như là việc mã hoá dữ liệu khi truyền và nhận tập tin, cũng như mã hoá dữ liệu nhạy cảm của người dùng khi lưu vào cơ sở dữ liệu.

Nhóm cũng xin gửi lời cảm ơn đến giáo viên hướng dẫn là thầy Bùi Xuân Giang đã hướng dẫn và trả lời các câu hỏi mà nhóm gặp phải trong quá trình thực hiện ứng dụng.