ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA KHOA HỌC VÀ KỸ THUẬT MÁY TÍNH

----------o0o----------

BÀI TẬP LỚN SỐ 1
MẠNG MÁY TÍNH (CO3093)
XÂY DỰNG MẠNG LIKE-TORRENT

Nhóm NKer

GVHD:	**BÙI XUÂN GIANG**
SVTH:	**MSSV**
Nguyễn Anh Tấn	2114741
Lê Minh Chiến	2112933
Nguyễn Văn Nam	2114121
Trần Đức Mạnh	2114026

TP.HCM,THÁNG 5 NĂM 2024

Mục lục

I MỤC TIÊU VÀ YÊU CẦU 1
II CƠ SỞ LÝ THUYẾT 1
1 Giới thiệu mạng peer to peer (P2P) 1
2 Mô hình giao thức UDP 2
III CÁC CÔNG NGHỆ SỬ DỤNG 4
1 Python 4
2 Socket 4
3 OS… 4
4 Threading 4
5 Pickle 4
IV HIỆN THỰC 5
1 Mô tả ứng dụng 6
2 Hiện thực các chức năng 6
3 Sơ đồ hoạt động 7
4 Cách chạy Project 7
5 Các chức năng 7
5.1 Node (peer) 7
5.2 Tracker 8
6 Kết luận 9
7 Hạn chế 9
Tài liệu tham khảo 9

I MỤC TIÊU VÀ YÊU CẦU

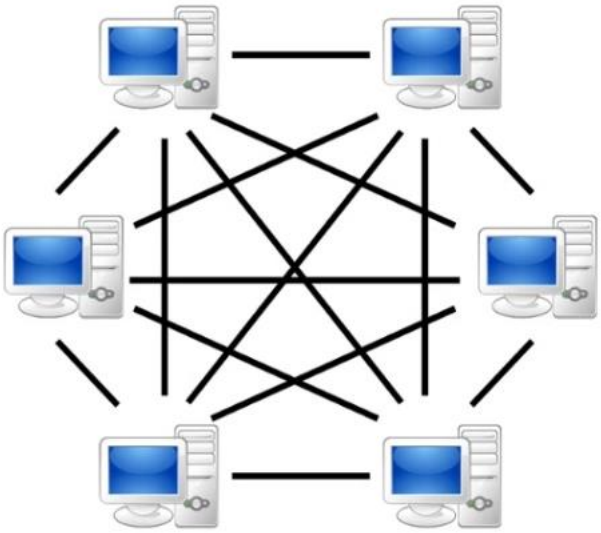
Yêu cầu: Xây dựng một mạng bit-torren đơn giản.

Mục tiêu: Giúp sinh viên hiểu được cách giao tiếp chia sẻ dữ liệu giữa các máy tính.

II CƠ SỞ LÝ THUYẾT

1 Giới thiệu mạng peer to peer (P2P)

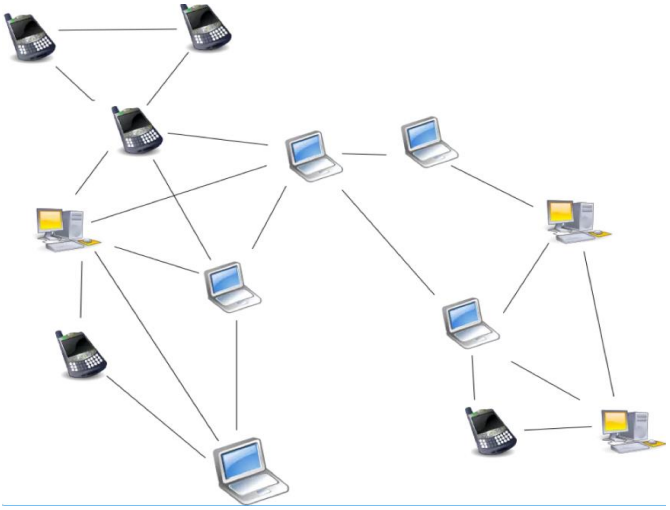
Mạng peer to peer (P2P) là một kiến trúc ứng dụng phân tán nhằm phân vùng nhiệm vụ hoặc khối lượng công việc giữa các peer. Các peer là những thiết bị tham gia trong ứng dụng có đặc quyền như nhau. Chúng tạo thành một mạng lưới các node ngang hàng.



Mô hình P2P

Phân loại:

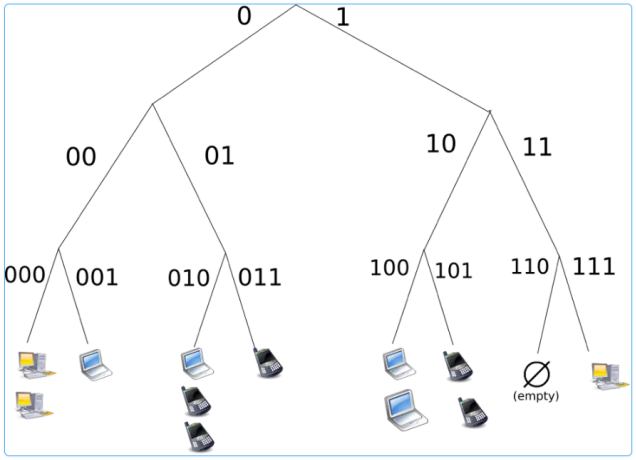
+ Mạng không có cấu trúc:



Mô hình mạng không có cấu trúc

- Mạng ngang hàng peer to peer không có cấu trúc không áp đặt một cấu trúc cụ thể lên mạng lớp phủ theo thiết kế, mà được tạo bởi các node hình thành các kết nối ngẫu nhiên với nhau.
- Vì không có cấu trúc nào được áp đặt trên toàn cầu nên các mạng không có cấu trúc rất dễ xây dựng và cho phép tối ưu hóa bản địa hóa cho các vùng khác nhau của lớp phủ. Ngoài ra bởi vì vai trò của tất cả các peer trong mạng là như nhau, các mạng không có cấu trúc rất mạnh và tốt khi đối mặt tỷ lệ "churn" cao (khi một số lượng lớn các peer thường xuyên tham gia và rời khỏi mạng).
- Do việc thiếu cấu trúc dẫn đến một số hạn chế. Đặc biệt khi một peer muốn tìm một phần dữ liệu mong muốn trong mạng.

+ Mạng có cấu trúc:



Mô hình mạng peer to peer có cấu trúc

- Trong mạng peer to peer có cấu trúc, lớp phủ được tổ chức thành một cấu trúc liên kết cụ thể và giao thức đảm bảo rằng bất kỳ node nào cũng có thể tìm kiếm file hoặc tài nguyên trên mạng một cách hiệu quả ngay cả khi tài nguyên đó cực kỳ hiếm.

+ Mô hình kết hợp:

- Là sự kết hợp của mô hình client-server và mô hình peer to peer. Kiểu mô hình này là sự kết hợp một server trung tâm giúp các peer tìm thấy nhau.

Mô hình peer to peer được ứng dụng trong: chia sẻ tệp tin, truyền phát nội dung đa phương tiện, tìm kiếm thông tin, giao dịch tiền điện tử , kết nối và trao đổi thông tin giữa các thiết bị IoT.

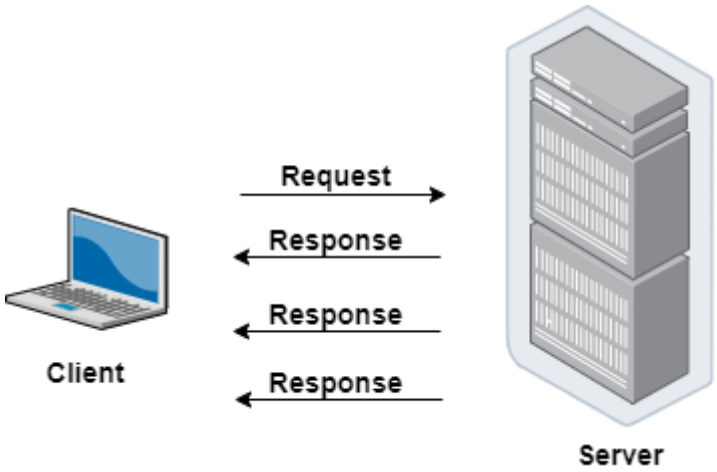
2 Mô hình giao thức UDP

UDP (User Datagram Protocol) là một trong những giao thức cốt lõi của giao thức TCP/IP. Dùng UDP, chương trình trên mạng máy tính có thể gửi những dữ liệu ngắn được gọi là datagram tới máy khác. UDP không cung cấp sự tin cậy và thứ tự truyền nhận mà TCP làm; các gói dữ liệu có thể đến không đúng thứ tự hoặc bị mất mà không có thông báo. Tuy nhiên UDP nhanh và hiệu quả hơn đối với các mục tiêu như kích thước nhỏ và yêu cầu khắt khe về thời gian. Do bản chất không trạng thái của nó nên nó hữu dụng đối với việc trả lời các truy vấn nhỏ với số lượng lớn người yêu cầu.

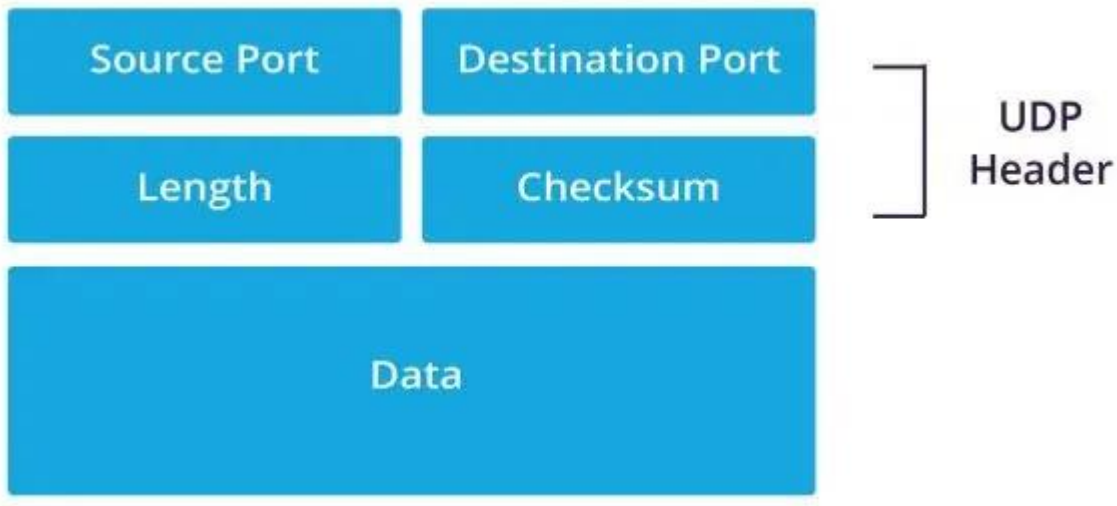
Những ứng dụng phổ biến sử dụng UDP như DNS (Domain Name System), ứng dụng streaming media, Voice over IP, Trivial File Transfer Protocol (TFTP), và game trực tuyến.

UDP hoạt động như thế nào?

Giao thức UDP hoạt động tương tự như TCP, nhưng nó bỏ qua quá trình kiểm tra lỗi. Khi một ứng dụng sử dụng giao thức UDP, các gói tin được gửi cho bên nhận và bên gửi không phải chờ để đảm bảo bên nhận đã nhận được gói tin, do đó nó lại tiếp tục gửi gói tin tiếp theo. Nếu bên nhận bỏ lỡ một vài gói tin UDP, họ sẽ mất vì bên gửi không gửi lại chúng. Do đó thiết bị có thể giao tiếp nhanh hơn.



Cấu trúc UDP Header



- **Source port**: Số cổng của thiết bị gửi. Trường này có thể đặt là 0 nếu máy tính đích đến không cần trả lời người gửi.
- **Destination port**: Số cổng của thiết bị nhận.
- **Length**: Xác định chiều dài của toàn bộ datagram: phần header và dữ liệu. Chiều dài tối thiểu là 8 byte khi gói tin không có dữ liệu, chỉ có header.

Checksum: Kiểm tra lỗi của phần header và dữ liệu. Việc sử dụng checks.

III CÁC CÔNG NGHỆ SỬ DỤNG

1. Python

Python là một ngôn ngữ lập trình mạnh mẽ và linh hoạt, được sử dụng rộng rãi trong việc phát triển các chương trình chia sẻ file. Với Python, bạn có thể xây dựng các ứng dụng cho phép người dùng chia sẻ và truy cập vào các tệp tin và thư mục trên mạng.

Python cung cấp các thư viện và module hữu ích để làm việc với file và mạng như socket, os, threading, shutil, http.server, ftplib và nhiều thư viện khác. Bạn có thể sử dụng các thư viện này để tạo ra các chương trình cho phép người dùng tải lên, tải xuống và chia sẻ file qua giao thức HTTP, FTP hoặc thông qua kết nối socket.

Python cũng hỗ trợ việc xử lý các định dạng file phổ biến như CSV, JSON, XML và nhiều định dạng khác, giúp bạn có thể đọc và ghi dữ liệu từ các file này trong quá trình chia sẻ và truy cập.

Với cú pháp đơn giản, cộng đồng lớn và tài liệu phong phú, Python là một lựa chọn tuyệt vời cho việc lập trình chương trình chia sẻ file. Nó giúp bạn nhanh chóng xây dựng các ứng dụng mạnh mẽ và linh hoạt để quản lý và truy cập vào các tệp tin trên mạng.

Một số thư viện và module mà nhóm đã áp dụng:

2. Socket

Socket là một module quan trọng trong việc phát triển các ứng dụng mạng. Nó cung cấp các công cụ để tạo và quản lý các kết nối mạng, cho phép các ứng dụng gửi và nhận dữ liệu thông qua các giao thức mạng như TCP hoặc UDP.

Module socket cho phép các ứng dụng Python kết nối và tương tác với các ứng dụng khác trên cùng một mạng hoặc trên mạng Internet. Nó cũng cho phép các ứng dụng tạo ra các socket server để lắng nghe và xử lý các yêu cầu từ các ứng dụng khác.

Module socket cung cấp một loạt các hàm và lớp để tạo và quản lý các kết nối mạng. Các hàm này bao gồm socket(), bind(), listen(), accept(), connect(), send() và recv(), giúp việc tạo và quản lý kết nối mạng trở nên dễ dàng hơn.

Với module socket, người dùng có thể tạo ra các ứng dụng mạng đơn giản hoặc phức tạp trên nền tảng Python. Module socket là một công cụ hữu ích cho việc

phát triển các ứng dụng mạng, giúp người dùng tạo ra các ứng dụng đáp ứng được nhu cầu đa dạng của người dùng.

3. OS

Module os trong Python cung cấp các công cụ để tương tác với hệ điều hành. Nó cho phép bạn thực hiện các thao tác như quản lý file và thư mục, thực thi các lệnh hệ thống, xử lý biến môi trường và nhiều hơn nữa.

Khi phát triển các ứng dụng chia sẻ file, module os là một công cụ hữu ích để quản lý các file và thư mục trên hệ điều hành. Bạn có thể sử dụng các hàm như os.listdir() để liệt kê các file và thư mục trong một thư mục, os.path.join() để tạo đường dẫn tuyệt đối cho file hoặc thư mục, os.mkdir() để tạo mới các thư mục, os.remove() để xóa file, và nhiều hàm khác.

Với module os, bạn có thể tạo ra các ứng dụng cho phép người dùng tải lên, tải xuống và chia sẻ file qua giao thức HTTP, FTP hoặc thông qua kết nối socket.

4. Threading

Module threading trong Python cung cấp các công cụ để tạo và quản lý các luồng (threads) trong một chương trình. Với module threading, bạn có thể thực hiện việc chia sẻ file giữa các luồng một cách an toàn và hiệu quả.

Khi phát triển các ứng dụng chia sẻ file, việc sử dụng luồng giúp tăng khả năng đồng thời xử lý và tăng hiệu suất của chương trình. Bằng cách sử dụng module threading, bạn có thể tạo ra các luồng riêng biệt để thực hiện các nhiệm vụ như đọc, ghi hoặc xử lý dữ liệu từ file.

Với module threading, bạn có thể tạo ra các luồng đọc và ghi file song song, giúp tăng tốc độ xử lý và chia sẻ dữ liệu giữa các luồng. Tuy nhiên, khi làm việc với file, cần chú ý đến việc đồng bộ hóa truy cập để tránh xung đột dữ liệu.

Bằng cách sử dụng module này, bạn có thể xây dựng các ứng dụng chia sẻ file đa luồng, đảm bảo tính toàn vẹn và an toàn của dữ liệu.

5. Pickle

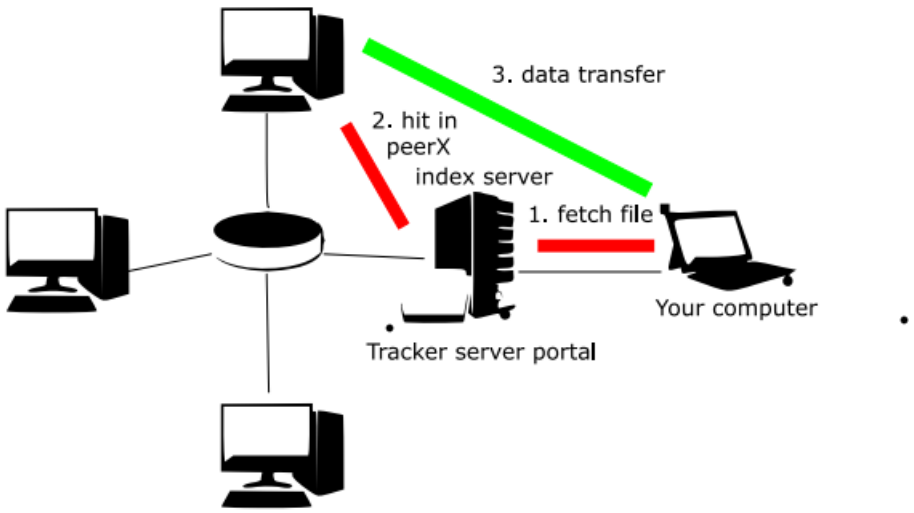
Module pickle trong Python cung cấp các công cụ để chuyển đổi các đối tượng Python thành một chuỗi byte có thể được lưu trữ hoặc truyền tải qua mạng. Công cụ này cho phép bạn lưu trữ dữ liệu phức tạp dưới dạng file và sau đó khôi phục lại chúng khi cần thiết.

Khi phát triển các ứng dụng lưu trữ và truyền tải dữ liệu, module pickle là một công cụ hữu ích để lưu trữ và khôi phục các đối tượng Python. Bạn có thể sử dụng các hàm như pickle.dump() để lưu trữ đối tượng vào một file, pickle.load() để khôi phục đối tượng từ file, và nhiều hàm khác để thực hiện các thao tác khác liên quan đến chuyển đổi dữ liệu.

Với module pickle, bạn có thể lưu trữ và truyền tải các đối tượng Python phức tạp như danh sách, tuple, dictionary, class, function và nhiều đối tượng khác, thuận tiện cho việc chia sẻ file dưới nhiều định dạng khác nhau.

IV HIỆN THỰC

1 Mô tả ứng dụng



Một máy chủ tập trung theo dõi những máy khách nào được kết nối và lưu trữ những tập tin nào.

Máy khách thông báo cho máy chủ biết tập tin nào được lưu trữ trong nơi lưu trữ cục bộ của máy khách.

Khi một máy khách yêu cầu một file thì nó sẽ gửi yêu cầu tìm file đến máy chủ. Máy chủ sẽ xác định máy khách nào đang lưu trữ file cần tìm và gửi thông tin của máy khách đang lưu trữ file cho máy khách yêu cầu file. Máy khách sẽ yêu cầu đến máy khách đang lưu trữ file mà không cần sự can thiệp của máy chủ nữa.

Nhiều máy khách có thể tải xuống nhiều tệp khác nhau từ một máy khách tại một thời điểm.

2 Hiện thực các chức năng

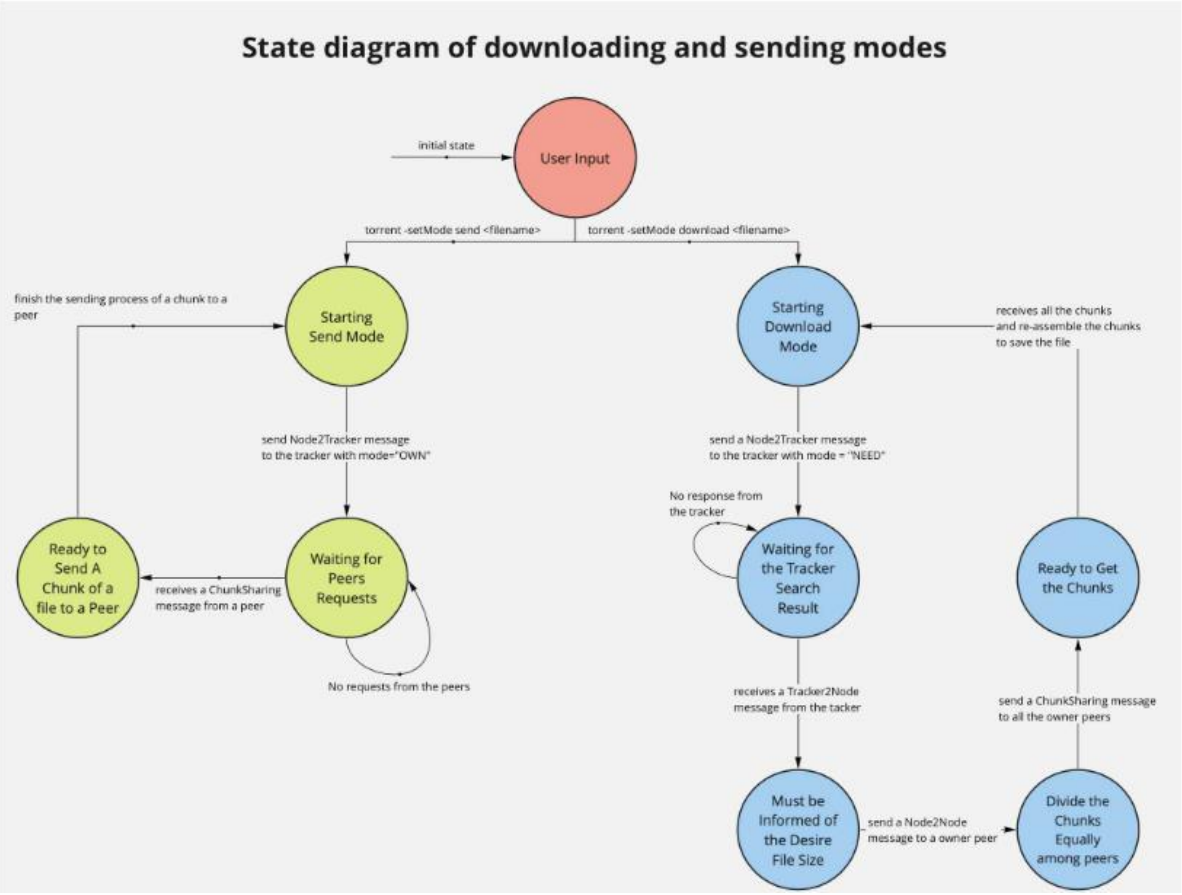
Máy tính Client:

+ Gửi thông báo tới máy tính server về các thông tin mà mình muốn chia sẽ.
+ Khi máy tính phía client cần một file mà trong máy không giữ thì sẽ một yêu cầu đến máy server về thông tin file mà mình đang cần.
+ Sau khi nhận được thông tin về máy client đang giữ nội dung file thì máy tính đang cần file sẽ gửi yêu cầu về file đến máy tính client đang giữ file đó.
+ Khi một máy tính client 1 nhận được thông tin yêu cầu file từ một máy tính client 2 thì nó sẽ chia sẽ file đến client 2.

Máy tính server:

+ Máy tính server sẽ lưu trữ thông tin các file mà các client trong hệ thống mình đang lưu trữ.
+ Cập nhật danh sách khi có máy tính client thông báo có lưu file mới.
+ Khi một máy tính client yêu cầu thông tin một file nào đó thì máy tính server sẽ kiểm tra danh sách mình lưu trữ và trả về thông tin máy tính client đang giữ file đó. Trường hợp file cần tìm không có trong danh sách lưu trữ của máy tính server thì sẽ trả về thông tin "File không tồn tại trong hệ thống".

3. Sơ đồ hoạt động



4. Cách chạy Project

Có hai module chính trong BitTorrent, (1) node hay còn gọi là peer và (2) tracker. Vì vậy, chúng ta phải chạy mỗi module này một cách riêng biệt:

- Với tracker.py:

  $ python tracker.py

- Với node.py: Một ví dụ về việc tạo hai node như sau. (Lưu ý rằng mỗi node phải được chạy trong một cửa sổ terminal riêng biệt nếu đang chạy dự án này trên một máy tính cục bộ)

  $ python3 node.py -node_id 1

  # trong một tab khác của terminal

  $ python3 node.py -node_id 2

5. Các chức năng

5.1 Node (peer)

Mỗi Node sẽ có các chức năng sau:

send (upload): Tại bất kỳ thời điểm nào, một node i có thể muốn tải lên một tập tin trong torrent cho một peer khác. Đầu tiên, node i thông báo cho tracker rằng nó có tệp này trên hệ thống của mình và muốn ở trong trạng thái chờ yêu

cầu của các peer khác cho tập tin cụ thể đó. Một node có thể nhập chế độ này bằng cách nhập như sau:

torrent -setMode send <filename>

download: Nếu node i muốn tải xuống một tập tin, nó phải trước tiên thông báo cho tracker rằng nó cần tập tin này. Sau đó, tracker tìm kiếm tập tin đó trong torrent và sắp xếp các peer sở hữu tập tin này dựa trên danh sách tần suất tải lên của họ, càng nhiều node tải lên, càng có cơ hội được chọn. Sau đó, một số lượng cố định các peer được chọn cho node i để sau này yêu cầu tập tin đó từ họ. Tiếp theo, node i yêu cầu tập tin đó từ những peer láng giềng đó và thực hiện một kết nối UDP để nhận một phần của tập tin từ peer đó.

torrent -setMode download <filename>

exit: Nếu một node muốn rời khỏi torrent nó sẽ gọi hàm để thông báo cho tracker biết rằng một node đã rời khỏi torrent một cách có chủ ý. Nhưng tracker cũng tự động nhận biết rằng một node đã rời khỏi bằng cơ chế duyệt theo với thời gian định kì nếu node đó không gọi exit.

torrent -setMode exit

Mỗi node và cũng như tracker đều có một tệp nhật ký riêng trong thư mục logs/ nơi mọi sự kiện trong torrent liên quan đến node hoặc tracker đó sẽ được ghi lại.

Ngoài ra, có thể thêm bất kỳ tệp nào trong thư mục cục bộ của mỗi node mà tham gia vào torrent. Điều này có thể được tìm thấy trong node_files/.

5.2 Tracker

REGISTER: Tracker nhận loại thông điệp này trong hai trường hợp. Đầu tiên là khi một node gia nhập torrent. Bằng cách này, node thông báo cho tracker biết rằng nó đang trong torrent. Thứ hai, mỗi t giây một node thông báo cho tracker biết rằng nó vẫn còn trong torrent.

OWN: Khi một peer vào chế độ OWN, nó gửi thông điệp này đến tracker. Sau đó, tracker cập nhật cơ sở dữ liệu của mình về các tập tin trong torrent.

NEED: khi một peer cần một tập tin, nó thông báo cho tracker biết rằng nó cần tập tin f. Tracker tìm kiếm trong torrent và sắp xếp các chủ sở hữu của tập tin đó dựa trên một thuật toán ưu tiên những peer có tần suất tải lên cao.

UPDATE: Khi một tập tin đã được gửi bởi một peer đến một node khác, tần suất tải lên của nó phải được tăng. Điều này được thực hiện bởi tracker.

EXIT: Khi một peer thoát khỏi torrent, tất cả thông tin liên quan đến peer này phải được xóa khỏi cơ sở dữ liệu của tracker.

6. Kết luận

- Thông qua việc thiết kế này nhóm dễ dàng hiểu rõ hơn về kiến trúc máy tính peer to peer cũng như cách thức giao tiếp giữa các máy tính.
- Nhóm đã cũng cố thêm kiến thức ngôn ngữ Python.
- Nhóm đã điện thực thành công một hệ thống chia sẻ file đơn giản, hiệu quả.

7. Hạn chế

- Chưa có giao diện trực quan cho người dùng.

Tài liệu tham khảo

[1] Computer Networking: A Top-Down Approach - James F. Kurose , Keith W. Ross.