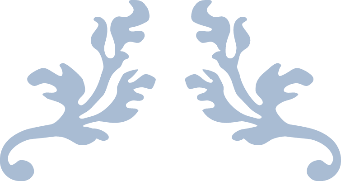


**MẠNG MÁY TÍNH NÂNG CAO**

****

**ĐỒ ÁN**

**Ứng dụng truyền nhận file qua mạng**

## Lê Ngọc Ân - [20424002](mailto:ndhuy@fit.hcmus.edu.vn)

## Dương Mạnh Cường - 20424008

## Phạm Nguyễn Mỹ Diễm – 20424013

## Bùi Tiến Đức - 20424016

## 

## Khoa Công Nghệ Thông Tin

## Đại học Khoa Học Tự Nhiên TP.HCM

Tp. Hồ Chí Minh, tháng 05 năm 2021

**Mục lục**

1. Thông tin nhóm 3
2. Bảng phân công việc 4

3 Mô tả cách thiết kế giao thức, thiết kế hệ thống 5

* 1. [Mô tả cách thiết kế giao thức 5](#_TOC_250001)
  2. [Mô tả cách thiết kế hệ thống 5](#_TOC_250000)

1. Bảng đánh giá chức năng 6

4.1 [Bảng đánh giá 6](#_TOC_250001)

4.2 [Danh sách các chức năng 6](#_TOC_250000)

1. Giao diện – Kết quả 7

5.1 [Master server 7](#_TOC_250001)

5.2 [File server 7](#_TOC_250000)

5.3 [Client 7](#_TOC_250000)

6 Tài liệu tham khảo 8

**1**

**Thông tin nhóm**

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **MSSV** | **Họ tên** | **Email** |
| 1 | 20424002 | Lê Ngọc Ân | [20424002@student.hcmus.edu.vn](mailto:20424002@student.hcmus.edu.vn) |
| 2 | 20424008 | Dương Mạnh Cường | [20424008@student.hcmus.edu.vn](mailto:20424008@student.hcmus.edu.vn) |
| 3 | 20424013 | Phạm Nguyễn Mỹ Diễm | [20424013@student.hcmus.edu.vn](mailto:20424013@student.hcmus.edu.vn) |
| 4 | 20424016 | Bùi Tiến Đức | [20424016@student.hcmus.edu.vn](mailto:20424016@student.hcmus.edu.vn) |

**2**

**Bảng phân công công việc**

|  |  |  |  |
| --- | --- | --- | --- |
| **MSSV** | **Họ tên** | **Nhiệm vụ** | **Mức độ hoàn thành (%)** |
| 20424002 | Lê Ngọc Ân |  |  |
| 20424008 | Dương Mạnh Cường |  |  |
| 20424013 | Phạm Nguyễn Mỹ Diễm |  |  |
| 20424016 | Bùi Tiến Đức |  |  |

**3**

**Mô tả cách thiết kế giao thức, thiết kế hệ thống**

**3.1 Mô tả cách thiết kế giao thức**

* Hệ thống sử dụng hai giao thức chính là TCP và UDP.
* Trong đó, kết nối giữa File Server - Master Server, Client - Master Server sử dụng giao thức TCP.
* Kết nối giữa Client - File Server sử dụng giao thức UDP.

**3.2 Mô tả cách thiết kế hệ thống**

* Hệ thống được xây dựng bằng ngôn ngữ Java 11, gồm 3 class chính là: Client, Master Server (là cầu nối giữa Client và File Server), File Server (nơi lưu trữ dữ liệu để Client có thể tải về).
* Về cách hoạt động, trước tiên sẽ cho Master Server chạy trước, Master Server sử dụng ServerSocket (một class của Java 11 chuẩn) trên một PORT được chỉ định từ phía quản trị viên, chờ các kết nối có thể có từ File Server và Client.
* Sau khi Master Server khởi chạy, ta có thể bắt đầu chạy File Server, File Server sau khi chạy thành công sẽ gửi toàn bộ thông tin các file (kèm theo là IPv4 và PORT để Client có thể truy cập vào File Server tải file) mà nó đang lưu trữ cho Master Server thông qua Socket và PORT được chỉ định của Master Server (một class của Java 11 chuẩn) sau khi gửi thành công File Server sẽ đóng kết nối đến Master Server.
* Sau khi File Server đóng kết nối đến Master Server, quản trị viên có thể tiến hành mở kết nối DatagramSocket (một class của Java 11 chuẩn) tại PORT được chỉ định bởi quản trị viên, chờ các request tải file từ Client.
* Lúc này, khi người dùng chạy Client, trước tiên người dùng sẽ nhận được các thông tin các file (gồm IPv4 và PORT của File Server tương ứng đính kèm) của các File Server đang khởi chạy. Người dùng tiến hành chọn file cần tải bằng ID của file, sau đó Client sẽ gửi một DatagramPacket (một class của Java 11 chuẩn) về tên file cần tải thông qua IPv4 và PORT được đính kèm theo file này đến cho cho File Server.
* File Server do đang mở từ trước, nhận được request từ Client sẽ gửi lại một DatagramPacket chứa FileInfo, gồm các thông tin là tên file này, kích thước, và vì UDP chỉ có thể truyền tối đa 65508 byte trong một DatagramPacket nên ta phải chia file này thành các partition nhỏ, mỗi partition chứa tối đa 1024 \* 32 byte, và lưu vào FileInfo số partition này đồng thời giá trị byte cuối của file sau khi chuyển thành byte array. Cuối cùng FileInfo được gửi tới Client trong DatagramPacket thông qua UDP.
* Tiếp theo File Server sẽ gửi từng partition cho Client.
* Client sau khi nhận FileInfo sẽ truy cập vào partition và nhận từng DatagramPacket, các DatagramPacket này được FileServer gửi đến sẽ cách nhau một khoảng x milisecond (do FileServer định nghĩa, trong hệ thống hiện tại là 100 milisecond) dùng để hạn chế việc Client chưa kịp đọc file và ghi vào BufferedOutputStream (một class của Java 11 chuẩn) thì File Server đã gửi ồ ạt các DatagramPacket đến cho Client.
* Sau khi nhận đủ số partition, Client tiến hành chuyển BufferedOutStream này thành file và tiến hành checksum nó bằng thuật toán SHA-256, nếu match thì Client sẽ đóng kết nối đến File Server, ngược lại nó sẽ gửi lại một request để tại lại file này.

- File Server có thể dừng, trước khi dừng File Server sẽ gửi một request đến Master Server kèm lời nhắn là đóng kết nối, Master Server sau khi nhận lời nhắn này sẽ tiến hành xóa các file liên quan đến File Server này trên vùng nhớ của nó.

**4**

**Bảng đánh giá chức năng**

**4.1 Bảng đánh giá**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **STT** | **Đối tượng** | **Mô tả hệ thống** | **Hoàn thiện (%)** | **Ghi chú** |
| 1 | Master Server | Có địa chỉ IP: port cố định. Chứa thông tin về các file được chia sẻ bởi file server, thông tin IP:port của file server quản lý tương ứng. | 100 |  |
| Cung cấp service **(#1)** để ghi nhận thông tin mà file server gửi lên gồm: danh sách các file, IP:port của file server. Service này sử dụng giao thức TCP tại tầng Transport. | 100 |  |
| Cung cấp service **(#2)** để client có thể lấy thông tin danh sách các file được chia sẻ, kèm theo IP:port của file server quản lý file. Service này sử dụng giao thức TCP tại tầng Transport. | 100 |  |
| Cho phép nhiều client, nhiều file server kết nối tới cùng một thời điểm. | 100 |  |
| 2 | File Server | Chứa các files có thể chia sẻ được với client. |  |  |
| Khi file server khởi động, nó kết nối đến master server và gọi service (1) để gửi thông tin của chính nó lên file server gồm: danh sách file có thể chia sẻ, địa chỉ IP, port mà client có thể kết nối tới để tải file. |  |  |
| Cung cấp service **(#3)** để client tải file với input là tên file cần tải. Service này sử dụng giao thức UDP tại tầng Transport. |  |  |
| Cho phép nhiều client kết nối đến cùng một thời điểm. |  |  |
| Khi file server ngừng hoạt động, master server cần loại bỏ danh sách file của file server tương ứng. |  |  |
| 3 | Client | Có thể sử dụng được service **(2)** do master server cung cấp và service **(3)** do file server cung cấp. |  |  |
| Có thể tải nhiều file cùng một thời điểm, từ nhiều file server khác nhau. |  |  |

**4.2 Danh sách các chức năng**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **STT** | **Tên hàm** | **Chức năng** | **Cách thức hoạt động** | **Hoàn thiện(%)** |
| 1 | FileServer.sendFile() | gửi đến Master Server thông tin các file mà File Server đang có | Kết nối đến Master Server bằng giao thức TCP qua PORT và IPv4 của Master Server, sau khi kết nối thành công thì gửi toàn bộ thông tin các file đính kèm PORT và IPv4 của File Server sau đó đóng kết nối. | 100% |
| 2 | FileServerCloser.run() | đóng DatagramSocket của File Server (class này đã multi-thread) | Dùng giao thức TCP để gửi cho Master Server nội dụng là sẽ đóng DatagramSocket ở phía File Server, sau đó gửi request xong thì DatagramSocket tiến hành đóng và đóng ứng dụng. |  |
| 3 | FileServerShipper.run() | gửi các partition của file mà Client cần tải | FileServerController.run() sau khi nhận một DatagramPacket về file cần tải bởi Client, nó sẽ tạo một FileServerShipper object với parameter là DatagramPacket về thông tin file mà nó Client cần tải, FileServerShipper mở và đọc nội dung trong DatagramPacket này và tìm file mà Client cần, sau đó gửi FileInfo cho Client và tiến hành gửi từng partition bằng DatagramSocket, sau khi gửi đầy đủ các partition nó đóng thread đang chạy |  |
| 4 | FileServerController.run() | nhận các request tải file từ Client thông qua UDP | Nó sẽ chạy một DatagramSocket tại PORT File Server chỉ định, sau khi nhận DatagramPacket từ Client chứa tên file mà Client cần tải, nó tạo một FileServerShipper object và chạy object này trên một luồng riêng biệt. |  |
| 5 | MasterServer.run() | nhận các TCP request từ Client và MasterServer | Chạy object ServerSocket trên PORT chỉ định, nếu request tới là của FileServer, nó gọi phương thức cho FileServer, nếu FileServer chỉ gửi thông tin các file mà FileServer có thì nó sẽ gọi phương thức handleFileClient(), nếu là ngắt kết nối thì nó tiến hành xóa các thông tin file trên vùng nhớ của nó. Nếu request từ Client thì sẽ gửi toàn bộ danh sách file của của FileServer đang chạy cho Client. |  |
| 6 | handleFileServer() | nhận các file đính kèm với IPv4 và PORT từ FileServer | Đọc gói tin thông qua TCP và hiển thị các file này vào vùng nhớ. |  |
| 7 | Client.getFiles() | gửi request đến MasterServer và nhận về danh sách các file đang sống. | Gửi request đến cho FileServer bằng TCP, sau đó nhận về từ FileServer là danh sách các file của các FileServer còn sống. |  |
| 8 | ClientController.run() (class này chạy multi-thread) | gửi cho FileServer tên file cần tải, tải file về. | Gửi tên file cần tải cho FileServer bằng giao thức UDP, sau khi nhận một FileInfo từ FileServer đính kèm trong DatagramPacket, đọc DatagramPacket này và nhận từng partition do FileServer gửi về và bỏ vào BufferedOutputStream, sau khi nhận đủ thì convert byte array này sang file tương ứng, tiến hành checksum với FileInfo, nếu match thì đóng DatagramSocket, nếu không thì gửi lại. |  |

**5**

**Giao diện – Kết quả**

**5.1 Master server**

* abc

**5.2 File server**

**5.3 Client**

**6**

**Tài liệu tham khảo**

* [*https://gist.github.com/absalomhr/ce11c2e43df517b2571b1dfc9bc9b487*](https://gist.github.com/absalomhr/ce11c2e43df517b2571b1dfc9bc9b487)
* [*https://stackoverflow.com/questions/5079172/java-server-multiple-ports*](https://stackoverflow.com/questions/5079172/java-server-multiple-ports)