BỘ GIÁO DỤC VÀ ĐÀO TẠO

TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP. HCM

KHOA CÔNG NGHỆ THÔNG TIN

BÁO CÁO BÀI TẬP LẬP TRÌNH MẠNG

Đề tài: Chương trình sao chép an toàn nhiều file sử dụng SSH và xác thực Public-key

GVHD: Nguyễn Đăng Quang

MÃ HP: NSMS432280

SINH VIÊN THỰC HIỆN:

STT	HỌ VÀ TÊN	MSSV
1	Nguyễn Quang Hùng	22162014
2	Lê Quang Trọng Nghĩa	22162029

TP.HCM, 19/03/2025

Mục Lục

I. PHÂN CÔNG THỰC HIỆN NHIỆM VỤ

1. Nguyễn Quang Hùng - 22162014
    - Nhiệm vụ:
        - Lập trình phần Client (Client.c).
        - Thiết lập môi trường thử nghiệm trên máy cục bộ.
        - Viết báo cáo: [Các phần cụ thể, ví dụ: Hướng dẫn setup môi trường].
2. Lê Quang Trọng Nghĩa - 22162029
    - Nhiệm vụ:
        - Lập trình phần Server (Server.c).
        - Thiết lập môi trường thử nghiệm trên máy từ xa.
        - Viết báo cáo: [Các phần cụ thể, ví dụ: Giải thích code quan trọng].

II. HƯỚNG DẪN SETUP MÔI TRƯỜNG THỰC THI/BIÊN DỊCH

1. Yêu cầu phần mềm

- Hệ điều hành: Kali linux
- Trình biên dịch: GCC
- Thư viện: OpenSSH, libssh,....

2. Cài đặt môi trường

- Trên máy cục bộ (Client):
    - Cài OpenSSH:
        - sudo apt update
        - sudo apt install openssh-client

    - Cài libssh:
        - sudo apt install libssh-dev
- Trên máy từ xa (Server):
    - [Ví dụ: Cài OpenSSH Server: sudo apt install openssh-server].

3. Biên dịch chương trình

- Client: gcc -o client Client.c -lssh
- Server: gcc -o server Server.c -lssh

4. Cấu hình chạy chương trình

- Chạy Server: ./server
- Chạy Client: ./Client.c push test /home/qhung

III. TRÌNH BÀY GIẢI PHÁP THỰC HIỆN THEO TIẾP CẬN TOP-DOWN REFINEMENT

1. Mức cao nhất

- Mục tiêu: Xây dựng hệ thống Client-Server sao chép an toàn nhiều file từ máy cục bộ (Kali) sang máy từ xa (Ubuntu) qua SSH, sử dụng xác thực public key.

2. Tinh chỉnh cấp 1

- Client: Quét thư mục nguồn, thiết lập kết nối SSH, gửi từng file qua SCP.
- Server: Thiết lập máy chủ SSH, nhận và lưu file vào thư mục đích.

3. Tinh chỉnh cấp 2

- Hiển thị thông tin kết nối, trạng thái xác thực, và thông tin sao chép (tên file, kích thước, tiến trình, thời gian).
- Ghi log chi tiết các sự kiện trên server vào file /home/qhung/ssh_server.log.

4. Tinh chỉnh cấp 3

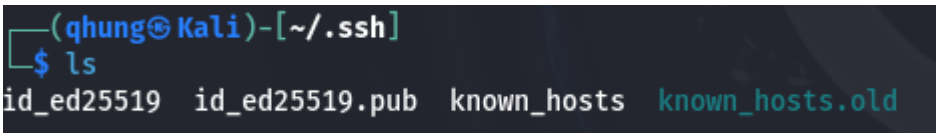
- Xác thực public key để đảm bảo an toàn.
- Tối ưu truyền dữ liệu bằng cách đọc/ghi file theo buffer (4096 bytes).
- Đo thời gian sao chép để đánh giá hiệu suất.

IV. MÔ TẢ CÁC BƯỚC THỰC HIỆN

1. Client.c

- Bước 1: Khởi tạo khóa

    ssh-keygen -t ed25519



Truyền khóa public qua server để xác thực sử dụng public key

- Bước 2: Khởi tạo và thiết lập

Thư viện và Biến: Code bao gồm các thư viện cần thiết (libssh, stdio, stdlib, v.v.) để sử dụng SSH, SCP, SFTP và xử lý tệp/thư mục. Các hàm tiện ích như get_file_size và is_directory được định nghĩa để hỗ trợ.

Kiểm tra Đối số: Trong main, chương trình kiểm tra đối số dòng lệnh. Nếu không đủ (dưới 4 đối số) hoặc không bắt đầu bằng "push", nó sẽ báo lỗi và thoát.

- Bước 3:Thiết lập Kết nối SSH

Tạo Phiên SSH:

ssh_new() tạo một phiên SSH mới (my_ssh_session).

Nếu thất bại, báo lỗi và thoát.

Cấu hình Phiên:

Thiết lập thông tin kết nối: địa chỉ máy chủ (192.168.189.128), cổng (2222), tên người dùng (qhung), và đường dẫn khóa riêng (/home/qhung/.ssh/id_ed25519) bằng ssh_options_set.

Kết nối đến Máy chủ:

ssh_connect() thực hiện kết nối. Thông báo "Attempting to connect..." trước khi kết nối và "Connection established..." nếu thành công. Nếu thất bại, báo lỗi và thoát.

Xác thực:

ssh_userauth_publickey_auto() xác thực bằng khóa công khai tự động. Nếu thành công, thông báo "Successfully connected and authenticated..."; nếu không, báo lỗi và thoát.

- Bước 4: Xử lý truyền tệp (Push)

a. Đẩy một tệp

Tạo Phiên SSH:

- ssh_new() tạo một phiên SSH mới (my_ssh_session).
- Nếu thất bại, báo lỗi và thoát.

Cấu hình Phiên:

- Thiết lập thông tin kết nối: địa chỉ máy chủ (192.168.189.128), cổng (2222), tên người dùng (qhung), và đường dẫn khóa riêng (/home/qhung/.ssh/id_ed25519) bằng ssh_options_set.

Kết nối đến Máy chủ:

- ssh_connect() thực hiện kết nối. Thông báo "Attempting to connect..." trước khi kết nối và "Connection established..." nếu thành công. Nếu thất bại, báo lỗi và thoát.

Xác thực:

- ssh_userauth_publickey_auto() xác thực bằng khóa công khai tự động. Nếu thành công, thông báo "Successfully connected and authenticated..."; nếu không, báo lỗi và thoát.

b. Đẩy nhiều tệp trong thư mục

Mở Thư mục:

- Dùng opendir để mở thư mục cục bộ. Nếu thất bại, báo lỗi.

Quá trình Đẩy:

- Thông báo "Starting to push files...".
- Duyệt qua từng mục trong thư mục bằng readdir.
- Bỏ qua "." và ".." (thư mục hiện tại và thư mục cha).

- Với mỗi mục, kiểm tra bằng is_directory. Nếu là tệp (không phải thư mục), tạo đường dẫn cục bộ và từ xa, sau đó gọi push_file.
- Đếm số tệp được đẩy (file_count).

Kết thúc:

- Đo tổng thời gian bằng clock().
- Đóng thư mục và thông báo "Completed pushing..." với số tệp và tổng thời gian.

c. Xử lý tệp hoặc thư mục

Tạo Thư mục Từ xa:

- Gọi create_remote_dir để đảm bảo thư mục đích từ xa tồn tại (dùng SFTP).

Phân loại và Xử lý:

- Nếu là thư mục (is_directory trả về true), gọi push_files_in_dir.
- Nếu là tệp, gọi push_file.

d. Hàm main

Duyệt Đối số:

- Lấy thư mục đích từ xa từ đối số cuối cùng (destination).
- Duyệt qua các mục cần đẩy (từ argv[2] đến argv[argc-2]).

Tạo Đường dẫn:

- Nếu mục là thư mục, dùng destination làm đường dẫn từ xa.
- Nếu là tệp, thêm tên tệp vào destination.

Thực thi:

- Gọi push_item cho từng mục. Nếu thất bại, báo lỗi.

Bước 5: Ngắt kết nối

Ngắt Kết nối:

- Thông báo "Disconnecting from server...".
- Gọi ssh_disconnect để ngắt kết nối SSH.

Giải phóng Bộ nhớ:

- Gọi ssh_free để giải phóng phiên SSH.
- Thông báo "Disconnected successfully.".

2. Server.c

- Bước 1: Thiết lập máy chủ SSH.
    - Khởi tạo ssh_bind và thiết lập các tùy chọn (cổng 2222, khóa host).
    - Lắng nghe kết nối từ client.
- Bước 2: Xác thực client bằng public key.
    - Sử dụng authenticate_user để kiểm tra khóa công khai của client so với authorized_keys.
    - Ghi log chi tiết vào /home/qhung/ssh_server.log.
- Bước 3: Nhận và lưu file qua SCP.
    - Sử dụng handle_scp để nhận nhiều file từ client.
    - Lưu file vào thư mục /home/qhung/.
    - Hiển thị thông tin: tên file, kích thước, tiến trình sao chép, và thời gian hoàn tất.
- Bước 4: Ghi log và đóng kết nối.

- Ghi log các sự kiện quan trọng (kết nối, xác thực, nhận file).
- Đóng channel và ngắt kết nối sau khi hoàn tất.

V. GIẢI THÍCH CÁC PHẦN CODE QUAN TRỌNG

Client:

Server:

1. Thiết lập máy chủ SSH và lắng nghe kết nối (main)

```
printf("Starting SSH server...\n");
sshbind = ssh_bind_new();
if (sshbind == NULL) {
    write_log("Lỗi khi tạo SSH bind");
    printf("Error creating sshbind\n");
    exit(1);
}
printf("sshbind created\n");

ssh_bind_options_set(sshbind, SSH_BIND_OPTIONS_LOG_VERBOSITY, &verbosity);
ssh_bind_options_set(sshbind, SSH_BIND_OPTIONS_BINDPORT, &port);
ssh_bind_options_set(sshbind, SSH_BIND_OPTIONS_HOSTKEY, KEYS_FOLDER "ssh_server_key");

if (ssh_bind_listen(sshbind) < 0) {
    snprintf(log_msg, sizeof(log_msg), "Lỗi khi lắng nghe socket: %s", ssh_get_error(sshbind));
    write_log(log_msg);
    printf("Bind error: %s\n", ssh_get_error(sshbind));
    ssh_bind_free(sshbind);
    exit(1);
}
snprintf(log_msg, sizeof(log_msg), "SSH server đang lắng nghe trên cổng %d", port);
write_log(log_msg);
printf("Server listening on port %d\n", port);
```

- Ý nghĩa:

Đoạn code này chịu trách nhiệm thiết lập máy chủ SSH và bắt đầu lắng nghe kết nối từ client.

- ssh_bind_new() tạo một đối tượng ssh_bind để quản lý kết nối server. Nếu thất bại, ghi log lỗi và thoát chương trình.
- ssh_bind_options_set() thiết lập các tùy chọn cho server:

- SSH_BIND_OPTIONS_LOG_VERBOSITY: Đặt mức độ chi tiết của log (ở đây là SSH_LOG_PROTOCOL).
- SSH_BIND_OPTIONS_BINDPORT: Đặt cổng lắng nghe là 2222 (biến port).
- SSH_BIND_OPTIONS_HOSTKEY: Chỉ định đường dẫn đến khóa host của server (/home/qhung/ssh_server_key).

- ssh_bind_listen() bắt đầu lắng nghe kết nối từ client trên cổng 2222. Nếu thất bại, ghi log lỗi và thoát.
- Sau khi thiết lập thành công, ghi log và in thông báo server đang lắng nghe. Đoạn code này là bước đầu tiên để server sẵn sàng nhận kết nối từ client, đảm bảo server hoạt động trên cổng 2222 với khóa host được chỉ định.

2. Xác thực client bằng public key (authenticate_user)

```
if (ssh_message_type(message) == SSH_REQUEST_AUTH) {
    if (ssh_message_subtype(message) == SSH_AUTH_METHOD_PUBLICKEY) {
        ssh_key client_pubkey = ssh_message_auth_pubkey(message);
        if (client_pubkey == NULL) {
            snprintf(log_msg, sizeof(log_msg), "Không thể lấy khóa công khai từ client");
            write_log(log_msg);
            ssh_message_auth_set_methods(message, SSH_AUTH_METHOD_PUBLICKEY);
            ssh_message_reply_default(message);
            ssh_message_free(message);
            continue;
        }

        int rc = ssh_pki_import_pubkey_file("/home/qhung/.ssh/authorized_keys", &pubkey);
        if (rc != SSH_OK) {
            snprintf(log_msg, sizeof(log_msg), "Lỗi khi đọc authorized_keys: %s", ssh_get_error(session));
            write_log(log_msg);
            ssh_message_auth_set_methods(message, SSH_AUTH_METHOD_PUBLICKEY);
            ssh_message_reply_default(message);
            ssh_key_free(client_pubkey);
            ssh_message_free(message);
            continue;
        }

        if (ssh_key_cmp(client_pubkey, pubkey, SSH_KEY_CMP_PUBLIC) == 0) {
            snprintf(log_msg, sizeof(log_msg), "Xác thực khóa công khai thành công cho user: %s", username);
            write_log(log_msg);
            ssh_message_auth_reply_success(message, 0);
            ssh_key_free(pubkey);
            ssh_key_free(client_pubkey);
            ssh_message_free(message);
            return SSH_AUTH_SUCCESS;
```

- Ý nghĩa:

Đoạn code này xử lý quá trình xác thực client bằng phương thức public key, đảm bảo chỉ client có khóa hợp lệ mới được phép kết nối.

- ssh_message_type() và ssh_message_subtype() kiểm tra xem tin nhắn từ client có phải là yêu cầu xác thực bằng public key (SSH_AUTH_METHOD_PUBLICKEY) hay không.
- ssh_message_auth_pubkey() lấy khóa công khai từ client. Nếu không lấy được, ghi log lỗi và yêu cầu client thử lại.
- ssh_pki_import_pubkey_file() đọc danh sách khóa công khai từ file /home/qhung/.ssh/authorized_keys trên server.
- ssh_key_cmp() so sánh khóa công khai của client với khóa trong authorized_keys. Nếu khớp, xác thực thành công, ghi log và trả về SSH_AUTH_SUCCESS.
- Nếu xác thực thất bại, ghi log và yêu cầu client thử lại phương thức xác thực khác.

  Đoạn code này đảm bảo tính bảo mật bằng cách chỉ cho phép client có khóa công khai hợp lệ kết nối với server.

3. Khởi tạo phiên SCP để nhận file (handle_scp)

```
scp = ssh_scp_new(session, SSH_SCP_READ, "/home/qhung/uploaded_file");
if (scp == NULL) {
    snprintf(log_msg, sizeof(log_msg), "Lỗi tạo SCP session: %s", ssh_get_error(session));
    write_log(log_msg);
    return -1;
}
write_log("SCP session được tạo");

rc = ssh_scp_init(scp);
if (rc != SSH_OK) {
    snprintf(log_msg, sizeof(log_msg), "Lỗi khởi tạo SCP: %s", ssh_get_error(session));
    write_log(log_msg);
    ssh_scp_free(scp);
    return -1;
}
write_log("Khởi tạo SCP thành công");

write_log("Chờ yêu cầu SCP từ client");
rc = ssh_scp_pull_request(scp);
if (rc != SSH_SCP_REQUEST_NEWFILE) {
    snprintf(log_msg, sizeof(log_msg), "Không nhận được yêu cầu file mới: rc=%d", rc);
    write_log(log_msg);
    ssh_scp_close(scp);
    ssh_scp_free(scp);
    return -1;
}
```

- Ý nghĩa:

Đoạn code này khởi tạo một phiên SCP để nhận file từ client.

- ssh_scp_new() tạo một phiên SCP với chế độ SSH_SCP_READ, chỉ định đường dẫn lưu file là /home/qhung/uploaded_file. Nếu thất bại, ghi log lỗi và thoát.
- ssh_scp_init() khởi tạo phiên SCP để sẵn sàng nhận dữ liệu. Nếu thất bại, ghi log và giải phóng tài nguyên.
- ssh_scp_pull_request() kiểm tra yêu cầu từ client. Nếu nhận được SSH_SCP_REQUEST_NEWFILE, server biết client đang gửi một file mới. Nếu không, ghi log lỗi và thoát.

  Đoạn code này là bước đầu tiên trong quy trình nhận file qua SCP, đảm bảo server sẵn sàng nhận dữ liệu từ client.

4. Nhận và lưu dữ liệu file (handle_scp)

```
long file_size = ssh_scp_request_get_size(scp);
char *filename = strdup(ssh_scp_request_get_filename(scp));
int permissions = ssh_scp_request_get_permissions(scp);
snprintf(log_msg, sizeof(log_msg), "Nhận yêu cầu file: %s, kích thước: %ld bytes, quyền: %o",
         filename, file_size, permissions);
write_log(log_msg);

ssh_scp_accept_request(scp);
write_log("Đã chấp nhận yêu cầu SCP");

FILE *file = fopen("/home/qhung/uploaded_file", "wb");
if (file == NULL) {
    snprintf(log_msg, sizeof(log_msg), "Lỗi mở file để ghi: %s", strerror(errno));
    write_log(log_msg);
    free(filename);
    ssh_scp_close(scp);
    ssh_scp_free(scp);
    return -1;
}
write_log("Mở file để ghi thành công");

char buffer[4096];
size_t total_received = 0;
int nbytes;
write_log("Bắt đầu nhận dữ liệu file");
while (total_received < file_size && (nbytes = ssh_scp_read(scp, buffer, sizeof(buffer))) > 0) {
    fwrite(buffer, 1, nbytes, file);
    total_received += nbytes;
    snprintf(log_msg, sizeof(log_msg), "Nhận %d bytes (tổng: %zu/%ld)", nbytes, total_received, file_size);
    write_log(log_msg);
}
```

- Ý nghĩa:

Đoạn code này xử lý việc nhận và lưu dữ liệu file từ client.

- ssh_scp_request_get_size(), ssh_scp_request_get_filename(), và ssh_scp_request_get_permissions() lấy thông tin về file (kích thước, tên, quyền truy cập) từ yêu cầu SCP.
- Ghi log thông tin file để theo dõi.
- ssh_scp_accept_request() chấp nhận yêu cầu SCP để bắt đầu nhận dữ liệu.
- fopen() mở file /home/qhung/uploaded_file ở chế độ ghi nhị phân (wb).

- ssh_scp_read() đọc dữ liệu từ SCP session vào buffer (kích thước 4096 bytes).
- fwrite() ghi dữ liệu từ buffer vào file trên server.
- total_received theo dõi tổng số byte đã nhận, đảm bảo nhận đủ dữ liệu theo kích thước file (file_size).
- Ghi log tiến trình nhận dữ liệu để theo dõi.

  Đoạn code này đảm bảo dữ liệu file được nhận và lưu trữ chính xác trên server.

5. Ghi log các sự kiện (write_log)

```
void write_log(const char *message) {
    FILE *log_file = fopen("/home/qhung/ssh_server.log", "a");
    if (log_file == NULL) {
        perror("Không thể mở tệp log");
        return;
    }
    time_t now;
    time(&now);
    char *time_str = ctime(&now);
    time_str[strlen(time_str) - 1] = '\0';
    fprintf(log_file, "[%s] %s\n", time_str, message);
    fclose(log_file);
}
```

- Ý nghĩa:

Hàm này ghi lại các sự kiện quan trọng của server vào file log.

- fopen() mở file /home/qhung/ssh_server.log ở chế độ append (a) để thêm thông tin mới mà không ghi đè dữ liệu cũ.
- time() và ctime() lấy thời gian hiện tại để thêm vào log, giúp xác định thời điểm xảy ra sự kiện.
- fprintf() ghi thông điệp log với định dạng [thời gian] thông điệp.
- fclose() đóng file log sau khi ghi để tránh rò rỉ tài nguyên.
  Hàm này hỗ trợ việc theo dõi và debug bằng cách ghi lại toàn bộ hoạt động của server, từ kết nối, xác thực đến nhận file.

VI. KẾT QUẢ BIÊN DỊCH VÀ CHẠY CHƯƠNG TRÌNH

1. Biên dịch

- Trên Client:

```
┌──(qhung㉿Kali)-[~/C-Network-Programming/Chap03/SSH_Lab_01/ssh_client]
└─$ gcc scp_client.c -o scp_client -lssh
```

- Trên Server

```
┌──(qhung㉿Kali)-[~/C-Network-Programming/Chap03/SSH_Lab_01/ssh_server]
└─$ gcc ssh_scp_server.c -o ssh_scp_server -lssh -lpthread
```

2. Chạy chương trình

- Trên Client:

```
┌──(qhung㉿Kali)-[~/C-Network-Programming/Chap03/SSH_Lab_01/ssh_client]
└─$ ls
example.txt              pull_test1.txt           scp_test.txt      ssh_client_key_interactive
local_file.txt           remote_port_forwarding   ssh_client        ssh_client_key_interactive.c
local_port_forwarding    remote_port_forwarding.c ssh_client.c      ssh_scp_client
local_port_forwarding.c  scp_client               ssh_client_key    ssh_scp_client.c
pull_test.txt            scp_client.c             ssh_client_key.c  test

┌──(qhung㉿Kali)-[~/C-Network-Programming/Chap03/SSH_Lab_01/ssh_client]
└─$ cd test

┌──(qhung㉿Kali)-[~/…/Chap03/SSH_Lab_01/ssh_client/test]
└─$ ls
local1.txt  local2.txt
```

```
┌──(qhung㉿Kali)-[~/C-Network-Programming/Chap03/SSH_Lab_01/ssh_client]
└─$ ./scp_client push test /home/qhung/
Attempting to connect to 192.168.189.128:22 as user qhung...
Connection established. Authenticating...
Successfully connected and authenticated to 192.168.189.128:22 as qhung!
Starting to push files from test to /home/qhung/
Starting transfer: test/local2.txt (Size: 18 bytes)
Transferring test/local2.txt: 18/18 bytes completed
Completed transfer: test/local2.txt to /home/qhung//local2.txt (Size: 18 bytes, Time: 0.001 seconds)
Starting transfer: test/local1.txt (Size: 16 bytes)
Transferring test/local1.txt: 16/16 bytes completed
Completed transfer: test/local1.txt to /home/qhung//local1.txt (Size: 16 bytes, Time: 0.000 seconds)
Completed pushing 2 files from test to /home/qhung/ (Total time: 0.004 seconds)
Disconnecting from server...
Disconnected successfully.
```

Trên server đã có 2 file local1.txt và local2.txt

```
┌──(qhung㉿Kali)-[~]
└─$ ls
8821au-20210708          linux-headers-6.6.15-amd64_6.6.15-2kali1+b1_amd64.de
C-Network-Programming    linux-headers-6.6.15-common_6.6.15-2kali1_all.deb
Desktop                  linux-kbuild-6.6.15_6.6.15-2kali1+b1_amd64.deb
Documents                local1.txt
Downloads                local2.txt
```

- Trên Server:

Starting SSH server...

sshbind created

Server listening on port 2222

New connection accepted

Authentication successful

Receiving file1.txt (13 bytes)

Progress: 13/13 bytes

Completed: file1.txt (13 bytes) in 0.01 seconds

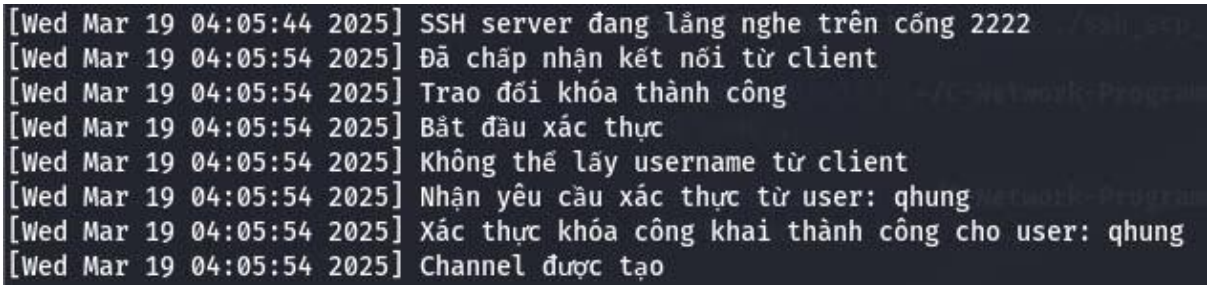
Receiving file2.txt (13 bytes)

Progress: 13/13 bytes

Completed: file2.txt (13 bytes) in 0.01 seconds

All files received

- log (/home/qhung/ssh_server.log)



3. Đánh giá

- Chương trình chạy ổn định, đáp ứng yêu cầu sao chép an toàn nhiều file qua SSH.
- Hiển thị đầy đủ thông tin về tên file, kích thước, tiến trình, và thời gian sao chép.
- Log chi tiết giúp dễ dàng theo dõi và debug.

VII. KẾT LUẬN

- Thành tựu:
    - Hoàn thành hệ thống Client-Server sao chép an toàn nhiều file qua SSH với xác thực public key.
    - Server hiển thị đầy đủ thông tin về file nhận được và ghi log chi tiết.
- Đề xuất:
    - Chuyển sang sử dụng SFTP thay vì SCP để tránh cảnh báo deprecated và hỗ trợ tốt hơn cho các tính năng nâng cao.
    - Thêm tính năng nén file trước khi truyền để giảm thời gian sao chép.
    - Thêm kiểm tra lỗi và xử lý ngoại lệ chi tiết hơn (ví dụ: kiểm tra quyền thư mục, dung lượng đĩa).