**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**

**KHOA MẠNG MÁY TÍNH VÀ TRUYỀN THÔNG**

A blue logo with a black background

Description automatically generated

**BÁO CÁO THỰC HÀNH**

**LAB 4: PKI AND HASH FUNCTIONS**

**MÔN HỌC: MẬT MÃ HỌC**

**Họ và tên: LẠI QUAN THIÊN**

**Mã số sinh viên: 22521385**

**Lớp: NT219.O21.ANTT**

**TP. HỒ CHÍ MINH, THÁNG 07 NĂM 2024**

**MỤC LỤC**

[PHẦN I: TỔNG QUAN VÀ MÔ TẢ 2](#_Toc171035851)

[1.1. Thông tin cá nhân: 2](#_Toc171035852)

[1.2. Thông tin thiết bị: 2](#_Toc171035853)

[PHẦN II: NỘI DUNG THỰC HÀNH 2](#_Toc171035854)

[2.1. Hash Functions: 2](#_Toc171035855)

[2.1.1. Hash Functions GUI 2](#_Toc171035856)

[2.1.2. Hash Functions CLI 3](#_Toc171035857)

# PHẦN I: TỔNG QUAN VÀ MÔ TẢ

## 1.1. Thông tin cá nhân:

Họ và tên: Lại Quan Thiên

Mã số sinh viên: 22521385

Lớp thực hành: NT219.O21.ANTT.1

Link github: [Cryptography-Course/Labs/OffClass/Lab\_1 at main · WanThinnn/Cryptography-Course (github.com)](https://github.com/WanThinnn/Cryptography-Course/tree/main/Labs/OffClass/Lab_1)

## 1.2. Thông tin thiết bị:

**- Thiết bị:** Macbook Air 2019 – RAM 8GB (LPDDR3 2133MHz) – SSD 128GB

**- Hệ điều hành:**

+ Windows 11 Pro 21H2 (cài đặt thông qua Bootcamp của Apple)

+ Ubuntu 22.04 Jammy Jellyfish (cài đặt thông qua [t2linux.org](https://t2linux.org/))

**- Bộ xử lý:**

+ Intel Core i5 8210Y – 1.60GHz – Turbo Boost 3.60Ghz

+ Intel UHD Graphics 617

+ Apple T2 Security Chip

**- Thông tin về bộ xử lý:**

+ [Intel Core i58210Y Processor 4M Cache up to 3.60 GHz Thông số kỹ thuật sản phẩm](https://ark.intel.com/content/www/vn/vi/ark/products/189912/intel-core-i5-8210y-processor-4m-cache-up-to-3-60-ghz.html)

+ [Hỗ trợ Intel® UHD Graphics 617](https://www.intel.vn/content/www/vn/vi/support/products/189913/graphics/processor-graphics/intel-uhd-graphics-family/intel-uhd-graphics-617.html)

+ [Apple T2 Security Chip: Security Overview](https://www.apple.com/jp/mac/docs/Apple_T2_Security_Chip_Overview.pdf)

- Các thực nghiệm được thực thi trong quá trình laptop cắm điện, nhiệt độ phòng

A screenshot of a computer

Description automatically generated

Ảnh 1: Thông tin thiết bị

**1.3. Tổng quan:**

Bài báo cáo gồm các Task 1, 2, 3, 4 của Lab 4. Các chương trình được code bằng ngôn ngữ C++ và sử dụng thư viện CryptoPP, OpenSSL.

- Task 1:

+ Đối với CLI: sau khi xây dựng code xong thì tiến hành tạo 5 file test với kích thước khác nhau từ MB đến GB, thực hiện đo thời gian 1000 Băm trên cả hai Hệ điều hành Windows và Linux Viết bảng thống kê số liệu và vẽ biểu đồ phân tích và so sánh.

+ Đối với GUI: xuất file dll để build GUI trên Winform.

- Task 2: Sau khi xây dựng xong chương trình CLI App, tiến hành xuất file dll để build GUI trên Winform.

- Task 3,4: Sử dụng HashClash và HashPump để thực hành trên Ubuntu 22.04

# PHẦN II: NỘI DUNG THỰC HÀNH

## 2.1. Hash Functions:

### 2.1.1. Hash Functions GUI

Sau khi code xong phần CLI, em xuất file sang dll để có thể build được giao diện. Sử dụng Visual Studio 2022 để build giao diện Winform. Dưới đây là một số ảnh minh hoạ:

A screenshot of a computer

Description automatically generated

Ảnh 2: Input được nhập từ File

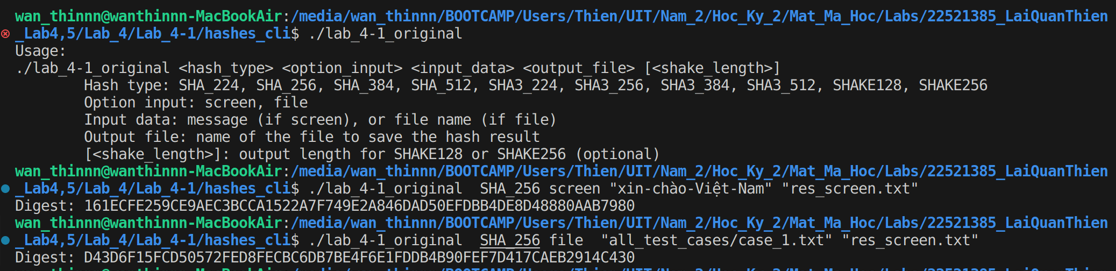
A screenshot of a computer

Description automatically generated

Ảnh 3: Input được nhập từ Bàn phím

### 2.1.2. Hash Functions CLI

- Dưới đây là hình ảnh minh hoạ cách sử dụng của các hàm Băm (SHA2, SHA3, SHAKE128, SHAKE256)



Ảnh 4: Minh hoạ cách sử dụng Hash Function CLI App

- Dưới đây là hình ảnh minh hoạ phần code chạy 1000 lần của các hàm Băm (SHA2, SHA3, SHAKE128, SHAKE256). Thời gian thực nghiệm được đo là tổng của bao gồm toàn bộ quá trình load file, check format, thực hiện mã hoá/giải mã và lưu/xuất file:

A screen shot of a computer code

Description automatically generated

Ảnh 5: Minh hoạ phần code chạy 1000 lần các hàm băm

##### 2.1.2.1 Bảng thống kê số liệu thực nghiệm

Dưới đây là bảng thống kê chi tiết thời gian trung bình thời gian băm của từng thuật toán (đơn vị thời gian: ***ms***):



Ảnh 6: Bảng thời gian trung bình 1000 lần Băm trên Windows 11



Ảnh 7: Bảng thời gian trung bình 1000 lần Băm trên Ubuntu 22.04

##### 2.1.2.2 Biểu đồ cột so sánh các thuật toán băm

Dựa theo số liệu từ các bảng ở trên, vẽ biểu đồ để dễ dàng so sánh một cách trực quan:

Ảnh 8: Biểu đồ thời gian trung bình 1000 lần Băm trên Windows 11

Ảnh 9: Biểu đồ thời gian trung bình 1000 lần Băm trên Ubuntu 22.04

##### 2.1.2.3 Biểu đồ đường so sánh thời gian thực thi trên hai hệ điều hành

Vì các test case với file nhỏ không phản ánh sự khác biệt quá rõ ràng (File\_1, File\_2, File\_3). Vì vậy để trực quan hơn về việc so sánh thời gian thực thi giữa 2 hệ điều hành, em xin so sánh chi tiết thời gian băm của File\_4 (165 MB) và File\_5 (1 GB). Dưới đây là biểu đồ so sánh thời gian băm 2 file đó trên 2 hệ điều hành:

Ảnh 10: Biểu đồ so sánh thời gian trung bình băm 1000 lần file 165 MB trên 2 hệ điều hành

Ảnh 11: Biểu đồ so sánh thời gian trung bình băm 1000 lần file 1 GB trên 2 hệ điều hành

##### 2.1.2.4. Phân tích và so sánh

- Trên cả 2 hệ điều hành, ta có thể thấy với kích thước file đầu vào càng lớn, thời gian chạy càng tăng. Khi so sánh thời gian chạy giữa các thuật toán với nhau trên cùng một độ lớnhững của file đầu vào, ta thấy với file nhỏ thì chênh lệch thời gian không đáng kể, tuy nhiên khi file càng lớn, sự khác biệt thời gian cũng lớn dần.

- Các thuật toán Hash càng phức tạp thì sẽ cho thời gian chạy càng lâu, tuy nhiên đánh đổi lại đó là độ bảo mật được tăng lên đáng kể. Do đó khi áp dụng thực tế, người dùng có thể linh động sử dụng các thuật toán khác nhau dựa vào nhu cầu.

- Nếu xét thời gian chạy của 2 hệ điều hành trên cùng 1 thuật toán với và cùng 1 kích thước file đầu vào, ta thấy rằng đa phần Ubuntu đều nhanh hơn Windows. Độ chênh lệch thời gian từ 5-10% tuỳ thuộc vào các điều kiện test khác nhau.

## 2.2. PKI and digital certificate

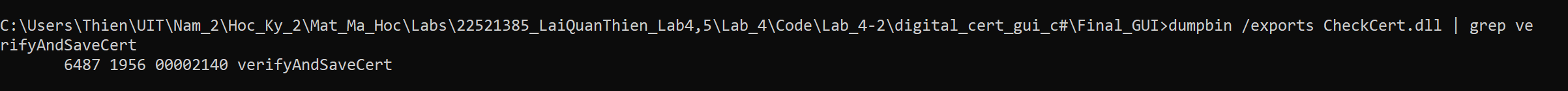
Tiến hành truy cập **apple.com**, **facebook.com** và **uit.edu.vn** để xuất những chứng chỉ cần thiết cho bài lab.

A screenshot of a computer

Description automatically generated

Ảnh 12: Ảnh minh hoạ cho việc xuất các chứng chỉ trên apple.com

### 2.2.1. PKI and digital certificate GUI



Ảnh 13: Check tên hàm dùng để xuất dll

A screenshot of a computer

Description automatically generated

Ảnh 14: Kết quả trả về nếu Certificate Valid

A screenshot of a computer

Description automatically generated

Ảnh 15: Kết quả trả về nếu Certificate Invalid

### 2.2.2. PKI and digital certificate CLI

A screenshot of a computer

Description automatically generated

Ảnh 16: Demo CLI Application

A black screen with many small colored text

Description automatically generated with medium confidence

Ảnh 17: Trường hợp Invalid

## 2.3. Collision and length extension attacks on Hash functions

### 2.3.1. Two collision messages have the same prefix string

A screenshot of a computer

Description automatically generated

Ảnh 18: Chuỗi Prefix

A screenshot of a computer program

Description automatically generated

Ảnh 19: Kết quả chạy ./md5\_fastcoll trên Ubuntu 22.04

A black screen with many squares

Description automatically generated

Ảnh 20: Check mã Hexa của 2 file

A screenshot of a computer

Description automatically generated

Ảnh 21: So sánh sự khác nhau của 2 output

- Sau khi clone HashClash về, em tiến hành Build theo hướng dẫn. Sau khi build xong, truy cập vào thư mục ../bin của hashclash để copy file thực thi md5\_fastcoll vào thư mục đang làm việc.

- Tiến hành tạo 1 file prefix.txt có chuỗi là “22521385-LaiQuanThien12345678” và chạy lệnh “./md5\_fastcoll -p prefix.txt -o output\_1.bin output\_2.bin”.

- Sau khi thực thi thành công lệnh trên, chương trình sẽ tạo ra 2 file output\_1.bin và output\_2.bin có chung prefix và có mã băm md5 giống nhau, nhưng file khác nhau hoàn toàn (ảnh 20 so sánh sự khác nhau giữa 2 file)

### 2.3.2. Two different C++ programs but have the same MD5

A black screen with white text

Description automatically generated

Ảnh 22: File\_1.cpp

A black screen with white text

Description automatically generated

Ảnh 23: File\_2.cpp

A computer screen shot of a program

Description automatically generated

Ảnh 24: Kết quả chạy

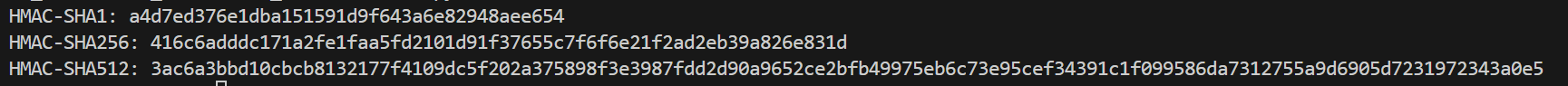
## 2.4. Length extension attacks on MAC  in form: H(k||m), k is secret key

### 2.4.1. Show length extension attacks on MAC using SHA1, SHA256, SHA512 using hashpump tool

A screen shot of a computer program

Description automatically generated

Ảnh 25: Chương trình Python để tạo ra các giá trị HMAC (SHA1, SHA256, SHA512)



Ảnh 26: Kết quả băm trả về dựa trên Key và Message

A screenshot of a computer program

Description automatically generated

Ảnh 27: Kết quả tấn công trả về lần lượt (SHA1, SHA256, SHA512) với padding là “con-meo-de-thuong-qua”

### 2.4.2. Coding self programs that can attacks on MAC using SHA256 (for bonus 5/100 points)

A screen shot of a computer program

Description automatically generated

Ảnh 28: Sử dụng thư viện CryptoPP để viết chương trình tấn công MAC SHA256

A screenshot of a computer

Description automatically generated

Ảnh 29: Kết quả trả về khi chạy HashPump và chương trình tự code có predicted sig giống nhau