**BÁO CÁO THỰC HÀNH LAB5-6**

**MẬT MÃ HỌC**

***Họ và tên: Nguyễn Văn Thọ***

***MSSV: 20521974***

***ANTN2020***

**Hardware:**

|  |  |
| --- | --- |
| **CPU** | **Intel core i5-8300H @ 2.3Ghz** |
| **RAM** | **16Gb** |

**LAB05 - HASH FUNCTION**

**5.1 Hash Functions**

**Ảnh có chứa bàn

Mô tả được tạo tự động**

**Ảnh có chứa bàn

Mô tả được tạo tự động**

* 1. **PKI and digital certificate**
* **Load file**

**Ảnh có chứa văn bản

Mô tả được tạo tự động**

* **Get Certificate Signature Algorithm**

**Ảnh có chứa văn bản

Mô tả được tạo tự động**

**\* Running**

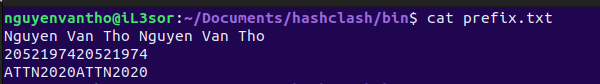
**Ảnh có chứa văn bản

Mô tả được tạo tự động**

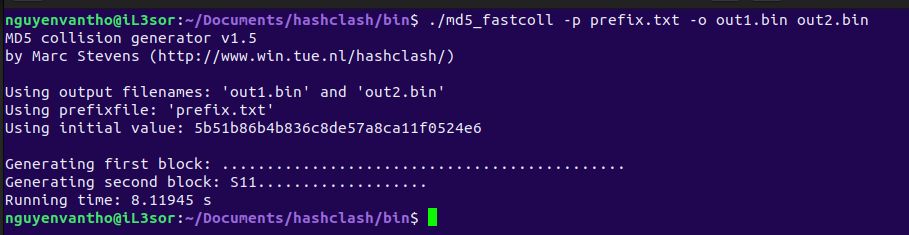
**LAB06 – MD5 COLLISION**

**6.1 - Two collision messages have the same prefix string**

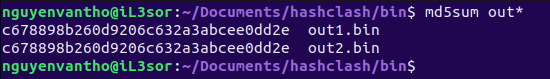
1. Tạo file text kích thước 1 block (64 bytes)



1. Tìm 2 file có md5 collision với dựa trên prefix trên

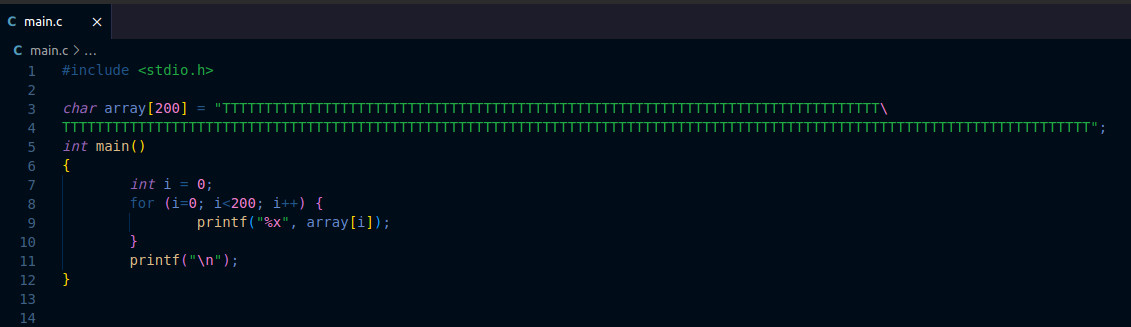


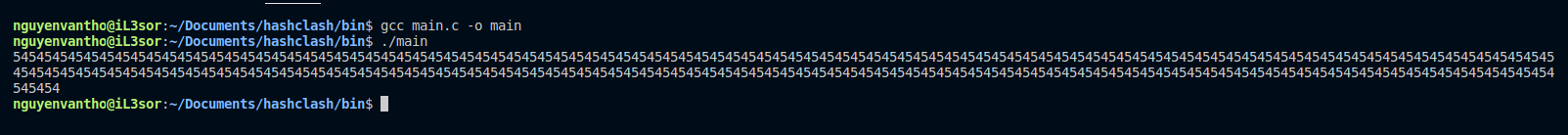
1. Thu được 2 file khác nhau, có cùng md5 hash digest



**6.1 - Two different C++ programs but have the same MD5.**

1. Tạo chương trình C như sau: (tên chương trình là main.c)





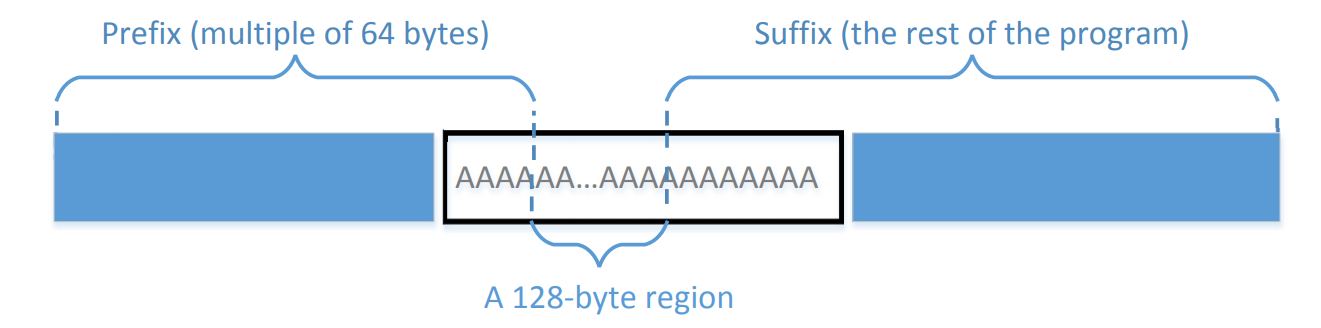
*Việc tạo mảng với 200 ký tự giống nhau giúp dễ tìm kiếm vị trí của chương trình trong đoạn mã nhị phân của nó hơn.*

Biên dịch chương trình xong, ta thấy được vị trí của mảng **array**.

Ảnh có chứa văn bản, thiết bị điện tử, máy tính

Mô tả được tạo tự động

**Ý tưởng thực hiện yêu cầu của đề như sau:**



1. Tách chương trình **main** làm 3 phần:
   * Prefix: từ đầu đến vị trí của mảng **array** (0x3020)
   * Phần sinh ra thêm sau khi tìm md5\_collision, nó sẽ nằm ở vị trí bên trong mảng **array**
   * Suffix: từ vị trí tiếp tục của phần bên trên đến hết chương trình
2. Tìm 2 file binary có md5 collision, sử dụng prefix (theo như hướng dẫn trên kia)
3. Kết hợp phần suffix vào 2 file binary trên (có cùng md5\_hash\_digest) tìm được = > tìm được 2 file có md5 collision.

Giải thích: Sở dĩ cách này không khiến 2 chương trình tạo ra không bị hỏng là do phần dôi ra ở bước b) (khi tính toán md5 collision) nằm trọn trong mảng **array**. Nên không khiến chương trình hỏng.

1. Tách prefix bằng lệnh: **header -c 12320 main > prefix**

(0x3020 tương đương với 12320 bytes)

Ảnh có chứa văn bản

Mô tả được tạo tự động

1. Dùng hashclash tìm 2 file binary có md5 collision cho phần prefix của chương trình **main**

Ảnh có chứa văn bản

Mô tả được tạo tự động

*Tìm được 2 file binary khác nhau có cùng md5 hash digest*

1. Chương trình ban đầu có 16240 bytes, trong khi 2 file vừa tìm được ở trên chỉ có 12480 bytes, vậy là cần lấy ra 3760 bytes làm suffix

Ảnh có chứa văn bản

Mô tả được tạo tự động

1. Tách 3760 bytes cuối của chương trình main

Ảnh có chứa văn bản

Mô tả được tạo tự động

1. Ghép prefix vs suffix lại để hoàn thành chương trình

Ảnh có chứa văn bản

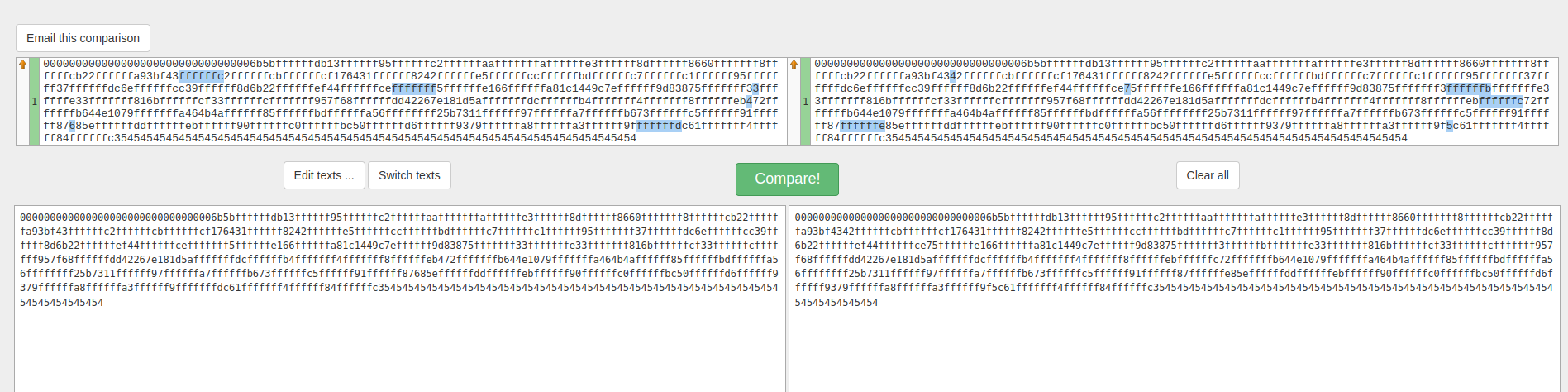
Mô tả được tạo tự động

1. Kiểm tra hoạt động của 2 chương trình

Ảnh có chứa văn bản, ảnh chụp màn hình

Mô tả được tạo tự động

*2 kết quả in ra khác nhau? -> kiểm chứng:*



1. Kiểm tra md5 hash digest của 2 chương trình

Ảnh có chứa văn bản

Mô tả được tạo tự động

* Tìm được 2 chương trình C khác nhau nhưng có cùng md5\_hash\_digest.

**6.2 – Length extension attacks on MAC in form: SHA256(k||m), k is secret key**

Tạo các dữ liệu cần thiết để chạy tool **hashpump**

Ảnh có chứa văn bản

Mô tả được tạo tự động

* Length extension attack with SHA1

Ảnh có chứa văn bản

Mô tả được tạo tự động

* Length extension attack with SHA256

Ảnh có chứa văn bản

Mô tả được tạo tự động

* Length extension attack with SHA512

Ảnh có chứa văn bản

Mô tả được tạo tự động