BÁO CÁO TASK 4

**Môn học: Mật mã học**

*GVHD: Nguyễn Ngọc Tự*

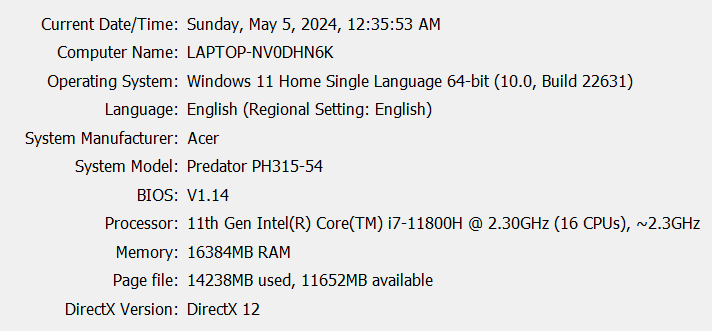
**THÔNG TIN CHUNG:**

Lớp: NT219.O21.ANTT

|  |  |  |
| --- | --- | --- |
| **Họ và tên** | **MSSV** | **Email** |
| Phan Quốc Đạt | 22520233 | 22520233@gm.uit.edu.vn |

BÁO CÁO CHI TIẾT

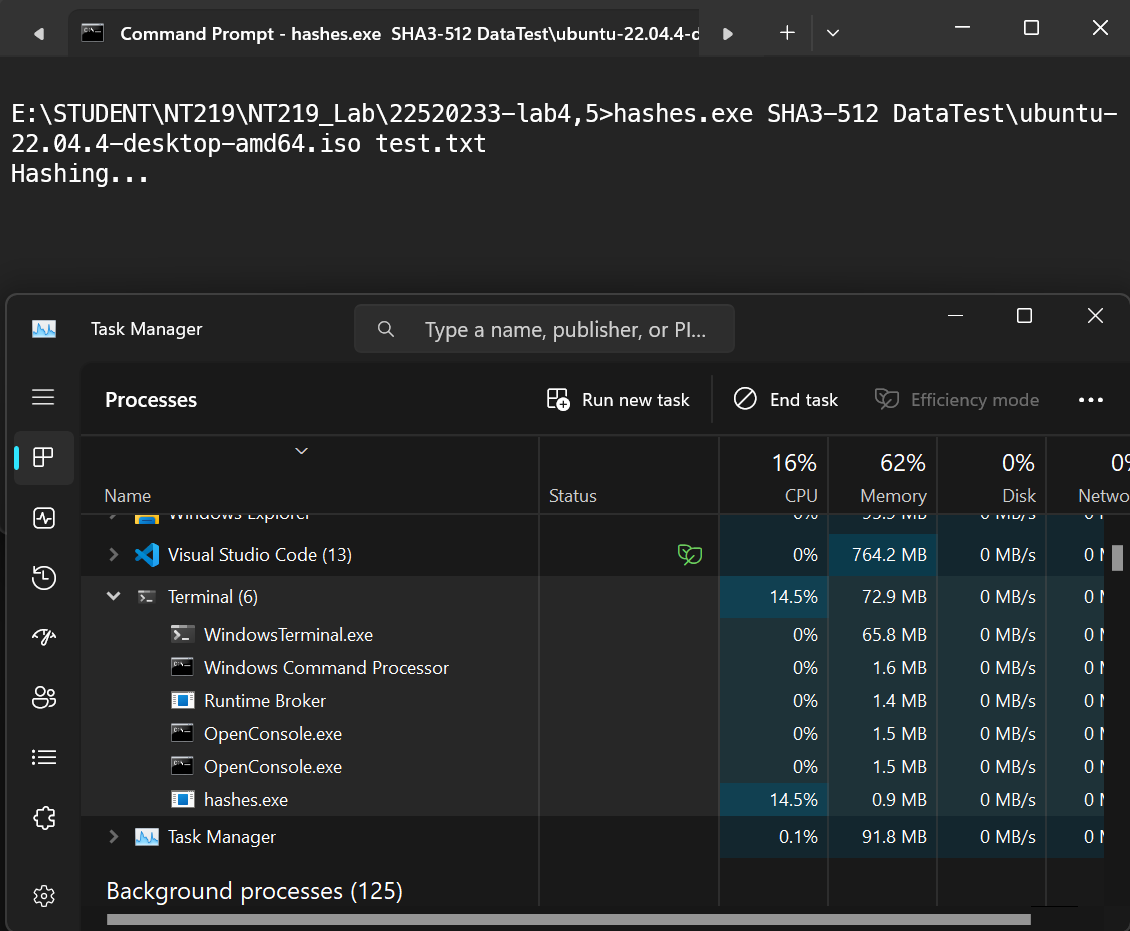
1. **Tài nguyên phần cứng**



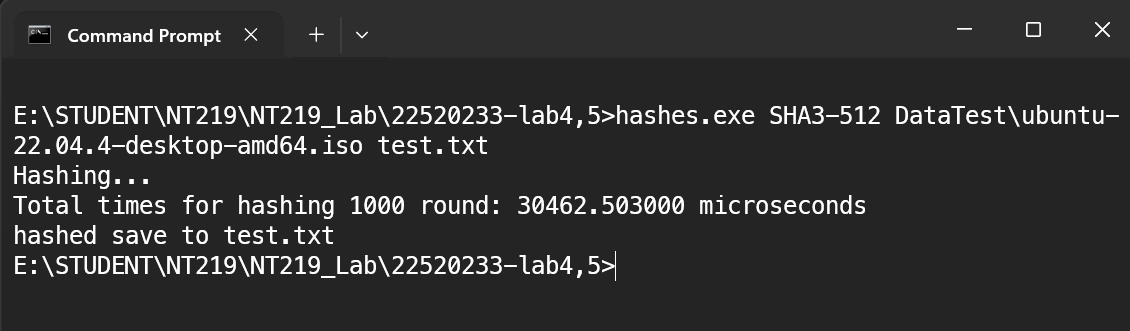


1. **Task 4.1: Hash Functions**

**Demo hashing**



Khi hashing chiếm khoảng 15% CPU



Kết quả được lưu vào file

**Thống kê kết quả**

Dưới đây là kết quả thống kế được từ Windows 11 và Ubuntu, kết quả thu lại được từ kích thước dữ liệu khác nhau. Số liệu ở dưới là thời gian được tính theo ms (millisecond). Kết quả ở đây được tính toán từ thời gian bắt đầu hash 1000 lần cho đến khi kết thúc (tính cả thời gian ghi vào file).

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Windows 11** | | | | | |
| **Hash** | **2mb** | **10mb** | **50mb** | **100mb** | **5gb** |
| **SHA224** | **0.013** | **0.027** | **0.079** | **0.147** | **6.693** |
| **SHA256** | **0.007** | **0.018** | **0.076** | **0.144** | **7.068** |
| **SHA384** | **0.010** | **0.029** | **0.133** | **0.242** | **11.824** |
| **SHA512** | **0.010** | **0.029** | **0.127** | **0.242** | **12.034** |
| **SHA3-224** | **0.011** | **0.037** | **0.163** | **0.324** | **15.575** |
| **SHA3-256** | **0.011** | **0.039** | **0.178** | **0.354** | **16.979** |
| **SHA3-384** | **0.0130** | **0.047** | **0.213** | **0.419** | **20.356** |
| **SHA3-512** | **0.016** | **0.063** | **0.290** | **0.574** | **28.234** |
| **SHAKE128** | **0.011** | **0.033** | **0.147** | **0.288** | **13.943** |
| **SHAKE256** | **0.011** | **0.039** | **0.174** | **0.345** | **16.436** |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Ubuntu** | | | | | |
| **Hash** | **2mb** | **10mb** | **50mb** | **100mb** | **5gb** |
| **SHA224** | **0.002** | **0.008** | **0.039** | **0.073** | **3.450** |
| **SHA256** | **0.003** | **0.010** | **0.038** | **0.076** | **3.437** |
| **SHA384** | **0.004** | **0.017** | **0.073** | **0.143** | **6.837** |
| **SHA512** | **0.005** | **0.019** | **0.072** | **0.143** | **6.828** |
| **SHA3-224** | **0.005** | **0.024** | **0.103** | **0.203** | **9.769** |
| **SHA3-256** | **0.005** | **0.024** | **0.108** | **0.214** | **10.493** |
| **SHA3-384** | **0.007** | **0.030** | **0.138** | **0.273** | **13.221** |
| **SHA3-512** | **0.011** | **0.041** | **0.192** | **0.383** | **18.849** |
| **SHAKE128** | **0.005** | **0.019** | **0.092** | **0.178** | **8.780** |
| **SHAKE256** | **0.005** | **0.028** | **0.111** | **0.214** | **10.905** |

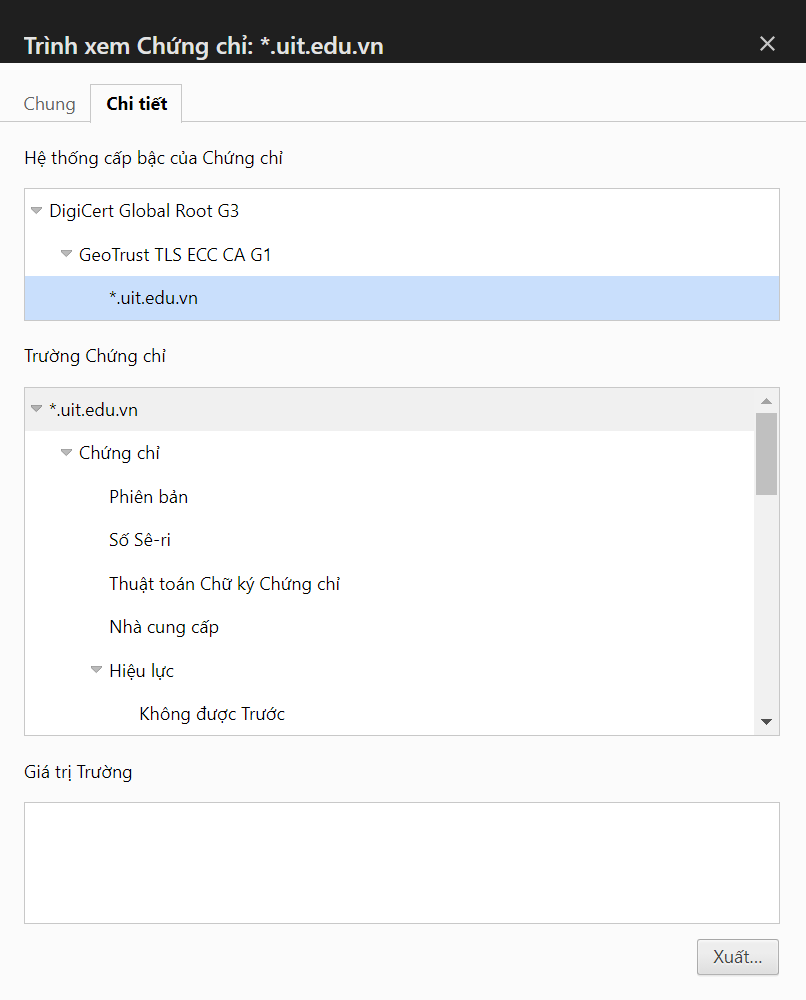
**Nhận xét**

- Trên cả 2 hệ điều hành, ta có thể thấy với input càng lớn, thời gian chạy càng tăng. Khi so sánh thời gian chạy giữa các thuật toán với nhau trên cùng một độ lớn input, ta thấy với input nhỏ thì chênh lệch thời gian không đáng kể, tuy nhiên khi input 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 với và cùng 1 độ dài input, ta thấy rằng đa phần Ubuntu nhanh hơn Windows, có sự chênh lệch thời gian không quá lớn.

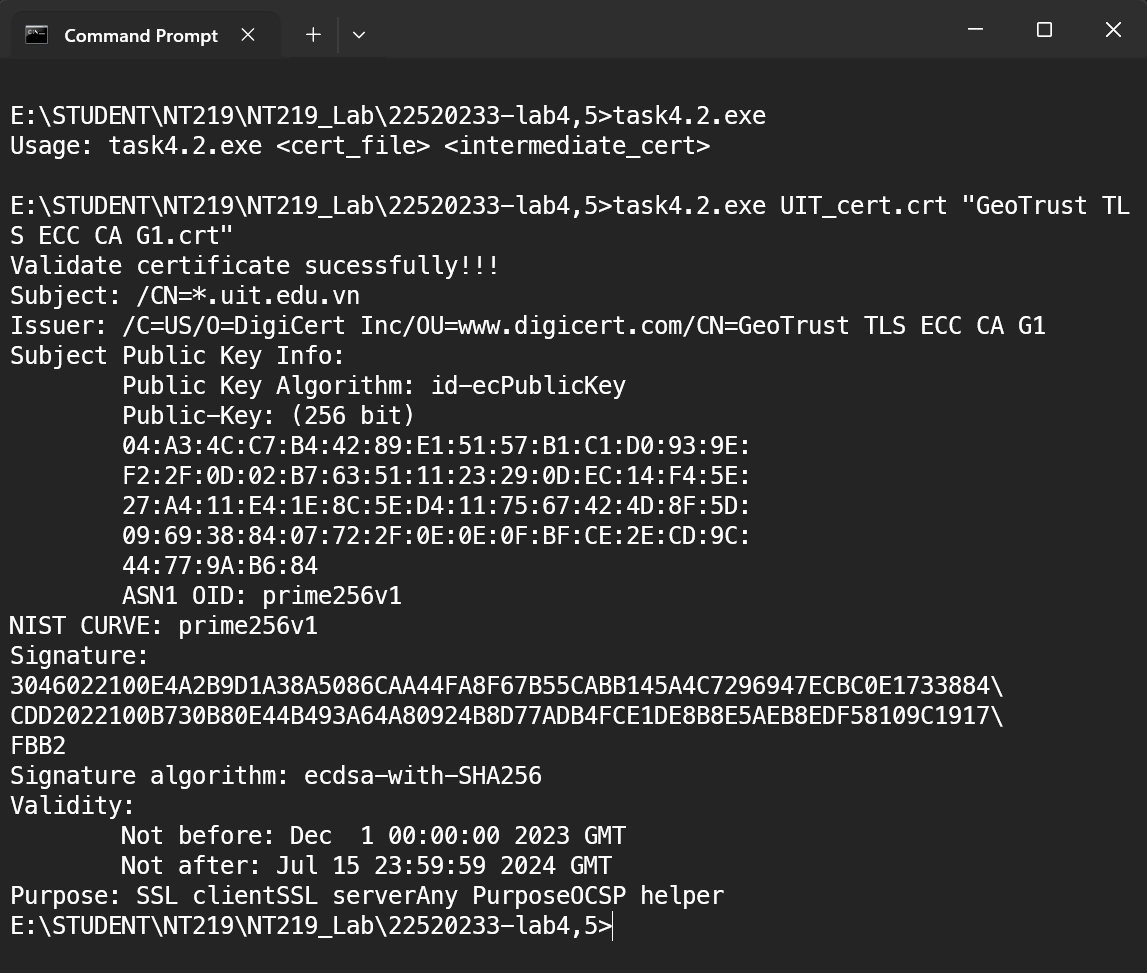
1. **Task 4.2: PKI and digital certificate**

- Em lấy chứng chỉ của trang uit.edu.vn để test tool



Lấy chứng chỉ của trang web uit.edu.vn và chứng chỉ CA

- Dưới đây là kết quả của tool em tạo để xác thực và phân tích tất cả các trường của chứng chỉ X.509

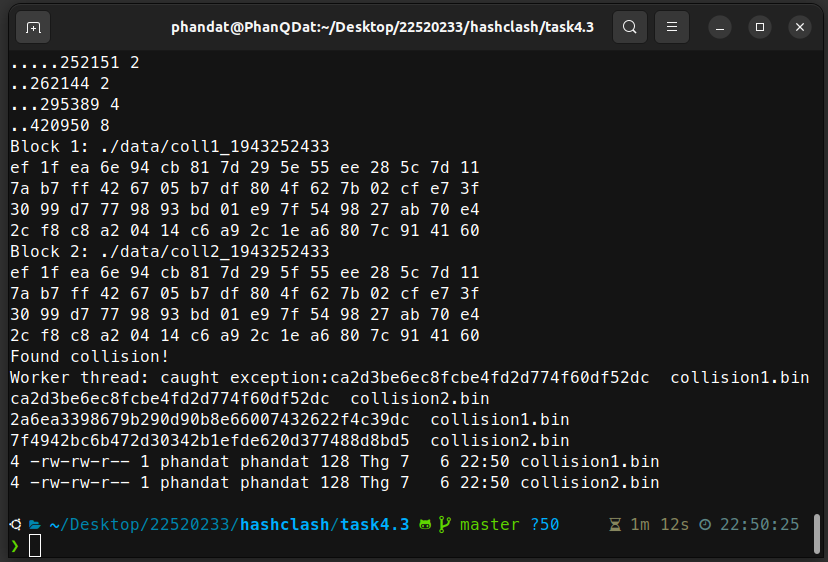


Đây là kết quả của tool đã phân tích các trường thông tin của chứng chỉ

1. **Task 4.3: Collision and length extension attacks on Hash functions**

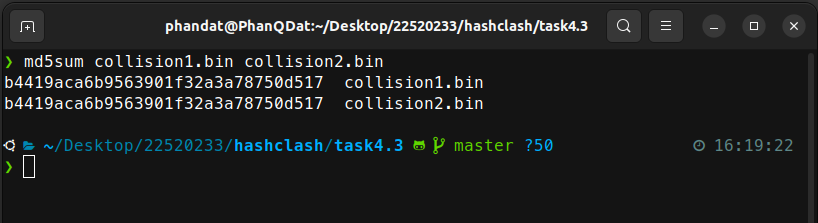
**Two collision messages have the same prefix string**

Chạy tool để tạo 2 file khác nhau có cùng tiền tố và cùng tiền tố băm MD5



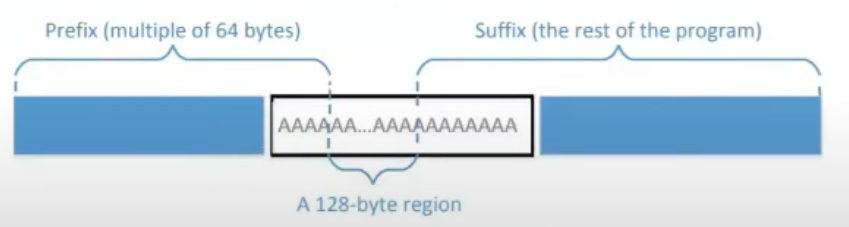
So sánh 2 output file => có thể thấy có 1 số điểm khác biệt

2 file có cùng tiền tố nhưng khác hậu tố. Giá trị của nó được băng ra vẫn bằng nhau



**Two different C++ programs but have the same MD5**

Ý tưởng:



- Đầu tiên, ta tách file thực thi làm 3 phần

+ Phần 1: tiền tố, bắt đầu từ vị trí đầu tệp đến vị trí nào đó ở giữa mảng, trong trường hợp trên, em cắt bớt byte đầu tiên của tệp thực thi, đồng nghĩa cắt bớt ký tự A trong mảng

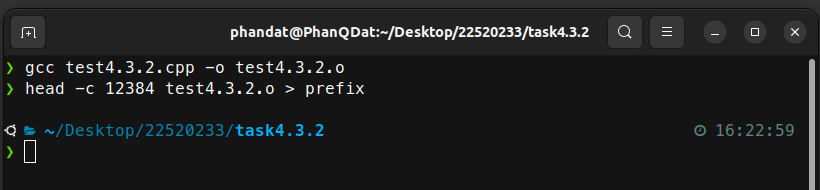
+ Phần 2: tìm thấy xung đột băm, là một phần của mảng

+ Phần 3: hậu tố, tiếp tục ở phần còn lại đến cuối tệp

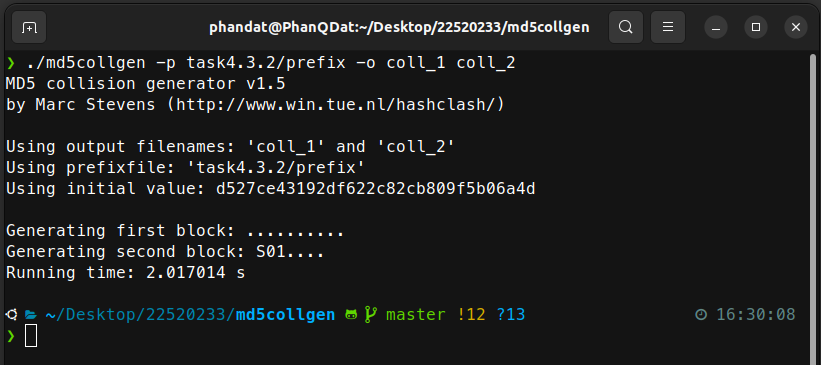
- Thứ 2: ta thấy có sự xung đột với tiền tố

- Thứ 3: nối phần hậu tố còn lại của file gốc với 2 file mới bị xung đột

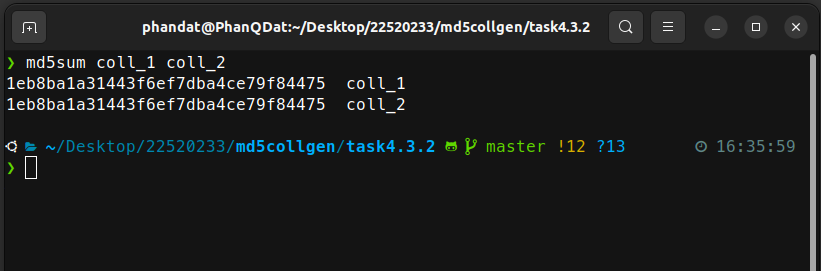
Ta sẽ thực hiện cắt bớt tiền tố chứa ký tự A dữ liệu



Với tiền tố, tôi có thể tìm thấy xung đột với lệnh sau:

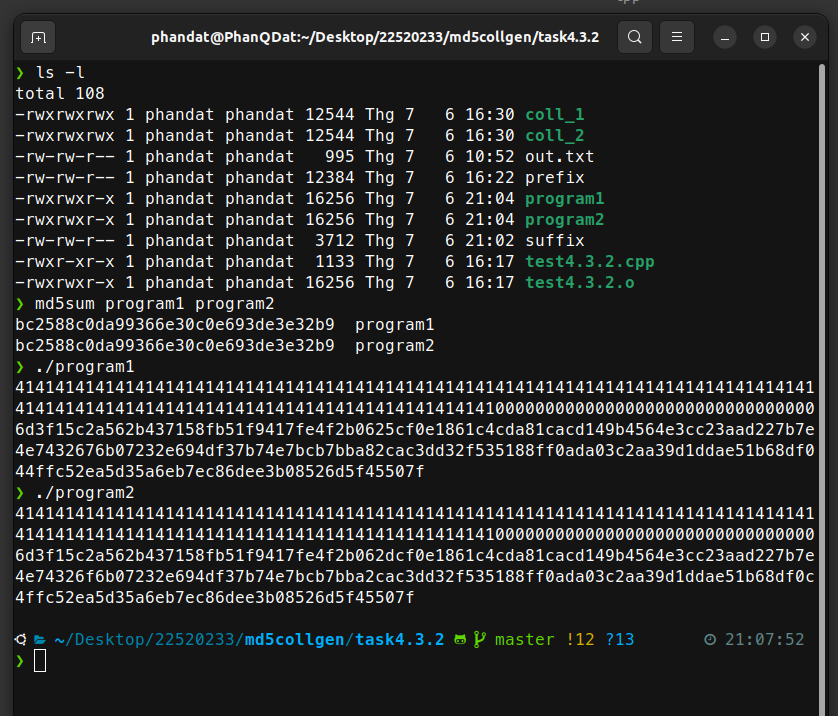
****

2 file có giá trị hàm băm bằng nhau mặc dù chúng khác nhau



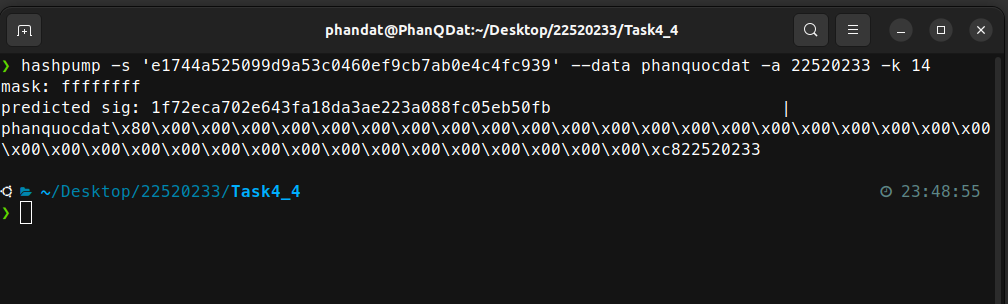
Tiếp theo, nối hậu tố vào vụ va chạm

Cho đến nay, tôi đã có 2 tệp thực thi giống hệt md5hash có kích thước là 16256 bytes

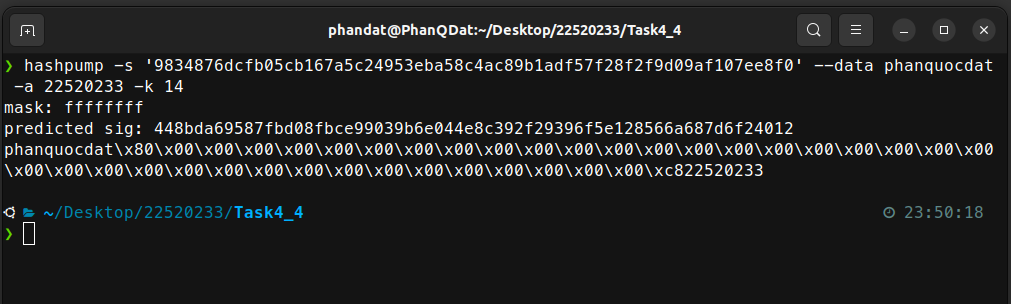


1. **Task 4.4 Length extension attacks on MAC in form: H(k||m), k is secret key**

Attack SHA1



Attack SHA256



Attack SHA512

