

**HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
KHOA AN TOÀN THÔNG TIN**



**BÁO CÁO BÀI THỰC HÀNH
HỌC PHẦN: MẬT MÃ HỌC CƠ SỞ
MÃ HỌC PHẦN: INT1344**

**XÂY DỰNG BÀI THỰC HÀNH TRÊN NỀN TẢNG LABTAINER
TÌM HIỂU COLLISION HASH TRÊN MD5**

Sinh viên thực hiện:

B22DCAT076 – Nguyễn Hữu Đạt

B22DCAT132 – Phí Công Huân

Giảng viên hướng dẫn: PGS.TS Đỗ Xuân Chợt

HỌC KỲ 2 NĂM HỌC 2024-2025

MỤC LỤC

MỤC LỤC	2
CHƯƠNG 1. GIỚI THIỆU CHUNG VỀ BÀI THỰC HÀNH	3
1.1 Mục đích.....	3
1.2 Tìm hiểu lý thuyết	3
1.2.1 Hàm băm mật mã là gì	3
1.2.2 Collision là gì ?	3
1.2.3 MD5 và lỗi hỏng collision.....	3
CHƯƠNG 2. NỘI DUNG THỰC HÀNH	5
2.1 Các bước thực hiện.....	5
2.1.1 TASK 1	5
2.1.2 TASK 2	6
2.1.3 TASK 3	7
2.1.4 Kết thúc lab:	8

CHƯƠNG 1. GIỚI THIỆU CHUNG VỀ BÀI THỰC HÀNH

1.1 Mục đích

Bài lab này nhằm giúp sinh viên hiểu và thực hành tấn công va chạm trên hàm băm MD5 – một trong những thuật toán băm phổ biến nhưng đã bị phá vỡ. Cụ thể, sinh viên sẽ:

- Tìm hiểu cơ chế hoạt động của hàm băm và khái niệm collision (xung đột giá trị băm).
- Quan sát và kiểm chứng trực tiếp hai tệp khác nhau nhưng có cùng mã băm MD5.
- Sử dụng công cụ thực tế như fastcoll để tạo ra collision.

1.2 Tìm hiểu lý thuyết

1.2.1 Hàm băm mật mã là gì

Hàm băm mật mã (Cryptographic Hash Function) là một hàm nhận đầu vào là chuỗi dữ liệu bất kỳ và trả về một chuỗi có độ dài cố định, gọi là **giá trị băm (hash)**. Ví dụ:

$$MD5("hello") = 5d41402abc4b2a76b9719d911017c592$$

Các tính chất quan trọng của hàm băm mật mã:

- Xác định: Một đầu vào cụ thể luôn tạo ra cùng một giá trị băm. Điều này có nghĩa là cùng một dữ liệu đầu vào sẽ luôn dẫn đến cùng một mã băm.
- Đơn hướng: Rất khó để tính ngược lại đầu vào từ giá trị băm, đảm bảo tính bảo mật.
- Tính toàn vẹn: Một thay đổi nhỏ trong dữ liệu đầu vào sẽ tạo ra một giá trị băm hoàn toàn khác, giúp phát hiện bất kỳ thay đổi hoặc sửa đổi nào trong dữ liệu.
- Hiệu quả: Hàm băm phải có khả năng xử lý và tạo giá trị băm một cách nhanh chóng ngay cả với lượng dữ liệu lớn.
- Không trùng lặp: Xác suất để hai dữ liệu khác nhau có cùng giá trị băm (còn gọi là "collision") phải cực kỳ thấp, đảm bảo rằng mỗi giá trị băm là duy nhất cho một tập dữ liệu cụ thể.

1.2.2 Collision là gì ?

Collision Attack là một trong những loại tấn công nguy hiểm nhất, khai thác khả năng tìm hai đầu vào khác nhau cho cùng giá trị băm.

Collision Attack làm suy yếu các hệ thống bảo mật dựa vào hàm băm (ví dụ: chứng chỉ số, xác thực tệp).

Trên lý thuyết chuồng chim bồ câu thì xung đột băm luôn tồn tại với mỗi thuật toán hash, các hàm băm hiện tại đều đang đảm bảo xác suất xảy ra xung đột ở mức thấp nhất, chứ không thể giảm xuống 0. Thật vậy, giả sử hàm hash H có đầu ra là một chuỗi nhị phân 256 bit, thì không gian giá trị có độ lớn 2^{256} khi đưa $2^{256} + 1$ thông điệp đầu vào khác nhau thực hiện băm, thì chắc chắn tồn tại hai thông điệp trả về cùng một kết quả.

1.2.3 MD5 và lỗ hổng collision

MD5 là một thuật toán băm 128-bit được phát triển từ năm 1991. Tuy nhiên, từ năm 2004, các nhà nghiên cứu (Wang et al.) đã chứng minh có thể tìm được collision trong vài phút, và đến nay có nhiều công cụ công khai như: fastcoll, unicoll, hashclash

Kết quả là: MD5 không còn an toàn cho mục đích xác thực hay chứng thực.

Nếu một hệ thống chỉ dựa vào giá trị hash để xác minh file hoặc chữ ký số, attacker có thể:

- Tạo 2 file có cùng hash, gửi bản "vô hại" để được ký/xác thực, rồi thay thế bằng bản độc hại có cùng hash.
- Gây tổn hại nghiêm trọng đến các hệ thống kiểm tra integrity, chứng cứ pháp lý, hợp đồng điện tử, v.v

CHƯƠNG 2. NỘI DUNG THỰC HÀNH

2.1 Các bước thực hiện

- Khởi động bài lab

imodule <https://github.com/chupinana04/Labtainer/raw/refs/heads/main/imodule.tar>

- Sinh viên khởi động bài lab

labtainer -r hash-collision

- (Chú ý: sinh viên sử dụng MÃ SINH VIÊN của mình để nhập thông tin người thực hiện bài lab khi có yêu cầu, để sử dụng khi chấm điểm.)
- Sau khi khởi động bài lab, một container hiện lên sinh viên thực hiện làm theo yêu cầu:

2.1.1 TASK 1

- Sinh viên tạo file1.txt chứa nội dung là mã sinh viên:

echo "{mã sinh viên}" > file1.txt

- Sinh viên sử dụng lệnh md5sum để băm file1.txt mình vừa tạo:

md5sum file1.txt

- Màn hình xuất hiện giá trị băm của file1.txt là một giá trị toán tắt thông điệp 128 bit. Việc tính toán giá trị tóm tắt MD5 được thực hiện trong các giai đoạn riêng biệt, xử lý từng khối dữ liệu 512 bit.
- Sinh viên tạo file2.txt có nội dung giống hệt file1.txt nhưng có thêm một kí tự khác ví dụ dấu chấm, dấu hỏi,..

echo "{mã sinh viên}?" > file2.txt

- Sinh viên băm file2.txt vừa tạo:

md5sum file2.txt

- Ta thấy rằng chỉ cần thay đổi 1 kí tự trong file cũng sẽ làm hàm băm thay đổi hoàn toàn. Thể hiện tính chất của hàm băm.

2.1.2 TASK 2

- Năm 2005, Wang Xiaoyun và nhóm nghiên cứu Trung Quốc đã công bố một cuộc tấn công hiệu quả vào hàm băm MD5, cho phép tạo hai thông điệp khác nhau nhưng có cùng giá trị băm trong thời gian ngắn bằng kỹ thuật phân tích vi sai. Đây là lần đầu tiên một collision thực tế được tạo ra nhanh chóng trên MD5.
- Năm 2009, hai nhà nghiên cứu Marc Stevens và Dan Shumow đã tạo ra thành công hai hình ảnh khác nhau nhưng có cùng giá trị băm MD5. Hai nhà nghiên cứu cũng nghiên cứu và phát triển các công cụ tạo collision như fastcoll, unicoll, hashclash,..
- Sau đây chúng ta có hai hình ảnh có cùng giá trị băm được tạo bởi công cụ HashClash của Marc Stevent chạy trên AWS. Để tạo được các collision block cho hai ảnh mất tổng cộng 16 tiếng chạy.
- Sử dụng lệnh `ls -l` thấy hai ảnh `ship.jpg` và `plane.jpg` có cùng kích thước:

ls -l

- Cài đặt radare2 để so sánh hai file và hiển thị sự khác biệt ở dạng hexdump:

sudo apt install radare2

- Xem hai ảnh bằng trình xem ảnh eog:

eog ship.jpg

eog plane.jpg

- So sánh 2 file dưới dạng hexdup:

radiff2 -x ship.jpg plane.jpg

- Ta thấy 2 ảnh hoàn toàn khác nhau, với lệnh `radiff2` sẽ so sánh 2 file, màu đỏ ứng với các vị trí khác nhau.
- Băm hai ảnh bằng md5:

md5sum ship.jpg plane.jpg

- Mặc dù 2 ảnh hoàn toàn khác nhau, kể ra ở dưới dạng hex, nhưng lại cùng giá trị băm. Điều này chứng tỏ tính chất băm của MD5 đã bị phá vỡ hoàn toàn.

2.1.3 TASK 3

- Ở task này, chúng ta sẽ tìm hiểu về Identical-Prefix Collision: 2 file có phần tiền tố giống nhau nhưng khác nhau ở phần dữ liệu tiếp theo, tạo ra cùng một giá trị băm: $H(P|S1) = H(P|S2)$. Và chúng ta sẽ tạo 2 file đơn giản khác nhau nhưng có cùng giá trị băm MD5 bằng công cụ fastcoll.

- Dùng lệnh ls thấy file zip hashclash-static-release-v1.2b.tar.gz

ls

- Giải nén file bằng lệnh:

tar -xvf hashclash-static-release-v1.2b.tar.gz

- Mở file vừa giải nén, trong thư mục bin có chứa công cụ md5_fastcoll
- Di chuyển file thực thi md5_fastcoll về thư mục home/ubuntu để làm việc:\

mv md5_fastcoll /home/ubuntu

- Quay về /home/ubuntu, tạo file tiền tố prefix.txt:

echo "PTIT" > prefix.txt

- Sử dụng công cụ fastcoll cho file prefix.txt:

./md5_fastcoll prefix.txt

- Đợi công cụ chạy xong, sử dụng lệnh ls, thấy xuất hiện 2 file mới prefix_msg1.txt và prefix_msg2.txt
- Sử dụng lệnh radiff2 lên 2 file này:

radiff2 -x prefix_msg1.txt prefix_msg2.txt

- Nhìn vào hình ta thấy, phần đầu của hai file là giống nhau (file prefix.txt), phần sau là các khối collision do fastcoll tính toán ra, các khối này nhìn rất giống dữ liệu ngẫu nhiên, nhưng thực tế chỉ khác nhau ở một vài bit có chủ đích để hai file cuối cùng có cùng giá trị băm.

- Sử dụng lệnh băm:

md5sum prefix_msg1.txt prefix_msg2.txt

- Ta thấy hai file khác nhau nhưng lại có cùng giá trị băm
- Thử lại với các hàm băm mạnh hơn, vd SHA256
-

sha256sum prefix_msg1.txt prefix_msg2.txt

- Giá trị băm của hai file khác nhau, và chậm chỉ tạo ra cho md5

2.1.4 Kết thúc lab:

- Trên terminal khởi động lab, sinh viên sử dụng lệnh:

stoplab

- Khi bài lab kết thúc, một tệp lưu kết quả được tạo và lưu vào một vị trí được hiển thị bên dưới stoplab. Sinh viên cần nộp file .lab để chấm điểm.
- Để kiểm tra kết quả khi trong khi làm bài thực hành sử dụng lệnh:

checkwork hash-collision

- Khởi động lại bài lab: Trong quá trình làm bài sinh viên cần thực hiện lại bài lab, dùng câu lệnh:

labtainer -r hash-collision

