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

# HỌC PHẦN: CÁC KỸ THUẬT GIẤU TIN
# MÃ HỌC PHẦN: INT14102

## Chủ đề: Giấu tin trong âm thanh

## Lab: lsb_ctf1

| | |
|---|---|
| Sinh viên thực hiện: | Nguyễn Trường Sơn |
| Mã sinh viên: | B21DCAT164 |
| Nhóm: | 04 |
| Giảng viên hướng dẫn: | Đỗ Xuân Chợ |

**HÀ NỘI 2025**

# Bài lab Các kỹ thuật giấu tin: lsb_ctf1

## 1. Mục đích

Giúp sinh viên hiểu được thuật toán giấu tin trong âm thanh sử dụng phương pháp LSB (Least Significant Bit), sử dụng các công cụ trong Linux để tìm kiếm và trích xuất file âm thanh được nối với file hình ảnh, từ đó lấy được thông điệp là một flag nằm trong cặp dấu {}.

## 2. Yêu cầu đối với sinh viên

Quen thuộc với hệ điều hành Linux và có kiến thức về kỹ thuật giấu tin.

## 3. Nội dung lý thuyết

Thuật toán giấu tin trong âm thanh sử dụng phương pháp LSB là phương pháp giấu tin bằng cách thay đổi **bit ít quan trọng nhất** của mỗi mẫu tín hiệu âm thanh mà không làm ảnh hưởng đáng kể đến chất lượng âm thanh.

Quy trình giấu tin: Trước tiên cần chia dữ liệu âm thanh gốc thành các đoạn. Thông thường, người giấu tin sẽ chia dữ liệu âm thanh thành các đoạn dựa trên độ dài bit của thông tin cần giấu. Sau đó các đoạn này được biến đổi thành vector giá trị của tín hiệu, rồi lưu vào mảng một chiều. Đối với thông tin cần giấu ($L$), chúng cũng được biến đổi nhị phân rồi tính chiều dài chuỗi bit $L$ cần giấu. Tiếp theo, người giấu tin sẽ chọn $k$ là số bit LSB của tín hiệu âm thanh sẽ giấu sao cho phù hợp nhất. Cuối cùng, chia chuỗi bit thông điệp thành các chuỗi con có độ dài $k$ bit. Trong đó, mỗi chuỗi con này sẽ được thay thế vào $k$ bit LSB của L/k tín hiệu âm thanh để giấu đủ $L$ bit thông điệp.

Trong bài lab này, thuật toán được mô tả như sau:

- **Đầu vào:**
  - Thông điệp bí mật (flag).
  - File âm thanh dùng để chứa flag.
  - File ảnh dùng để nối với file âm thanh.
- **Các bước thực hiện giấu thông điệp**
  - Bước 1: Chuyển đổi thông điệp thành dạng nhị phân.
  - Bước 2: Mở file MP3 ở chế độ nhị phân và chuyển dữ liệu thành byte array để có thể sửa đổi từng byte.
  - Bước 3: Giấu thông điệp vào LSB của từng byte trong MP3.
  - Bước 4: Ghép file PNG và MP3 lại với nhau.
- **Đầu ra:** File PNG đã nối với MP3, trong file MP3 có flag cần được trích xuất.

- Sinh viên sẽ được cung cấp file PNG đã qua các bước xử lý như trên và một file exploit.py để trích xuất flag. Sinh viên chỉ nên chạy file này sau khi đã hiểu hết các bước trong bài lab.

**4. Nội dung thực hành**

Khởi động bài lab:

Vào terminal, gõ:

*labtainer -r lsb_ctf1*

(Chú ý: Sinh viên sử dụng mã sinh viên của mình để nhập thông tin email người thực hiện bài lab khi có yêu cầu, để sử dụng khi chấm điểm)

**Nhiệm vụ 1: Xác định header của file âm thanh trong hình ảnh**

Chạy lệnh "file" để kiểm tra xem file PNG được cung cấp có thật sự là file PNG không, hay chỉ là 1 file đã được đổi extension.

*file Can_You_Find_Me.png*

Kết quả cho thấy tệp thực sự là một tệp PNG hợp lệ. Tuy nhiên, điều này không có nghĩa là nó không chứa dữ liệu ẩn. Vì vậy, sử dụng "binwalk" để kiểm tra cấu trúc tệp nhị phân, mục đích là xác định các định dạng file khác bên trong.

***binwalk Can_You_Find_Me.png***

Kết quả chứa địa chỉ "0x19F093", mô tả rằng có 1 tệp TIFF - một tệp ẩn hoặc 1 phần của dữ liệu file âm thanh được nhúng vào.

**Nhiệm vụ 2: Xác định meta data ứng với file âm thanh**

Sử dụng công cụ "hexedit" và tìm kiếm địa chỉ bắt đầu của file được nhúng vào hình ảnh.

***hexedit Can_You_Find_Me.png***

Dựa vào địa chỉ tìm được ở trên, nhấn tổ hợp phím "Ctrl + S" rồi nhập địa chỉ "0019F0", rồi Enter.

Tiếp tục Enter rồi nhập địa chỉ "19F0B0", sinh viên sẽ thấy có một chuỗi đặc biệt là "IEND", đây là chỉ báo kết thúc ảnh trong cấu trúc tệp PNG. Tuy nhiên, sau "IEND" vẫn còn thông tin, và thông tin đó bắt đầu bằng chuỗi "ID3", đây là một meta data của MP3, cho thấy rằng file PNG này đã được nối với một file MP3. Một điều quan trọng nữa cần thấy là địa chỉ của ID3 trong

trường hợp này là 0x19F0B1 (tương ứng với 1700017 trong hệ 10). Chúng ta sẽ cần thông tin này cho nhiệm vụ tiếp theo.

Thoát khỏi hexedit và đến nhiệm vụ tiếp theo.

**Nhiệm vụ 3: Xác định audio version**

Khi đã biết được địa chỉ và meta data của file âm thanh, sử dụng lệnh "dd" để trích xuất file âm thanh đó

*dd if=Can_You_Find_Me.png of=outfile bs=1 skip=1700017*

Giải thích:

| | |
|---|---|
| *if=Can_You_Find_Me.png* | Đọc dữ liệu từ file ảnh PNG |
| *of=outfile* | Ghi dữ liệu trích xuất ra file mới (outfile). |
| *bs=1* | Đọc và ghi từng byte một (độ chính xác cao). |
| *skip=1700017* | Bỏ qua 1.700.017 byte đầu tiên (tức là chỉ lấy dữ liệu bắt đầu từ byte 1.700.017 trở đi). |

Sau khi chạy lệnh dd, ta có file mới tên outfile. Ta kiểm tra xem nó có thực sự là file âm thanh không bằng lệnh file

***file outfile***

Nếu output là "outfile: Audio file with ID3 version 2.4.0"

=> Điều này xác nhận rằng outfile chính là một file MP3 chứa metadata ID3 version 2.4.0.

**Nhiệm vụ 4: Trích xuất thông điệp được giấu trong file âm thanh**

Sau khi đã hiểu cách hoạt động của bài lab, sinh viên sử dụng nano để xem cách file exploit hoạt động

*nano exploit.py*

Về cơ bản, script này sẽ đọc file PNG, tìm ra vị trí file MP3 bắt đầu, trích xuất file MP3 đó và sử dụng thuật toán LSB để trích xuất thông điệp từ file MP3.

Sinh viên chạy script được cung cấp để trích xuất thông điệp, kết quả hiện ra là thông điệp nằm trong cặp dấu {}.

***python3 exploit.py***

**Kết thúc bài lab:**

Kiểm tra checkwork:

*checkwork*

Trên terminal đầu tiên sử dụng câu lệnh sau để kết thúc bài lab:

*stoplab*

Khởi động lại bài lab:

*labtainer -r lsb_ctf1*