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

BÁO CÁO BÀI THỰC HÀNH
HỌC PHẦN: KỸ THUẬT GIẤU TIN
MÃ HỌC PHẦN: INT14102

NHÓM LỚP: D21CQAT03-B

TÊN BÀI: TẤN CÔNG ĐỌC TIN BỊ GIẤU BẰNG KỸ THUẬT FHSS TRONG TỆP ÂM THANH

Sinh viên thực hiện:

B21DCAT135 Đặng Quý Nam

Giảng viên: PGS.TS. Đỗ Xuân Chợ

HỌC KỲ 2 NĂM HỌC 2024-2025

MỤC LỤC

DANH MỤC CÁC HÌNH VẼ

DANH MỤC CÁC BẢNG BIỂU

CHƯƠNG 1: GIỚI THIỆU CHUNG VỀ BÀI THỰC HÀNH

1.1 Giới thiệu chung về bài thực hành

Bài thực hành này tập trung vào tấn công vào giấu tin trong âm thanh bằng cách sử dụng thuật toán nhảy tần – FHSS (Frequency Hopping Spread Spectrum).

Giấu tin trong file âm thanh bằng FHSS (Frequency Hopping Spread Spectrum) là một kỹ thuật mã hóa tín hiệu, nơi thông tin được giấu vào tần số của tín hiệu âm thanh. Cách thực hiện và tách thông tin này có thể được mô tả ngắn gọn như sau:

Cách giấu thông tin:

1. **Chia nhỏ thông tin cần giấu**: Thông tin (ví dụ: văn bản, hình ảnh) sẽ được mã hóa thành các bit.
2. **Tần số nhảy**: Sử dụng FHSS, thông tin được "nhảy" qua các tần số khác nhau trong tín hiệu âm thanh theo một dãy tần số ngẫu nhiên. Dãy tần số này được chọn sao cho nó chỉ có thể được giải mã bởi người nhận có thông tin về cách thức nhảy tần số.
3. **Gửi tín hiệu âm thanh**: Tín hiệu âm thanh chứa thông tin được ẩn giấu thông qua các tần số nhảy.

Cách tách thông tin:

1. **Xác định chu kỳ nhảy tần số**: Người nhận phải biết chính xác cách thức nhảy tần số (giống như người gửi).
2. **Lọc tần số**: Lọc và tái tạo tín hiệu âm thanh bằng cách sử dụng tần số nhảy đã biết, sau đó giải mã thông tin giấu bên trong.
3. **Giải mã**: Sau khi tách ra các phần tín hiệu, thông tin được giải mã từ các bit đã ẩn.

Kỹ thuật này mang lại sự bảo mật cao vì tần số nhảy thay đổi liên tục, khiến việc nhận diện và giải mã trở nên khó khăn nếu không biết chính xác cách thức nhảy tần số.

Bài thực hành thể hiện một file âm thanh được giấu thông điệp bên trong nhưng không thực hiện giả ngẫu nhiên các tần số giấu, dẫn đến bị tấn công đọc trộm tin đã giấu

1.2 Nội dung và hướng dẫn bài thực hành

1.2.1 Mục đích

Giúp sinh viên nắm bắt cách thức tách tin bằng FHSS và cách tấn công đơn giản vào FHSS.

1.2.2 Yêu cầu đối với sinh viên

- Nắm được nguyên lý hoạt động của thuật toán giấu tin Frequency Hopping Spread Spectrum (FHSS).
- Nắm được cách sử dụng công cụ audacity.

1.2.3 Nội dung thực hành

Khởi động bài lab, tải bài thực hành bằng imodule:

Imodule imodule.tar

Hoặc giải nén file imodule.tar vào foder trunk/labs
Vào terminal, gõ:

Labtainer -r stego-atk-fhss

(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)

Sau khi khởi động xong, màn hình sẽ xuất hiện một terminal đại diện cho bài thực hành

- Sinh viên được cung cấp một file âm thanh được giấu tin và file tách tin đang thiếu tham số đầu vào, nhiệm vụ của sinh viên phải tìm được các tham số quan trọng và tách được tin từ file.
- Sử dụng audacity lên file âm thanh, đọc file âm thanh dưới dạng đồ thị spectrogram, nhận ra các tần số được giấu trải đều một cách thẳng đều. Trả lời các câu hỏi sau từ đó tính toán các tham số:
    - Tìm tần số bắt đầu của đoạn giấu tin là bao nhiêu Hz?
    - Độ dài của giấu tin là bao nhiêu giây?
        - Với công thức: Thời gian của đoạn giấu tin (giây) = $\frac{\text{Độ dài thông điệp (bit)} * \text{Độ dài của từng frame}}{\text{Tần số lấy mẫu của file âm thanh}}$
    - Khoảng cách giữa các tần số giấu tin là bao nhiêu? Gợi ý: khoảng cách này nhỏ hơn 50.
- Điền các tham số tìm được vào file atk.py và chạy bằng lệnh python3 atk.py
- Thông điệp đọc viết vào file output.txt.
- Nếu nhận được thông điệp thì sinh viên đã hoàn thành bào lab.
- Trên terminal đầu tiên sử dụng câu lệnh sau để kết thúc bài lab:
  stoplab

Khi bài lab kết thúc, một tệp zip lưu kết quả được tạo và lưu vào một vị trí được hiển thị bên dưới stoplab.

Khởi động lại bài lab:

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

labtainer –r stego-atk-fhss

CHƯƠNG 2: PHÂN TÍCH YÊU CẦU BÀI THỰC HÀNH

2.1 Thiết kế bài thực hành

Bài lab gồm 1 container đại diện cho máy làm bài.

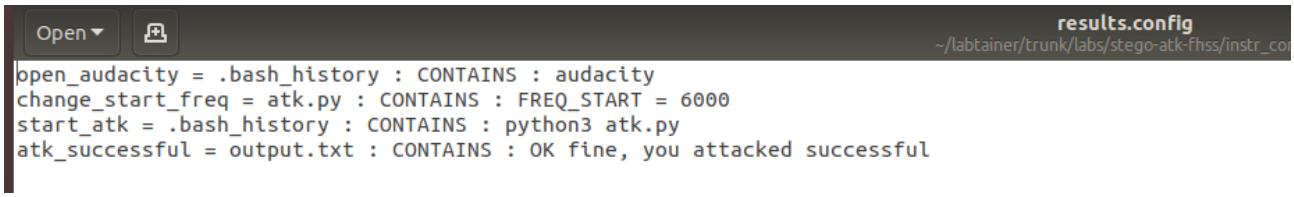
Cấu hình Docker

- Bài lab chạy với image là .base2
- Cần cài thêm thư viện numpy, scipy trong dockerfiles.
- Docs lưu lại hướng dẫn thực hành cho sinh viên

Các nhiệm vụ cần thực hiện để thành công:

- Sử dụng audacity để đọc file âm thanh.
- Điền tham số vào mã khai thác
- Tiến hành tấn công.
- Tấn công thành công và nhận được thông điệp.
- Kết thúc bài lab và đóng gói kết quả

Để đánh giá được sinh viên đã hoàn thành bài thực hành hay chưa, cần chia bài thực hành thành các nhiệm vụ nhỏ, mỗi nhiệm vụ cần phải chỉ rõ kết quả để có thể dựa vào đó đánh giá, chấm điểm. Do vậy, trong bài thực hành này hệ thống cần ghi nhận các thao tác, sự kiện được mô tả và cấu hình như bảng dưới đây:



Hình 1 File result.config

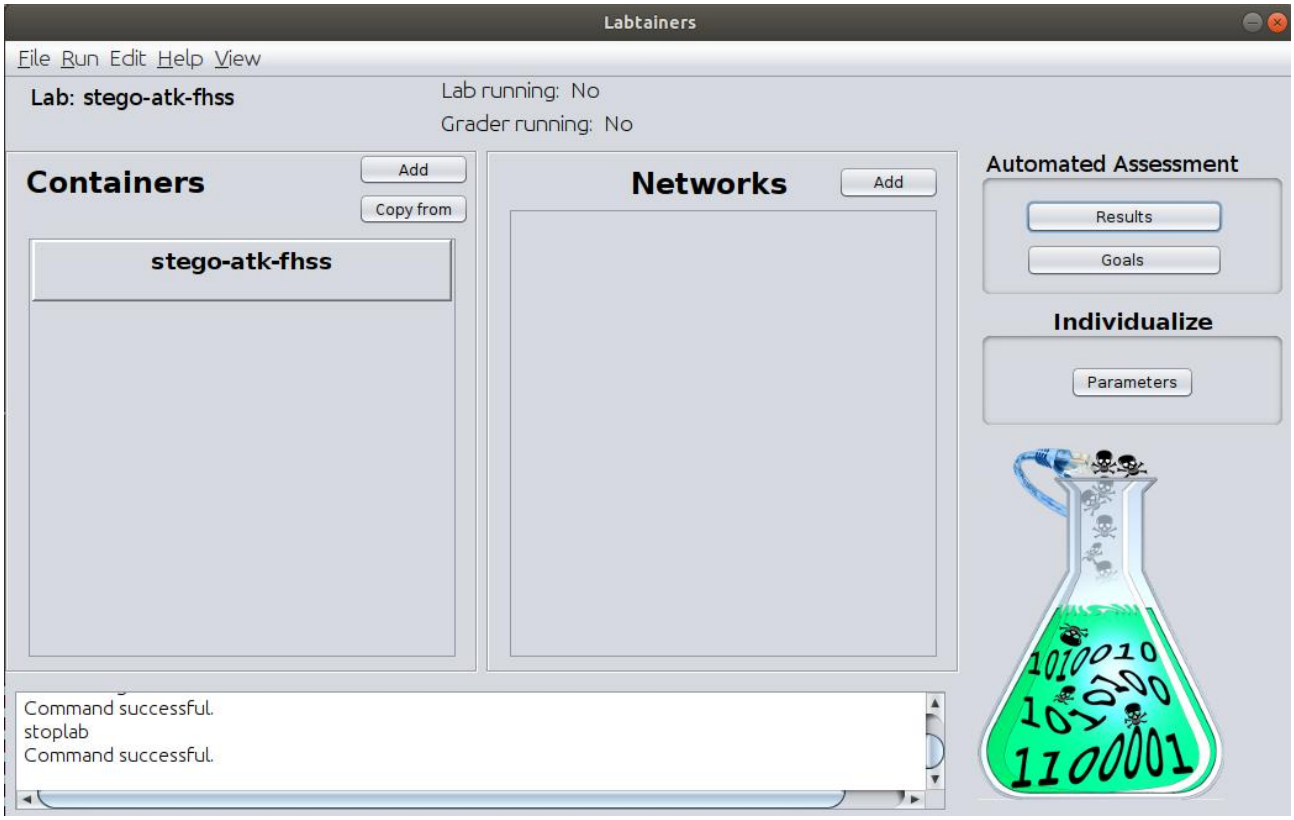
open_audacity : Khởi động audacity

change_start_freq : Kiểm tra điểm bắt đầu của tần số đúng hay chưa

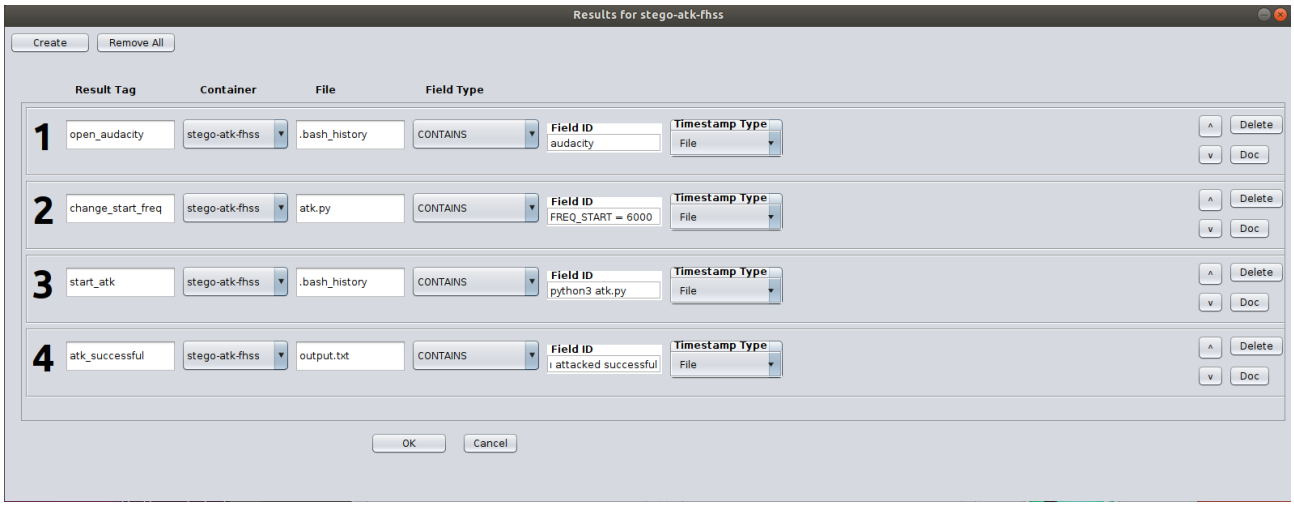
start_atk: Kiểm tra đã bắt đầu tấn công hay chưa

atk_successful: Kiểm tra thông điệp đã tách ra đúng hay chưa

2.2 Cài đặt và cấu hình máy ảo



Hình 2 Giao diện labedit



Hình 3 Giao diện result.config

2.3 Tích hợp và triển khai

Bài thực hành được tiển khai như sau:

Docker

Đường dẫn: Docker Hub

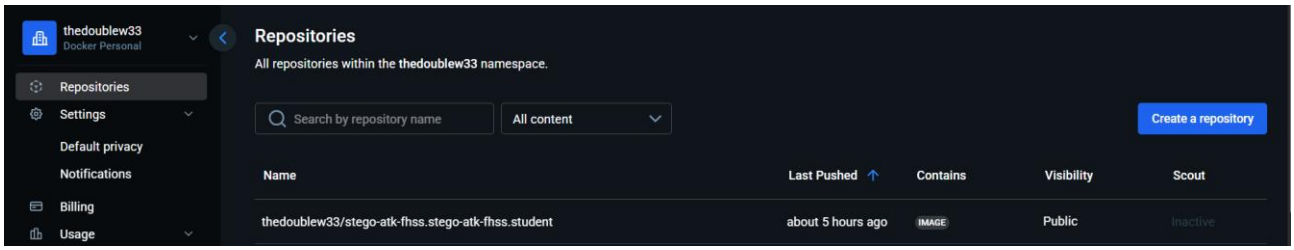
Thêm registry cho bài thực hành

Truy cập vào thư mục trunk/distrib gõ lệnh: *docker login* đăng nhập tài khoản DockerHub

Sử dụng lệnh *./publish.py -d -l stego-atk-fhss* để đẩy images của bài thực hành lên DockerHub

```
student@ubuntu:~/labtainer/trunk/distrib$ ./publish.py -d -l stego-atk-fhss
adding [nmaplab]
adding [httplab]
adding [liveforensics]
adding [bind-shell]
adding [tlab]
adding [metasploitable-test]
adding [kali-test]
adding [my-remote-dns]
adding [remote-dns2]
adding [remote-dns]
adding [backups]
adding [centos-log]
adding [dhcp-test]
adding [xlab]
adding [softplc]
adding [iptables]
adding [grfics]
adding [usbtest]
adding [ida]
adding [centossix]
adding [routing-basics2]
adding [shellbasics]
adding [ldaptst]
adding [mariadbtst]
```

Hình 4 Đẩy image bài thực hành lên docker hub



Hình 5 Image trên dockerhub

Github

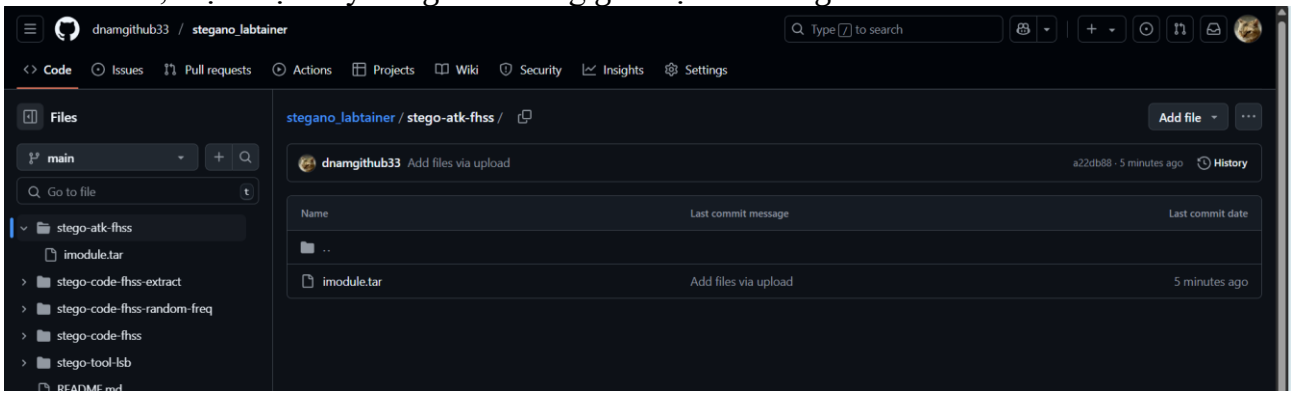
Đường dẫn: [tại đây]

Ở đường dẫn $LABTAINER_DIR/distrib, tạo file tar bằng create-imodules.sh

```
student@ubuntu:~/labtainer/trunk/scripts/designer/bin$ ./create-imodules.sh
lab is stego-atk-fhss
Do docs
************************************************************
**  Post /home/student/labtainer/trunk/imodule.tar to your web server **
************************************************************
student@ubuntu:~/labtainer/trunk/scripts/designer/bin$
```

Hình 6 Tạo file imodule

Sau đó, thực hiện đẩy lên github bằng git hoặc thủ công

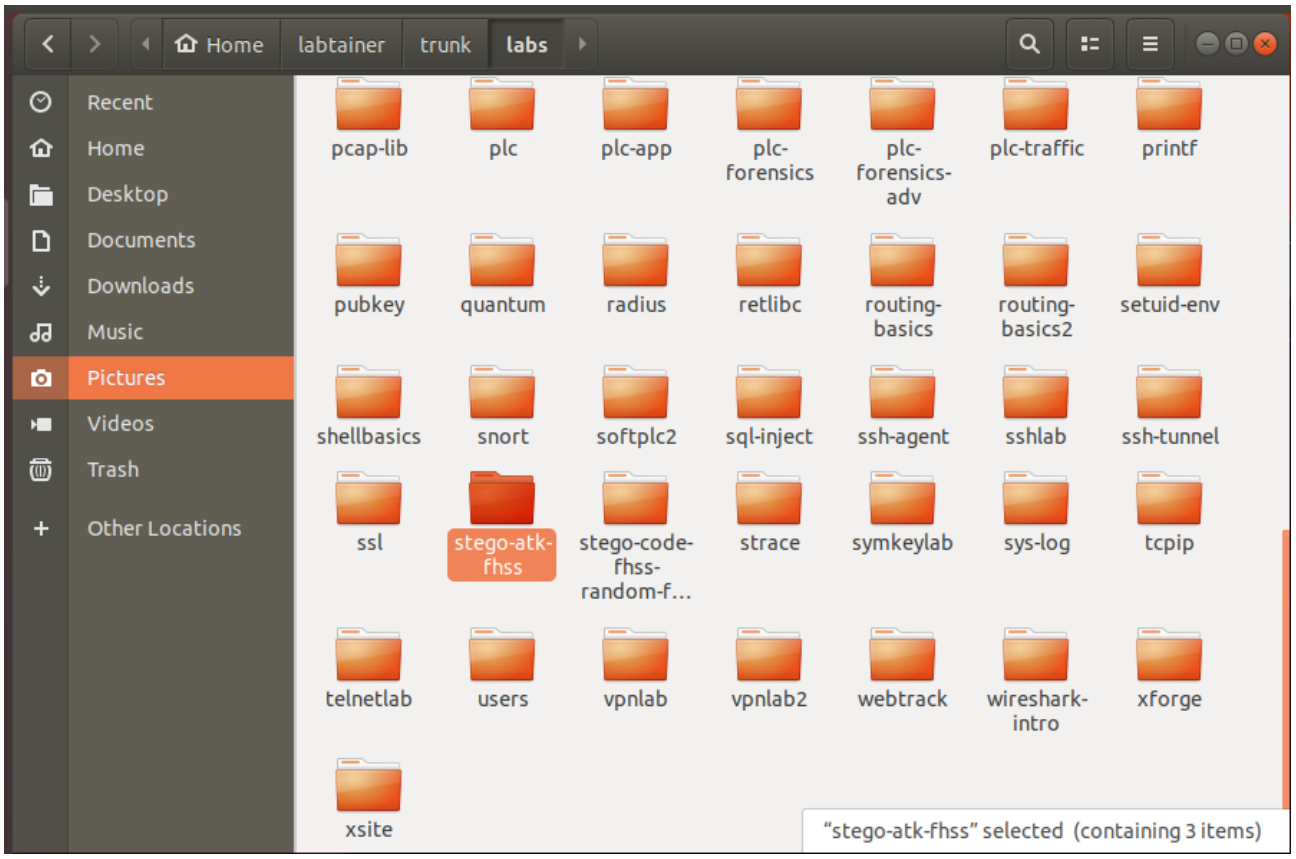


Hình 7 File imodule trên github

CHƯƠNG 3: THỬ NGHIỆM VÀ ĐÁNH GIÁ.

Bài thực hành đã được xây dựng thành công, dưới đây là hình ảnh minh họa về bài thực hành:

Khởi động bài lab, tải bài thực hành bằng imodule, trong trường hợp không dùng được imodule, giải nén file thực hành trong folder labtainer/trunk/labs:



Hình 8 Cài đặt bài thực hành

Vào terminal, gõ:

Labtainer -r stego-atk-fhss

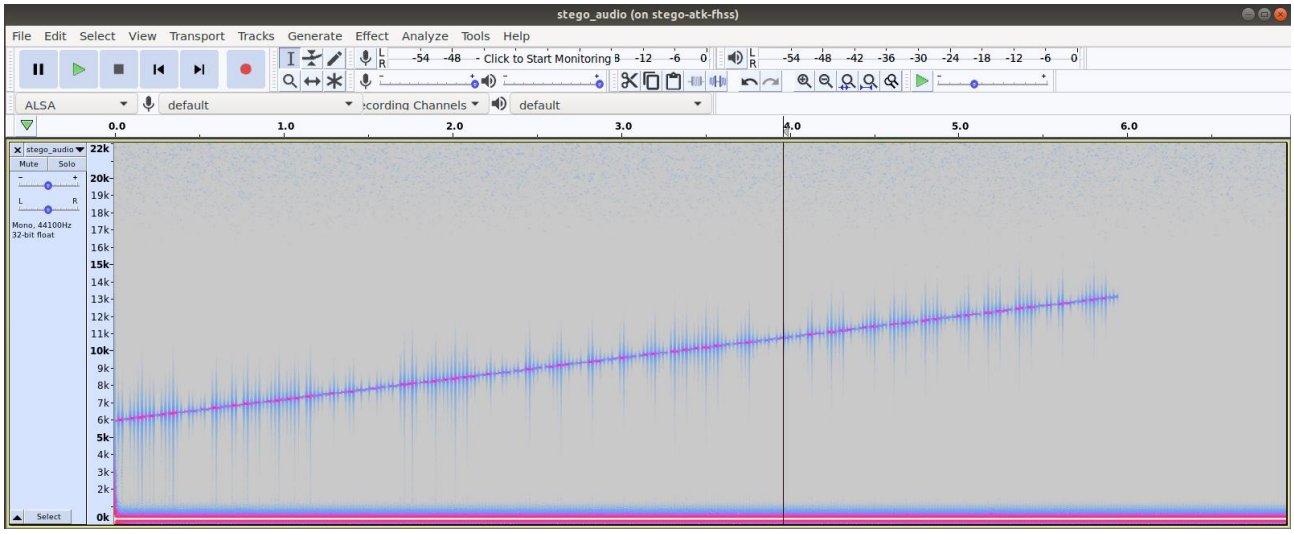
(chú ý: sinh viên sử dụng mã sinh viên của mình để nhập thông tin email người thực hiện bài lab khi có yêu cầu, để sử dụng khi chấm điểm)

```
student@ubuntu:~/labtainer/labtainer-student$ labtainer stego-atk-fhss
latest: Pulling from thedoublew33/stego-atk-fhss.stego-atk-fhss.student
72102740e68a: Pull complete
fc55c2cbce9e: Pull complete
8e7e24552aa8: Pull complete
b86a7c31d689: Pull complete
aa645b030694: Pull complete
7944fb2df358: Pull complete
297c6041672c: Pull complete
5c9f7b15d503: Pull complete
cf73ff9c361d: Pull complete
2bb283d72eff: Pull complete
ee3d6a2d68fc: Pull complete
eebcea2cb90c: Pull complete
a7ddb4f4b4d8: Pull complete
Digest: sha256:17d5bf9343e89203e2a690dfa9f84e0d6326be9922dbb0e335fd63e94fc478b5
Status: Downloaded newer image for thedoublew33/stego-atk-fhss.stego-atk-fhss.student:latest
non-network local connections being added to access control list

Please enter your e-mail address:  [b21dcat135]
Started 1 containers, 1 completed initialization. Done.
```

Hình 9 Khởi động badi lab

- Đọc file âm thanh bằng audacity



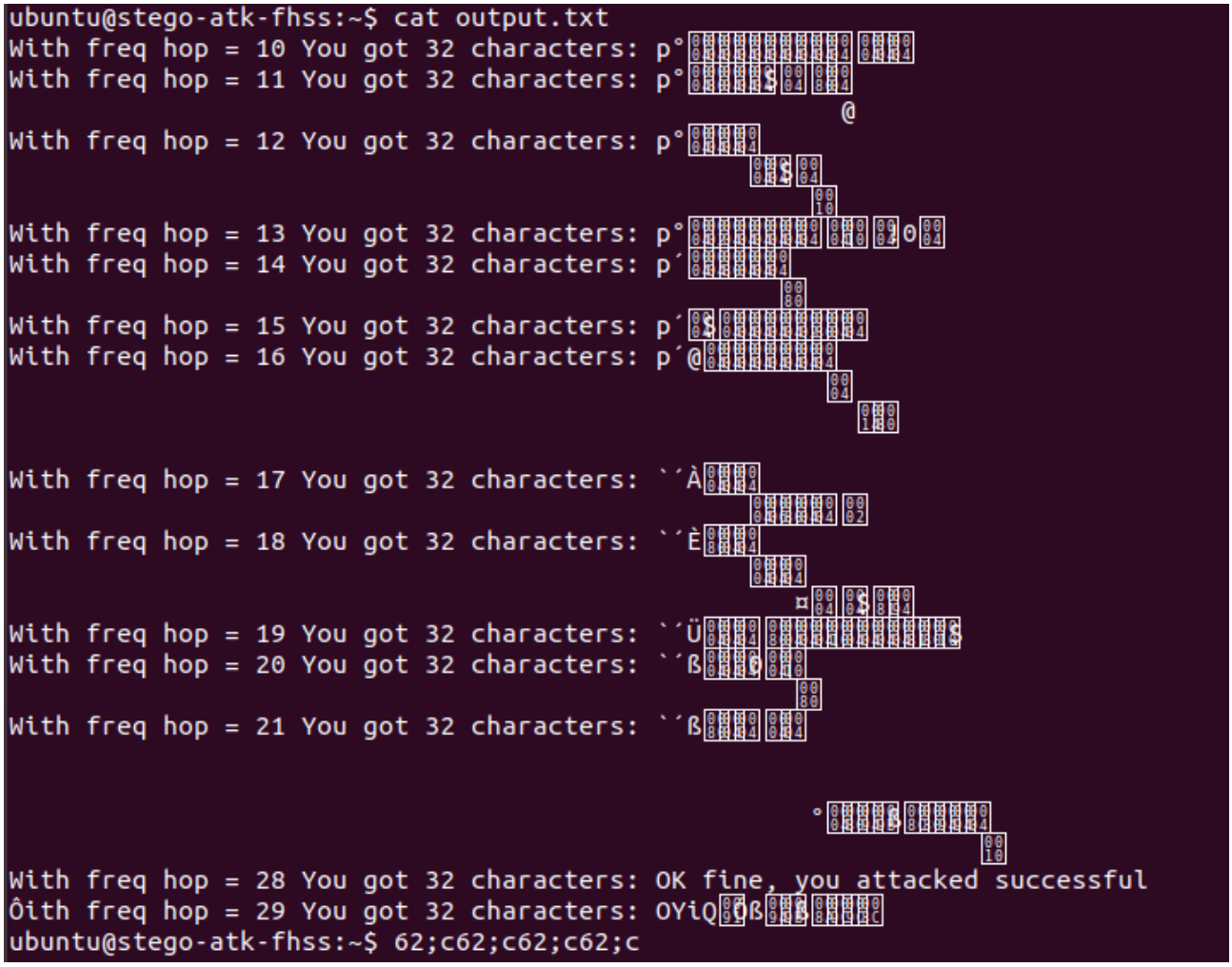
Hình 10 Đọc file âm thanh bằng audacity

- Sau khi tìm được các tham số tần số bắt đầu là 6000, độ dài tin giấu là 32 ký tự ,điền vào file atk.py và tiến hành tấn công.

```
ubuntu@stego-atk-fhss:~$ nano atk.py
ubuntu@stego-atk-fhss:~$ python3 atk.py
Done
ubuntu@stego-atk-fhss:~$
```

Hình 11 Tấn công

- Tìm thông điệp trong file output.txt:



Hình 12 Tấn công thành công

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

  stoplab

  Khi bài lab kết thúc, một tệp zip lưu kết quả được tạo và lưu vào một vị trí được hiển thị bên dưới stoplab.

  Kết quả checkwork

```
student@ubuntu:~/labtainer/labtainer-student$ checkwork
Results stored in directory: /home/student/labtainer_xfer/stego-atk-fhss
Labname stego-atk-fhss

Student             |   open_audacity | change_start_fr |       start_atk |  atk_successful |
==================== | =============== | =============== | =============== | =============== |
b21dcat135          |               Y |               Y |               Y |               Y |
What is automatically assessed for this lab:
```

Hình 13 Checkwork

Khởi động lại bài lab:

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

labtainer –r stego-atk-fhss

TÀI LIỆU THAM KHẢO

[1] Bài giảng Các kỹ thuật giấu tin, PGS. TS Đỗ Xuân Chợ