

**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: GIẤU TIN TRONG TỆP ÂM THANH
BẰNG KỸ THUẬT FHSS**

Sinh viên thực hiện:

B21DCAT135 Đặng Quý Nam

Giảng viên: PGS.TS. Đỗ Xuân Chợt

HỌC KỲ 2 NĂM HỌC 2024-2025

MỤC LỤC

DANH MỤC CÁC HÌNH VẼ	3
DANH MỤC CÁC BẢNG BIỂU	3
CHƯƠNG 1: GIỚI THIỆU CHUNG VỀ BÀI THỰC HÀNH	4
1.1 Giới thiệu chung về bài thực hành	4
1.2 Nội dung và hướng dẫn bài thực hành	4
CHƯƠNG 2: PHÂN TÍCH YÊU CẦU BÀI THỰC HÀNH	6
2.1 Thiết kế bài thực hành	6
2.2 Cài đặt và cấu hình máy ảo	7
2.3 Tích hợp và triển khai	7
CHƯƠNG 3: THỬ NGHIỆM VÀ ĐÁNH GIÁ.	11
TÀI LIỆU THAM KHẢO	16

DANH MỤC CÁC HÌNH VẼ

Hình 1 File result.config.....	6
Hình 2 File goals.config.....	6
Hình 3 Giao diện labedit	7
Hình 4 Giao diện result.config	7
Hình 5 Giao diện goal.config	7
Hình 6 Đẩy image bài thực hành lên docker hub	9
Hình 7 Image trên dockerhub	9
Hình 8 Tạo file imodule	10
Hình 9 Đẩy file imodule lên github.....	10
Hình 10 Cài đặt bài thực hành.....	11
Hình 11 Khởi động badi lab	11
Hình 12 Sửa thông điệp.....	12
Hình 13 Tiến hành giấu tin.....	12
Hình 14 Cài đặt audacity.....	12
Hình 15 Theo dõi 2 file âm thanh trong audacity.....	13
Hình 16 Theo dõi 2 file âm thanh dưới dạng đồ thị spectrogram.....	13
Hình 17 Đồ thị Plot Spectrum	14
Hình 18 Kết quả sau khi export.....	14
Hình 19 Checkwork.....	15

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 kỹ thuật tách tin giấu 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.

1.2 Nội dung và hướng dẫn bài thực hành

1.2.1 Mục đích

Giúp sinh viên nắm bắt cách thức giấu tin bằng FHSS và cách sử dụng công cụ điều chỉnh âm thanh audacity.

1.2.2 Yêu cầu đối với sinh viên

- Hiểu rõ khái niệm giấu tin và vai trò của giấu tin trong âm thanh.
- Nắm được nguyên lý hoạt động của thuật toán giấu tin Frequency Hopping Spread Spectrum (FHSS).

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 folder trunk/labs

Vào terminal, gõ:

Labtainer -r stego-code-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 1 terminal để làm bài thực hành

- Để phục vụ cho bài thực hành, sinh viên được cung cấp một file code python **hide.py** thực hiện giấu tin vào file âm thanh **original_audio.wav**.
- Trước khi giấu tin, tiến hành sửa thông điệp muốn giấu vào file **message.txt** mã sinh viên của mình (ví dụ: B21DCAT001).
- Sinh viên sử dụng lệnh: **python3 hide.py** để tiến hành giấu tin.
- Sau khi giấu tin xong, xuất hiện thêm file **stego_audio.wav** là file đã được giấu tin.
- Sinh viên cài đặt audacity bằng lệnh **sudo apt install audacity -y**
- Sinh viên sử dụng công cụ audacity để mở 2 file âm thanh đang có.
- Audacity sẽ hiển thị 2 file âm thanh dưới dạng biểu đồ **waveform**, nhấn vào dấu 3 chấm ở mỗi file, chuyển biểu đồ sang dạng **spectrogram**.
- Sinh viên chụp lại trong báo cáo sự khác nhau của 2 biểu đồ này.
- Tiếp theo, để xem tin được giấu ở vùng tần số nào, sinh viên chọn **Analyze -> Plot spectrum**.
- Xuất 2 biểu đồ dưới dạng text với tên **spectrum_original.txt** với file âm thanh gốc và **spectrum_stego.txt** với file âm thanh đã giấu tin.
- 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-code-fhss

CHƯƠNG 2: PHÂN TÍCH YÊU CẦU BÀI THỰC HÀNH

2.1 Thiết kế bài thực hành

Bài lab gồm 1 container là ubuntu.

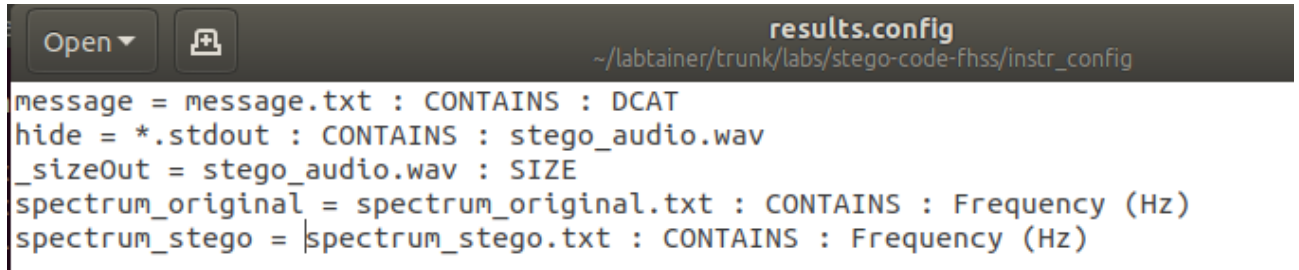
Cấu hình Docker

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

Các nhiệm vụ cần thực hiện để thành công:

- Viết thông điệp giấu tin
- Giấu tin vào file âm thanh có sẵn
- So sánh 2 file âm thanh: file âm thanh gốc và sau khi giấu tin
- 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

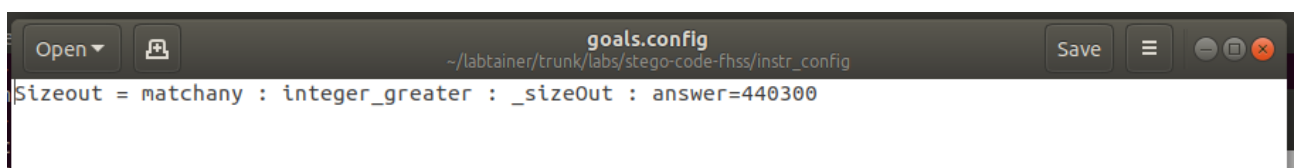
message : Tiến hành kiểm tra đã thêm mã sinh viên vào thông điệp hay chưa

hide : Kiểm tra đã thực hiện giấu tin hay chưa

_sizeOut : Lấy kích thước file giấu tin dưới dạng số nguyên (Kb)

spectrum_original: Kiểm tra đã xuất đồ thị plot spectrum của file âm thanh gốc hay chưa.

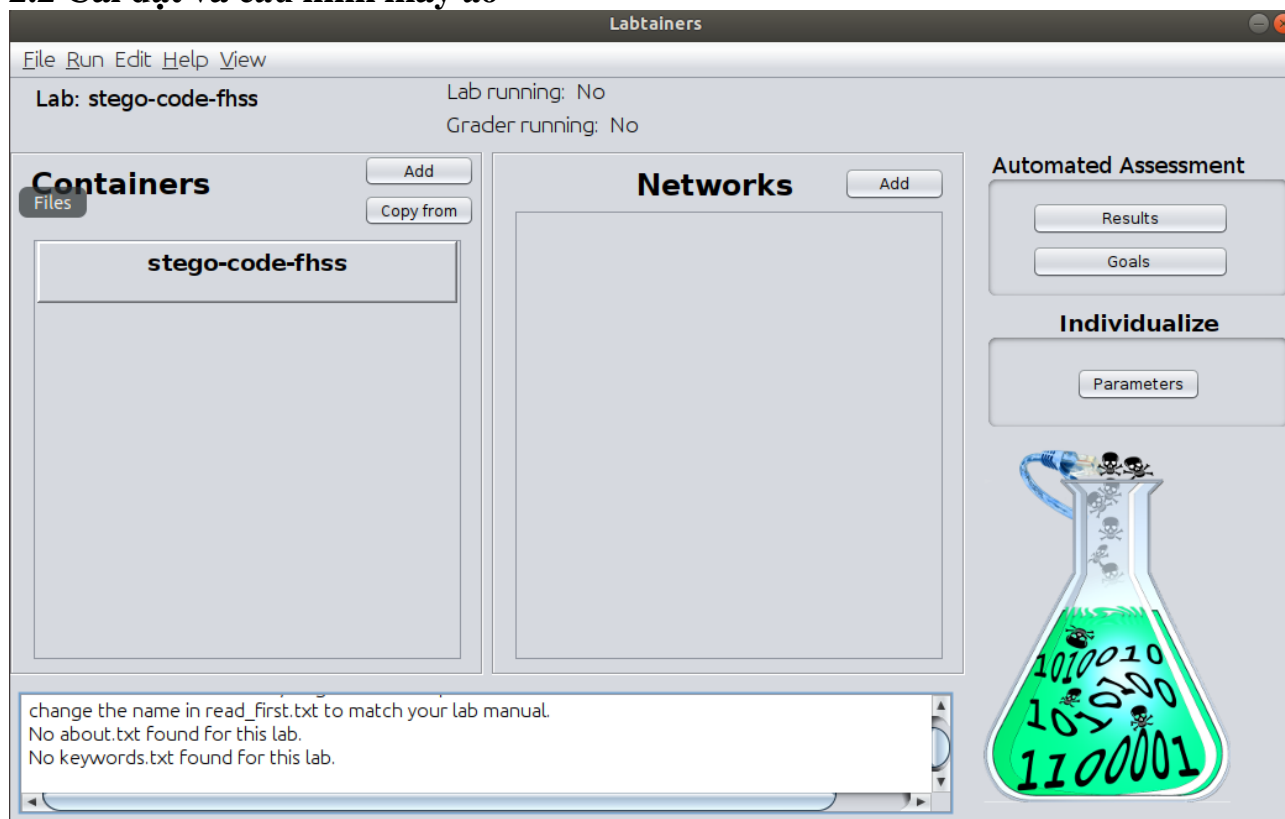
spectrum_stego: Kiểm tra đã xuất đồ thị plot spectrum của file âm thanh đã giấu tin hay chưa



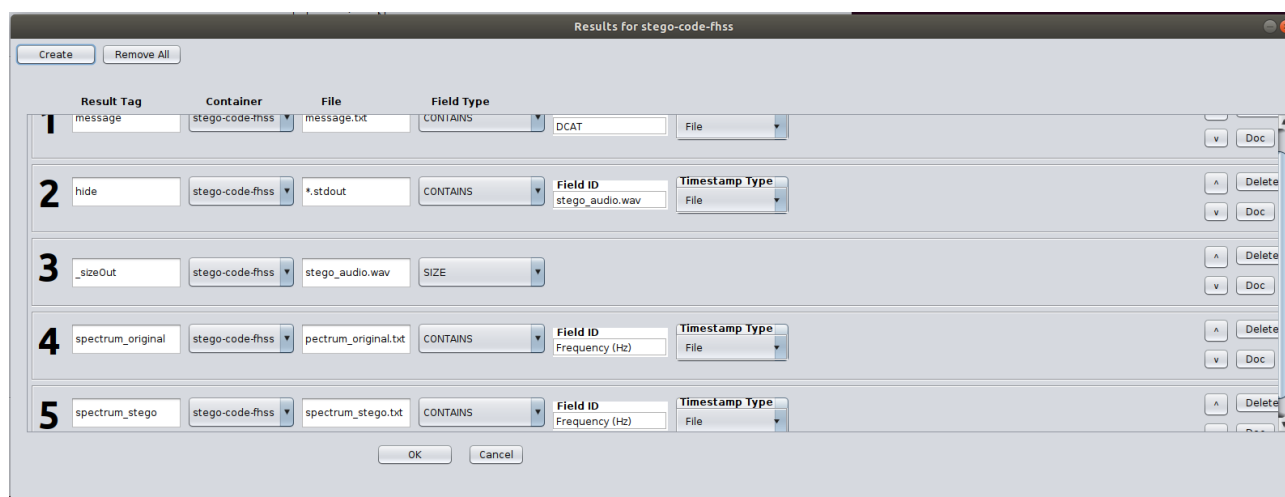
Hình 2 File goals.config

Sizeout: Kiểm tra kích thước file giấu tin.

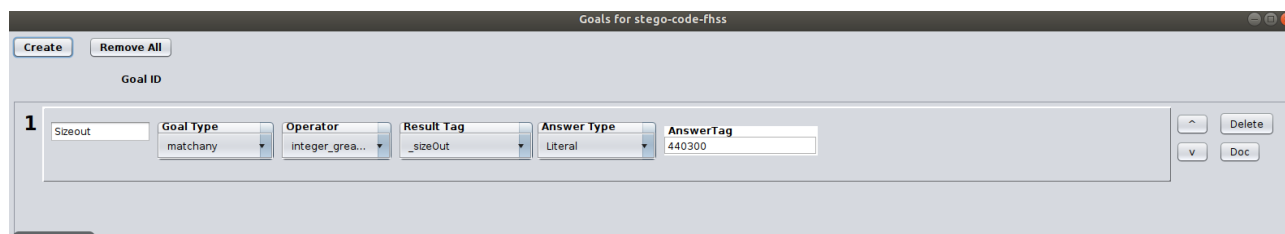
2.2 Cài đặt và cấu hình máy ảo



Hình 3 Giao diện labedit



Hình 4 Giao diện result.config



Hình 5 Giao diện goal.config

2.3 Tích hợp và triển khai

Bài thực hành được triển 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

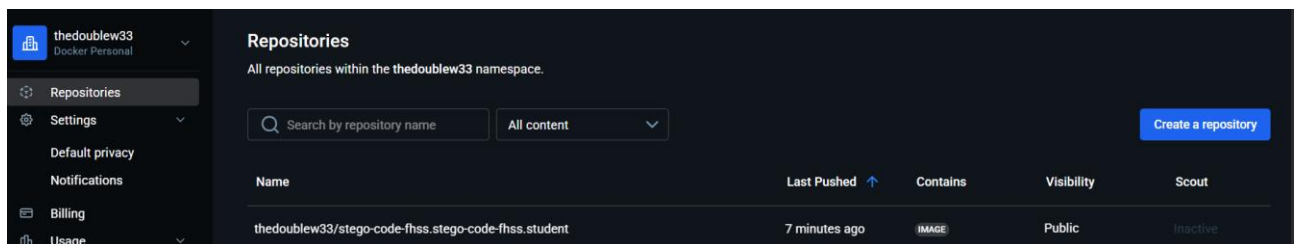
Sử dụng lệnh *./publish.py -d -l stego-code-fhss* để đẩy images của bài thực hành lên DockerHub


```

student@ubuntu:~/labtainer/trunk/distrib$ ./publish.py -d -l stego-code-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 [mariadbtest]
369098662ce5
1d37898af2b6
Untagged: stego-code-fhss.stego-code-fhss.student:latest
Deleted: sha256:b7d37b06727c695766389d20a21aa3f7f7aa3e0eedb7b52cb2b0e38f6028ba53
Deleted: sha256:31cdddef65db1a82fafbf02f2651e39e90e0a3e9e9a4112271d82b7aa69b4314
Untagged: stego-code-fhss.stego-code-fhss.student.tmp:latest
Deleted: sha256:2dd8ae7cf0a63c2a9d5e0a707a261d9381888f84356dcfc2edfe00f8db9c5822
Deleted: sha256:ba73d6107c4c210f6c213bcf39560008f8a45b6c29b9cddb003f3c59817cbab2
Deleted: sha256:a8b2d6c83ed8f5b9714886f872e9556bbe702f66b18add766e59c51cc3bfa16b
Deleted: sha256:883e4eb78df0d6738d57bff12daea8d82393d04d115b2f51ffdd67dd5d15d265
Deleted: sha256:38227f3dd6093616a2dcf5b2374ff2d6125b53ca83da8445e5ab23cb665f0324
Deleted: sha256:6a18f39a70018c4c0ca59d958a0588bf731f15651934d539de0bc01f35fcc4d3
Deleted: sha256:d9bd143e3b59db89a3804718582019437dfc40c510267182ad2399636642eac4
Deleted: sha256:3a97b0f9d46debfa5110b47a10ef85d497eafd57f293710dc0d17abcf9d3169
Deleted: sha256:45fc2a02b1e63126433a0e5a0f209d5827453293c8b522fd225156935b58e6ca
Deleted: sha256:7617a35b8a47f7517e59fc2d505a98100ab93d2c833d4444e6c20e269c1b6702

```

Hình 6 Đẩy image bài thực hành lên docker hub



Hình 7 Image trên dockerhub

Github

Đường dẫn:

https://github.com/dnamgithub33/stegano_labtainer/tree/e011f70cd9394a0cffb57067eedde409e2fd7027/stego-code-fhss

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

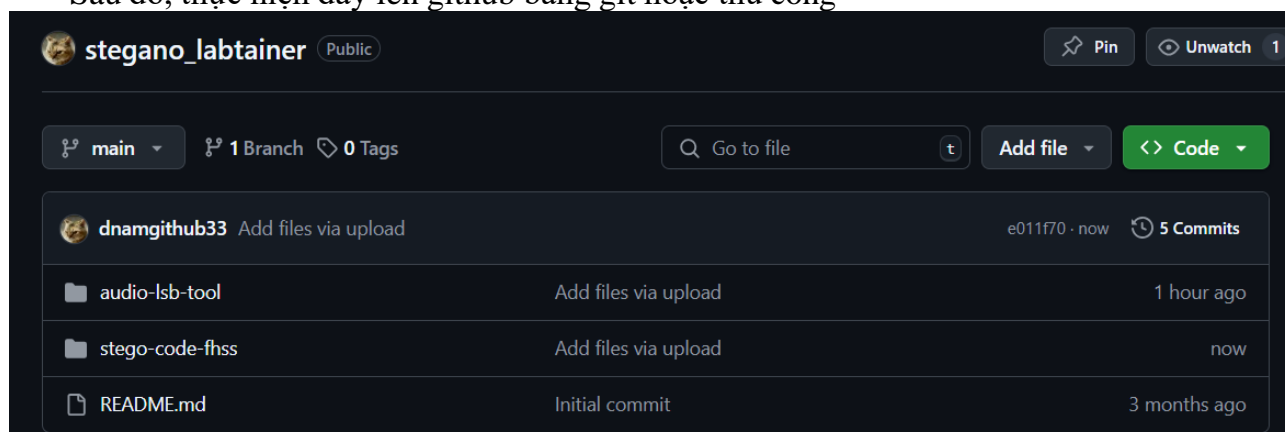
```

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

```

Hình 8 Tạo file imodule

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



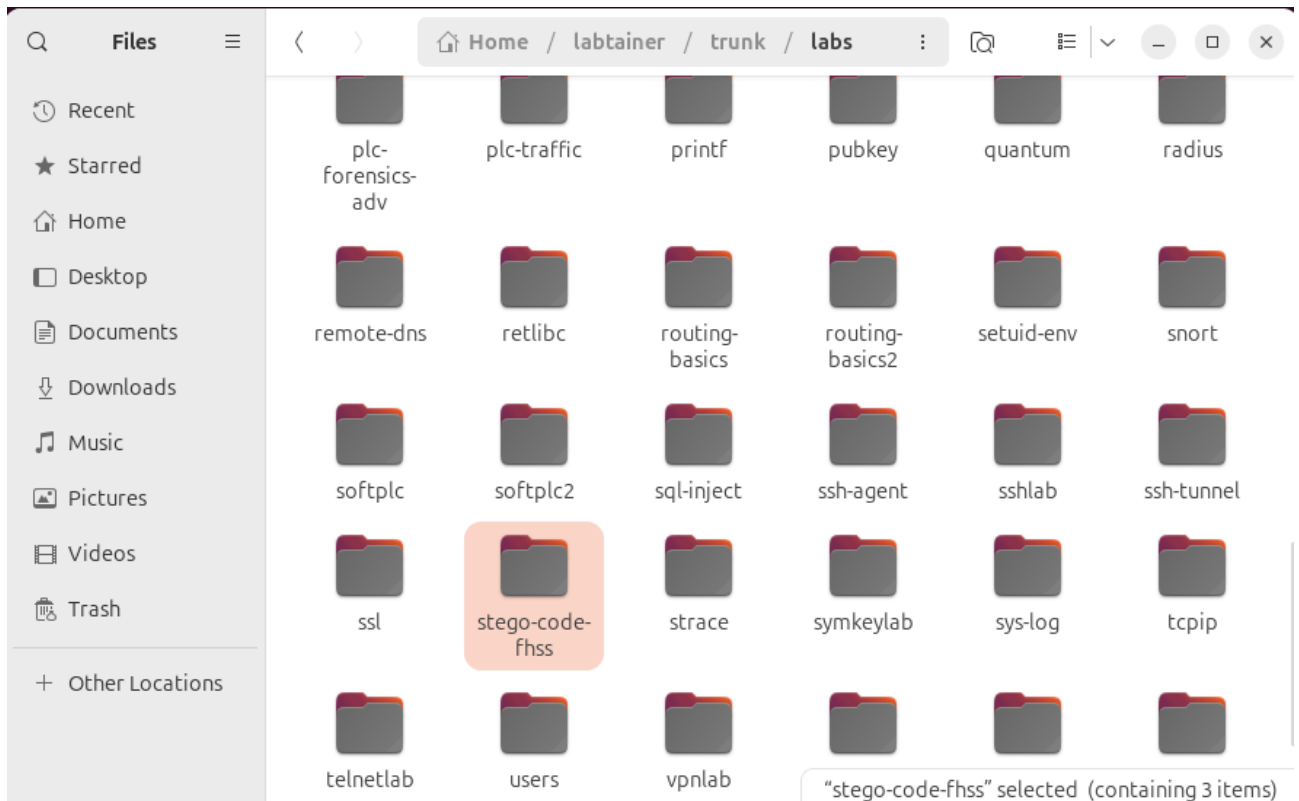
Hình 9 Đẩy file imodule lên github

File fhss.tar chứa bài thực hành

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 10 Cài đặt bài thực hành

Vào terminal, gõ:

```
Labtainer -r stego-code-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@LabtainerVMware:~/labtainer/labtainer-student$ labtainer stego-code-fhss
non-network local connections being added to access control list

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

The lab manual is at
file:///home/student/labtainer/trunk/labs/stego-code-fhss/docs/stego-code-fhss.pdf

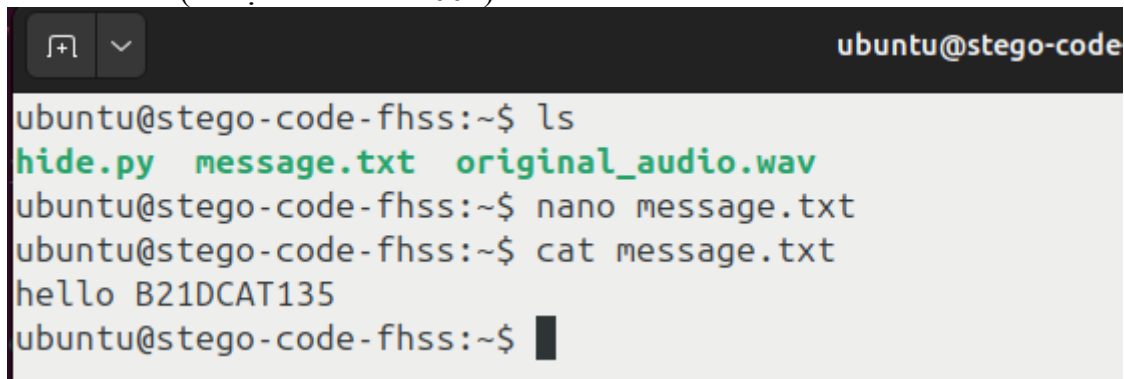
You may open these by right clicking
and select "Open Link".

Press <enter> to start the lab
```

Hình 11 Khởi động bài lab

- Trước khi giấu tin, tiến hành sửa thông điệp muốn giấu vào file **message.txt** mã sinh

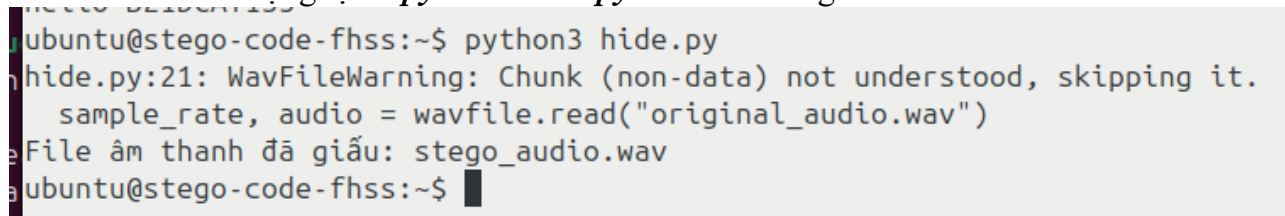
viên của mình (ví dụ: B21DCAT001).



```
ubuntu@stego-code
ubuntu@stego-code-fhss:~$ ls
hide.py message.txt original_audio.wav
ubuntu@stego-code-fhss:~$ nano message.txt
ubuntu@stego-code-fhss:~$ cat message.txt
hello B21DCAT135
ubuntu@stego-code-fhss:~$
```

Hình 12 Sửa thông điệp

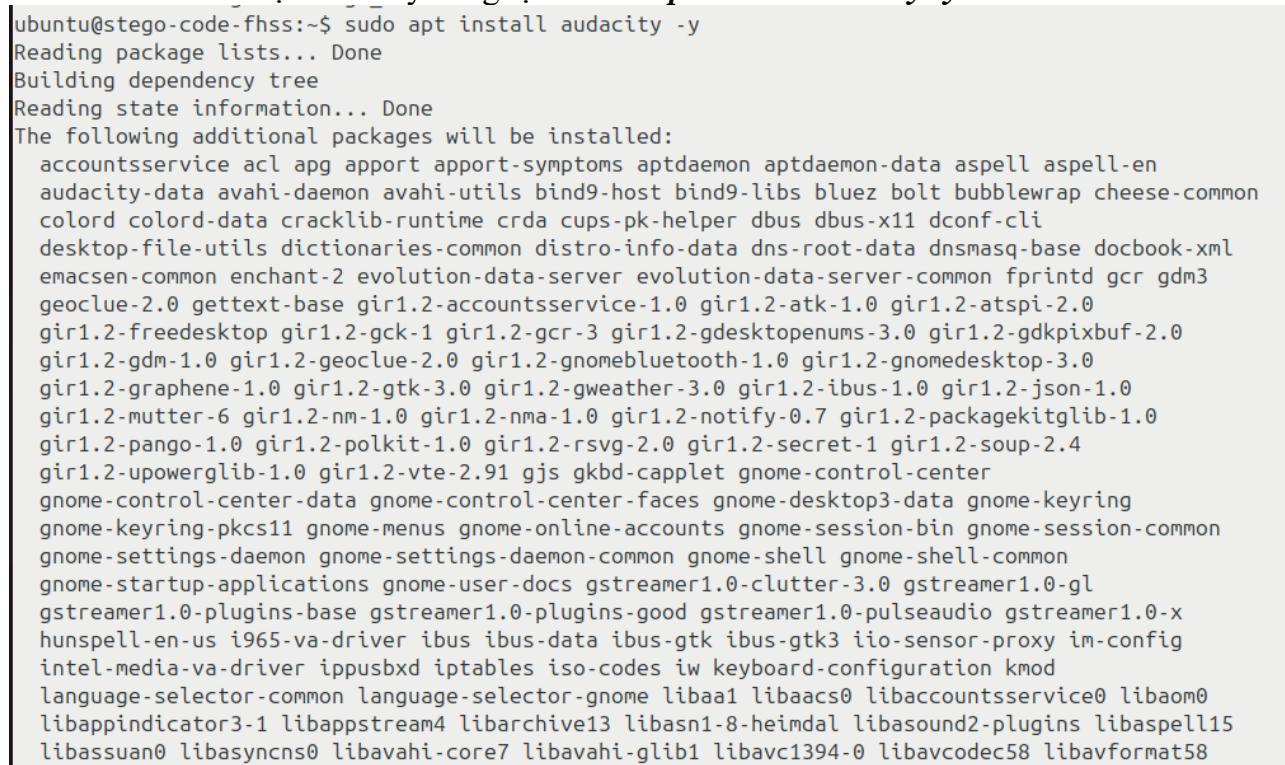
- Sinh viên sử dụng lệnh: **python3 hide.py** để tiến hành giấu tin.



```
ubuntu@stego-code-fhss:~$ python3 hide.py
hide.py:21: WavFileWarning: Chunk (non-data) not understood, skipping it.
  sample_rate, audio = wavfile.read("original_audio.wav")
File âm thanh đã giấu: stego_audio.wav
ubuntu@stego-code-fhss:~$
```

Hình 13 Tiến hành giấu tin

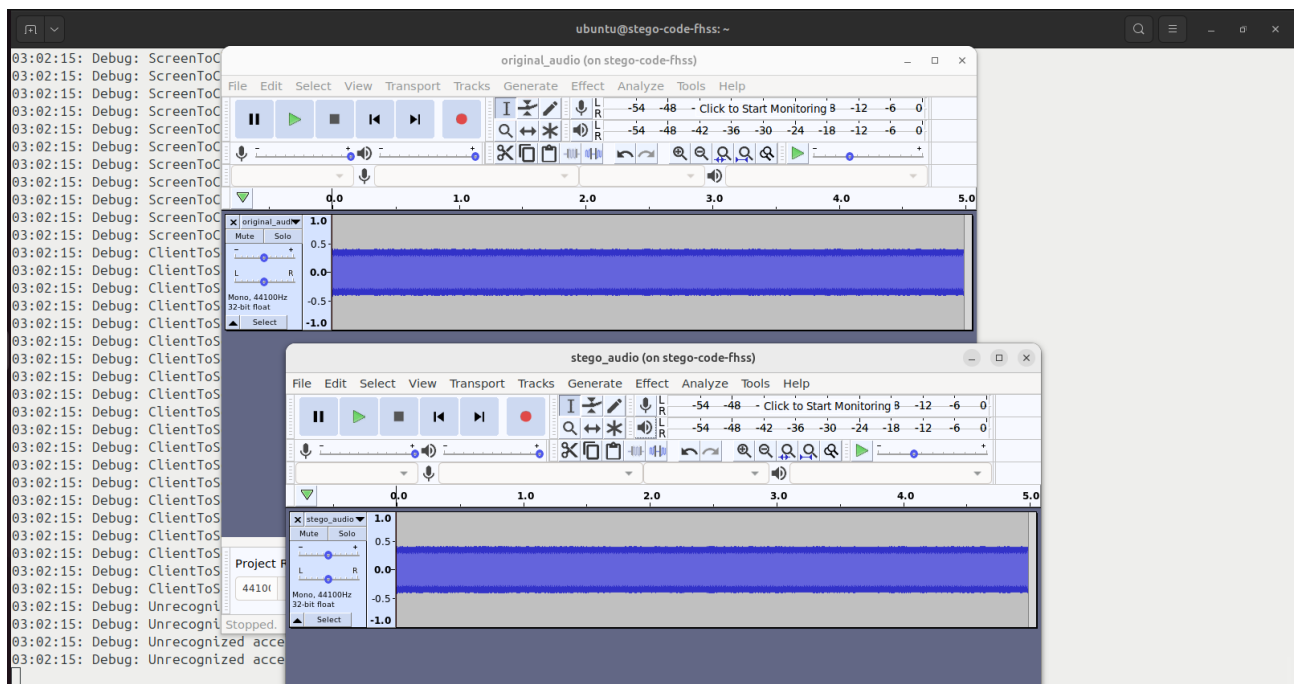
- Sau khi giấu tin xong, xuất hiện thêm file **stego_audio.wav** là file đã được giấu tin.
- Sinh viên cài đặt audacity bằng lệnh **sudo apt install audacity -y**



```
ubuntu@stego-code-fhss:~$ sudo apt install audacity -y
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  accountsservice acl apg apport apport-symptoms aptdaemon aptdaemon-data aspell aspell-en
  audacity-data avahi-daemon avahi-utils bind9-host bind9-libs bluez bolt bubblewrap cheese-common
  colord colord-data cracklib-runtime crda cups-pk-helper dbus dbus-x11 dconf-cli
  desktop-file-utils dictionaries-common distro-info-data dns-root-data dnsmasq-base docbook-xml
  emacs-common enchant-2 evolution-data-server evolution-data-server-common fprintd gcr gdm3
  geoclue-2.0 gettext-base gir1.2-accountsservice-1.0 gir1.2-atk-1.0 gir1.2-atspi-2.0
  gir1.2-freedesktop gir1.2-gck-1 gir1.2-gcr-3 gir1.2-gdesktopenums-3.0 gir1.2-gdkpixbuf-2.0
  gir1.2-gdm-1.0 gir1.2-geoclue-2.0 gir1.2-gnomebluetooth-1.0 gir1.2-gnomedesktop-3.0
  gir1.2-graphene-1.0 gir1.2-gtk-3.0 gir1.2-gweather-3.0 gir1.2-ibus-1.0 gir1.2-json-1.0
  gir1.2-mutter-6 gir1.2-nm-1.0 gir1.2-nma-1.0 gir1.2-notify-0.7 gir1.2-packagekitglib-1.0
  gir1.2-pango-1.0 gir1.2-polkit-1.0 gir1.2-rsvg-2.0 gir1.2-secret-1 gir1.2-soup-2.4
  gir1.2-upowerglib-1.0 gir1.2-vte-2.91 gjs gkbd-capplet gnome-control-center
  gnome-control-center-data gnome-control-center-faces gnome-desktop3-data gnome-keyring
  gnome-keyring-pkcs11 gnome-menus gnome-online-accounts gnome-session-bin gnome-session-common
  gnome-settings-daemon gnome-settings-daemon-common gnome-shell gnome-shell-common
  gnome-startup-applications gnome-user-docs gstreamer1.0-clutter-3.0 gstreamer1.0-gl
  gstreamer1.0-plugins-base gstreamer1.0-plugins-good gstreamer1.0-pulseaudio gstreamer1.0-x
  hunspell-en-us i965-va-driver ibus ibus-data ibus-gtk ibus-gtk3 iio-sensor-proxy im-config
  intel-media-va-driver ippusbxd iptables iso-codes iw keyboard-configuration kmod
  language-selector-common language-selector-gnome libaa1 libaacs0 libaccountsservice0 libaom0
  libappindicator3-1 libappstream4 libarchive13 libasn1-8-heimdal libasound2-plugins libaspell15
  libassuan0 libasyncns0 libavahi-core7 libavahi-glib1 libavc1394-0 libavcodec58 libavformat58
```

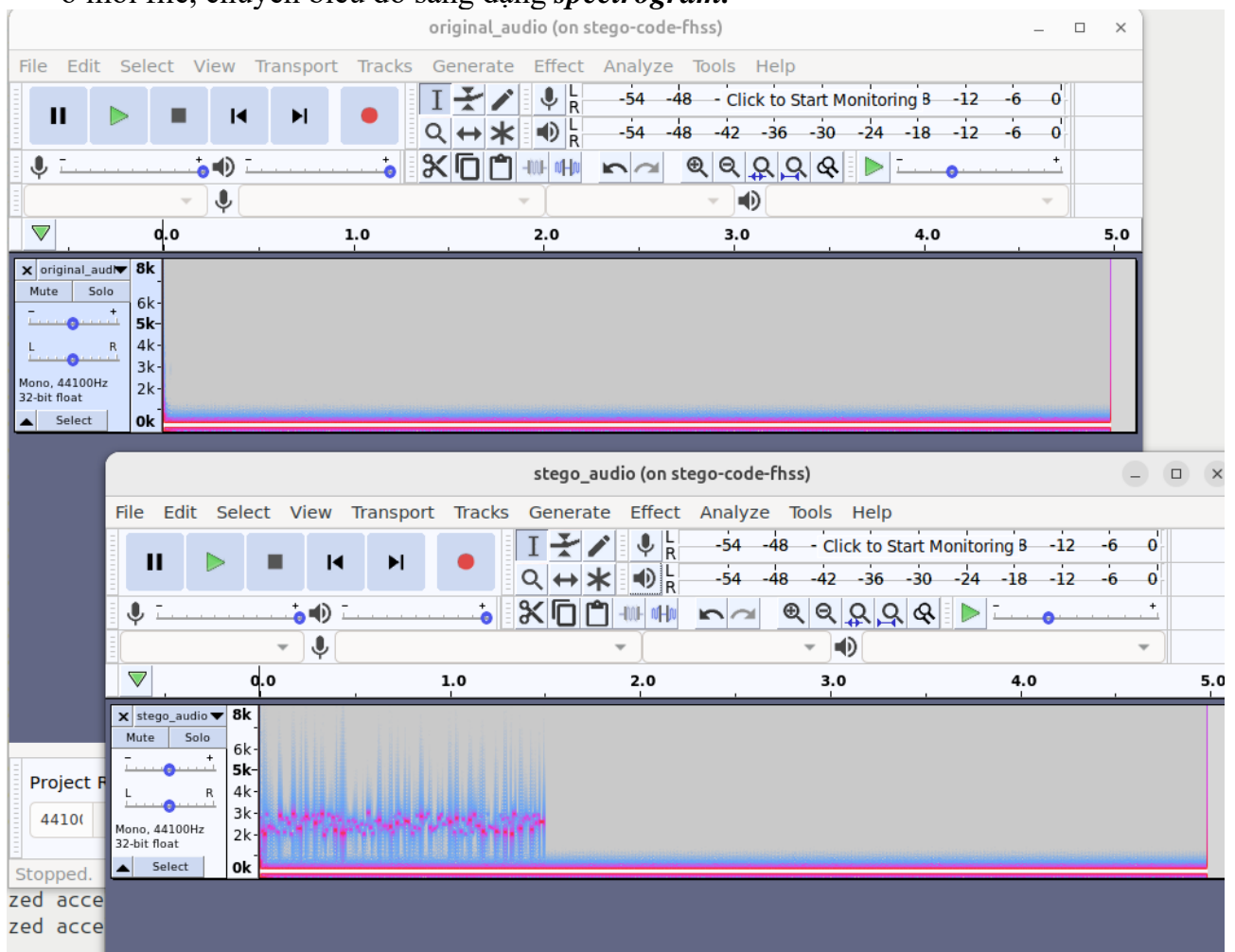
Hình 14 Cài đặt audacity

- Sinh viên sử dụng công cụ audacity để mở 2 file âm thanh đang có.



Hình 15 Theo dõi 2 file âm thanh trong audacity

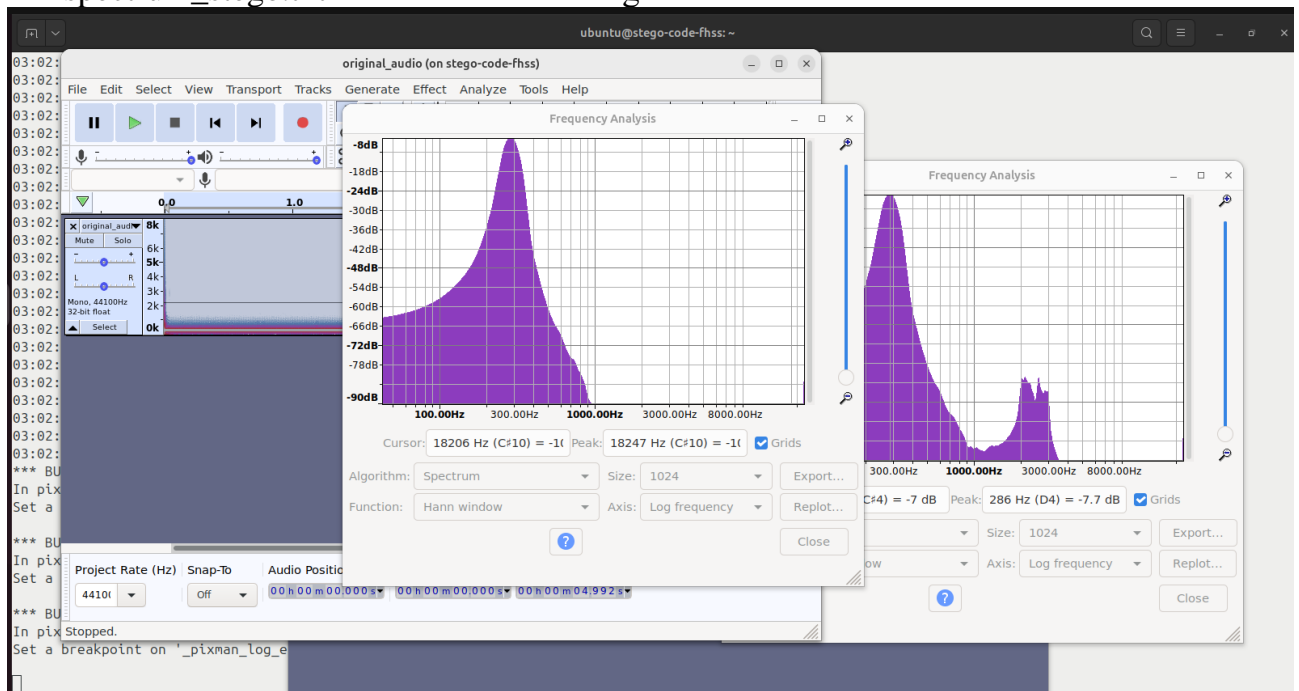
- Audacity sẽ hiển thị 2 file âm thanh dưới dạng biểu đồ **waveform**, nhấn vào dấu 3 chấm ở mỗi file, chuyển biểu đồ sang dạng **spectrogram**.



Hình 16 Theo dõi 2 file âm thanh dưới dạng đồ thị spectrogram

- Tiếp theo, để xem tin được giấu ở vùng tần số nào, sinh viên chọn **Analyze -> Plot spectrum**.

- Xuất 2 biểu đồ dưới dạng text với tên spectrum_original.txt với file âm thanh gốc và spectrum_stego.txt với file âm thanh đã giấu tin.



Hình 17 Đồ thị Plot Spectrum

```
ubuntu@stego-code-fhss:~$ ls
hide.py message.txt original_audio.wav spectrum_original.txt spectrum_stego.txt stego_audio.wav
ubuntu@stego-code-fhss:~$
```

Hình 18 Kết quả sau khi export

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

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@LabtainerVMware:~/labtainer/labtainer-student$ checkwork
Results stored in directory: /home/student/labtainer_xfer/stego-code-fhss
Successfully copied 486kB to stego-code-fhss-igrader:/home/instructor/b21dcat135.stego-code-fhss.lab
Successfully copied 2.05kB to /home/student/labtainer_xfer/stego-code-fhss
Labname stego-code-fhss
```

Student	Sizeout	message	hide	spectrum_origin	spectrum_stego
=====	=====	=====	=====	=====	=====
b21dcat135	Y	Y	Y	Y	Y

What is automatically assessed for this lab:

Hình 19 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-code-fhss

TÀI LIỆU THAM KHẢO

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