

**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: D21CQAT01-B
TÊN BÀI: KỸ THUẬT GIẤU TIN FHSS TRONG AUDIO**

Sinh viên thực hiện:

B21DCAT105 Đặng Thị Thanh Huyền

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
DANH MỤC CÁC TỪ VIẾT TẮT	4
CHƯƠNG 1: GIỚI THIỆU CHUNG VỀ BÀI THỰC HÀNH.....	5
1.1 Giới thiệu chung về bài thực hành	5
1.2 Nội dung và hướng dẫn bài thực hành	5
CHƯƠNG 2: PHÂN TÍCH YÊU CẦU BÀI THỰC HÀNH.....	8
2.1 Thiết kế bài thực hành	8
2.2 Cài đặt và cấu hình máy ảo	8
2.3 Tích hợp và triển khai	10
CHƯƠNG 3: THỬ NGHIỆM VÀ ĐÁNH GIÁ.....	13
TÀI LIỆU THAM KHẢO	17

DANH MỤC CÁC HÌNH VẼ

Hình 1: Yêu cầu checkwork.....	8
Hình 2: Giao diện Labedit.....	9
Hình 3: Cài đặt phần result.....	9
Hình 4: Nội dung dockerfiles	10
Hình 5: Đẩy image bài thực hành lên docker hub.....	11
Hình 6: Trạng thái bài thực hành trên Docker Hub.....	11
Hình 7: tạo thủ công ở thư mục ~/labs bằng lệnh	12
Hình 8: thực hiện đẩy lên github bằng git hoặc thủ công.....	12
Hình 9: tải bài thực hành bằng imodule	13
Hình 10: mở bài lab.....	13
Hình 11: Sinh viên thực hiện in ra màn hình nội dung file message.txt	14
Hình 12: Tiến hành in ra nội dung file nhị phân	14
Hình 13: Tiến hành giấu tin.....	14
Hình 14: Kiểm tra định dạng file đầu ra stego_audio.wav.....	15
Hình 15: tiến hành so sánh dung lượng của file audio trước và sau khi giấu tin.	15
Hình 16: Để kiểm tra sự khác nhau của header.....	16
Hình 17: Kết quả checkwork.....	16

DANH MỤC CÁC BẢNG BIỂU

DANH MỤC CÁC TỪ VIẾT TẮT

Từ viết tắt	Thuật ngữ tiếng Anh/Giải thích	Thuật ngữ tiếng Việt/Giải thích
FHSS	Frequency Hopping Spread Spectrum	Phổ tần nhảy tần
WAV	Waveform Audio File Format	Định dạng tệp âm thanh chuẩn
PCM	Pulse-code Modulation	Điều chế mã xung – phương pháp mã hóa âm thanh
DSP	Digital Signal Processing	Xử lý tín hiệu số
LSB	Least Significant Bit	Bit ít quan trọng nhất (kỹ thuật giấu tin cơ bản)
Stego	Steganographic	Liên quan đến giấu tin

CHƯƠNG 1: GIỚI THIỆU CHUNG VỀ BÀI THỰC HÀNH

1.1 Giới thiệu chung về bài thực hành

Bài thực hành này hướng dẫn sinh viên cách giấu thông tin (steganography) trong tín hiệu âm thanh bằng kỹ thuật nhảy tần (Frequency Hopping Spread Spectrum - FHSS). Steganography là một phương pháp che giấu dữ liệu sao cho sự tồn tại của thông tin được bảo mật, khác với mã hóa – vốn chỉ bảo vệ nội dung dữ liệu. Việc giấu thông tin trong âm thanh là một lĩnh vực thuộc steganography, trong đó dữ liệu bí mật được nhúng vào các thuộc tính của tín hiệu âm thanh như biên độ, tần số hoặc pha.

Trong bài này, kỹ thuật FHSS – một biến thể của phương pháp trải phổ – được sử dụng để phân tán dữ liệu giấu vào các dải tần khác nhau trong tín hiệu âm thanh. Về mặt lý thuyết, FHSS hoạt động bằng cách thay đổi tần số sóng mang theo một chuỗi tần số đã định sẵn (hopping pattern), nhằm tăng tính bảo mật và giảm khả năng bị phát hiện. Trong ngữ cảnh giấu tin, phương pháp này được điều chỉnh để dữ liệu nhị phân được nhúng vào các điểm cụ thể trong tín hiệu âm thanh dựa trên chuỗi nhảy tần đó.

Quá trình giấu tin diễn ra qua các bước chính sau:

1. Chuyển đổi thông điệp cần giấu từ dạng văn bản sang dạng nhị phân.
2. Ánh xạ các bit nhị phân vào các vị trí tần số xác định trong tín hiệu âm thanh theo thuật toán nhảy tần.
3. Tạo ra một tệp âm thanh mới (stego-audio) chứa thông điệp giấu bên trong nhưng vẫn đảm bảo chất lượng âm thanh không bị ảnh hưởng rõ rệt.

Thông qua bài thực hành, sinh viên không chỉ hiểu rõ về cách giấu tin trong tín hiệu âm thanh mà còn nắm được nguyên lý hoạt động của FHSS và quy trình xử lý tín hiệu số cơ bản như chuyển đổi định dạng, kiểm tra tần số và trích xuất dữ liệu.

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 cách sử dụng các đặc tính của tín hiệu âm thanh và nhảy tần.

1.2.2 Yêu cầu đối với sinh viên

Hiểu rõ khái niệm steganography và vai trò của giấu tin trong âm thanh.

Nắm được nguyên lý hoạt động của thuật toán FHSS.

Hiểu các khái niệm cơ bản về xử lý tín hiệu số (digital signal processing), chẳng hạn như lấy mẫu, tần số, và biến đổi tín hiệu.

1.2.3 Nội dung thực hành

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

imodule https://github.com/DTHuyn/steg_basic_fhss_hide/raw/master/steg_basic_fhss_hide.tar

Vào terminal, gõ:

Labtainer -r steg_basic_fhss_hide

(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 với người dùng **ubuntu**. Thực hiện lệnh kiểm tra các thư mục có sẵn trong container

Sinh viên thực hiện in ra màn hình nội dung file message.txt

Để tiến hành giấu tin, sinh viên cần phải chuyển message từ text sang dạng nhị phân. Đã có sẵn code chuyển nội dung tin cần giấu sang dạng nhị phân text_to_binary.py, tiến hành đọc file

ls

cat message.txt

cat text_to_binary.py

Tiến hành chuyển text sang nhị phân thông qua câu lệnh :

python3 text_to_binary.py

Màn hình hiện thị đã chuyển nội dung thành công và in mã nhị phân ra màn hình, đồng thời lưu vào file *binary_message.txt*. Tiến hành in ra nội dung file nhị phân này:

cat binary_message.txt

Sinh viên thực hiện đọc chương trình python giấu tin *hide_message.py*

Kiểm tra xem đã có đủ các file để tiến hành giấu tin chưa bằng câu lệnh:

ls -l

Đảm bảo đầu vào đã có file *original_audio.wav* là file âm thanh đầu vào và file *binary_message.txt* là nội dung tin giấu sau khi chuyển sang mã nhị phân. Và có file *hide_message.py* là file code giấu tin.

Tiến hành giấu tin bằng câu lệnh :

python3 hide_message.py

Thực hiện ls để xem file stego_audio.wav đã được sinh ra hay chưa.

Kiểm tra định dạng file đầu ra stego_audio.wav xem thỏa mãn yêu cầu không?

file stego_audio.wav

Đảm bảo file có định dạng: IFF (little-endian) data, WAVE audio, Microsoft PCM, 16 bit, mono 44100 Hz

Sau khi hoàn thành giấu tin, ta tiến hành so sánh dung lượng của file audio trước và sau khi giấu tin.

ls -l

Với phương pháp này thì nó sẽ không làm thay đổi dung lượng của file audio trước và sau khi giấu tin.

Để kiểm tra sự khác nhau của header, ta dùng lệnh:

xxd original_audio.wav /head -n 10

xxd stego_audio.wav /head -n 10

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 steg_basic_fhss_hide

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 mạng được để random do không cần đến địa chỉ mạng trong bài thực hành.

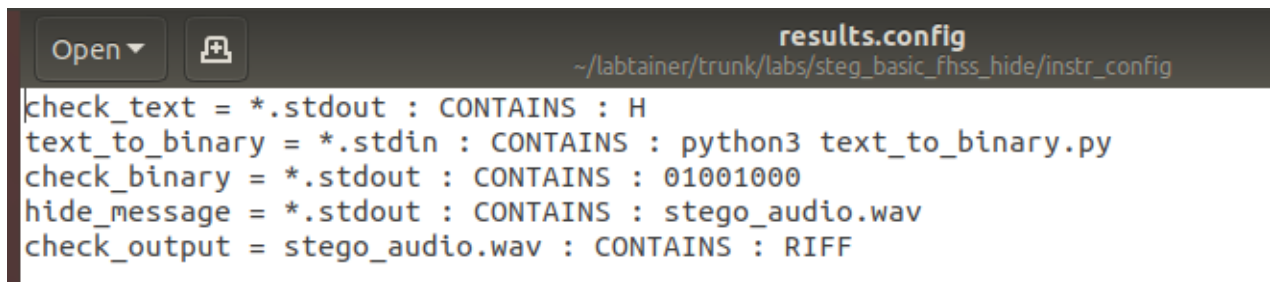
Cấu hình Docker

- Bài lab chạy với image là .base2
- Cần cài thêm môi trường ảo hóa venv và thư viện pycryptodome trong dockerfiles
- Docs lưu lại hướng dẫn thực hành cho sinh viên

Các nhiệm vụ cần thực hiện để thành công:

- Chuyển đổi ký tự dạng text sang nhị phân
- Kiểm tra chuỗi nhị phân đã trùng khớp với yêu cầu
- Tiến hành giấu tin thành công
- 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:



```
results.config
~/labtainer/trunk/labs/steg_basic_fhss_hide/instr_config

check_text = *.stdout : CONTAINS : H
text_to_binary = *.stdin : CONTAINS : python3 text_to_binary.py
check_binary = *.stdout : CONTAINS : 01001000
hide_message = *.stdout : CONTAINS : stego_audio.wav
check_output = stego_audio.wav : CONTAINS : RIFF
```

Hình 1: Yêu cầu checkwork

text_to_binary : Chuyển đổi ký tự dạng text sang nhị phân

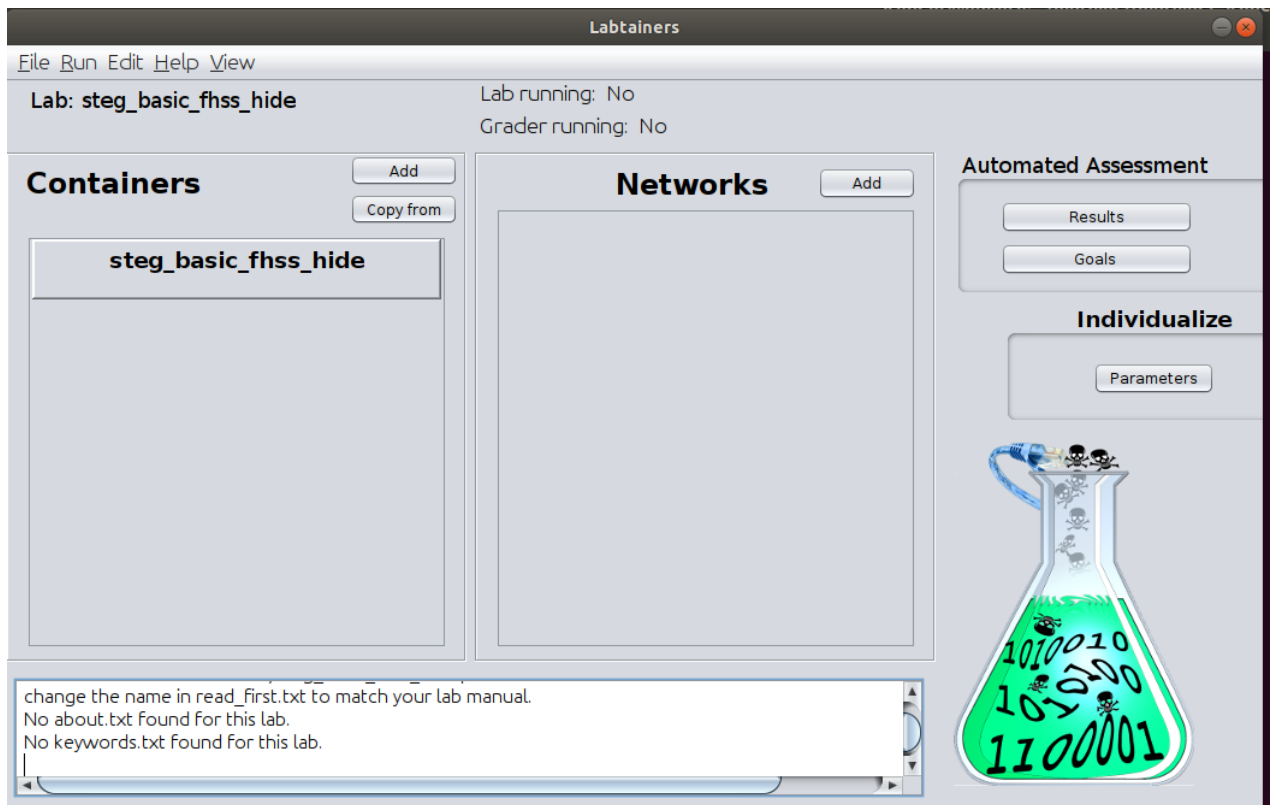
check_output: Tiến hành kiểm tra định dạng file được tạo

check_binary : Kiểm tra chuỗi nhị phân đã trùng khớp với yêu cầu

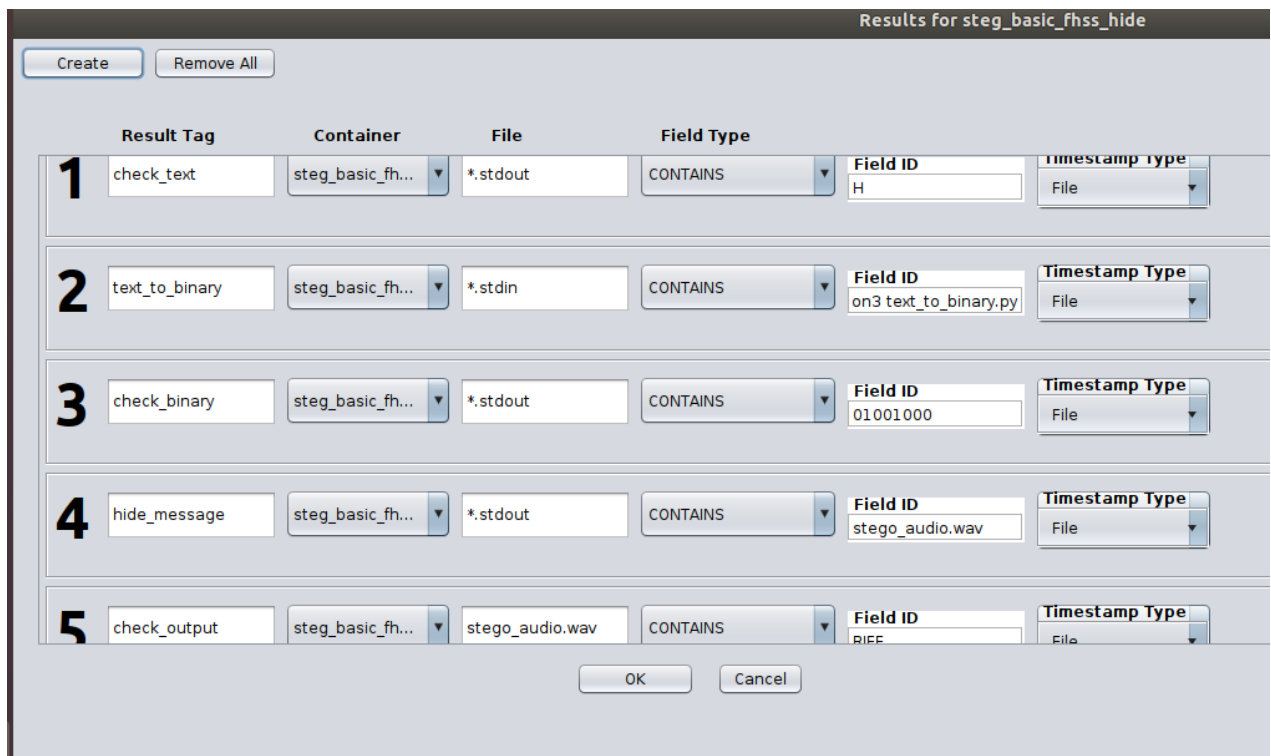
check_text : Kiểm tra thực hiện in ra thông điệp cần giấu

hide_mesage: Kiểm tra đã thực hiện giấu tin và in thành công file âm thanh sau khi giấu.

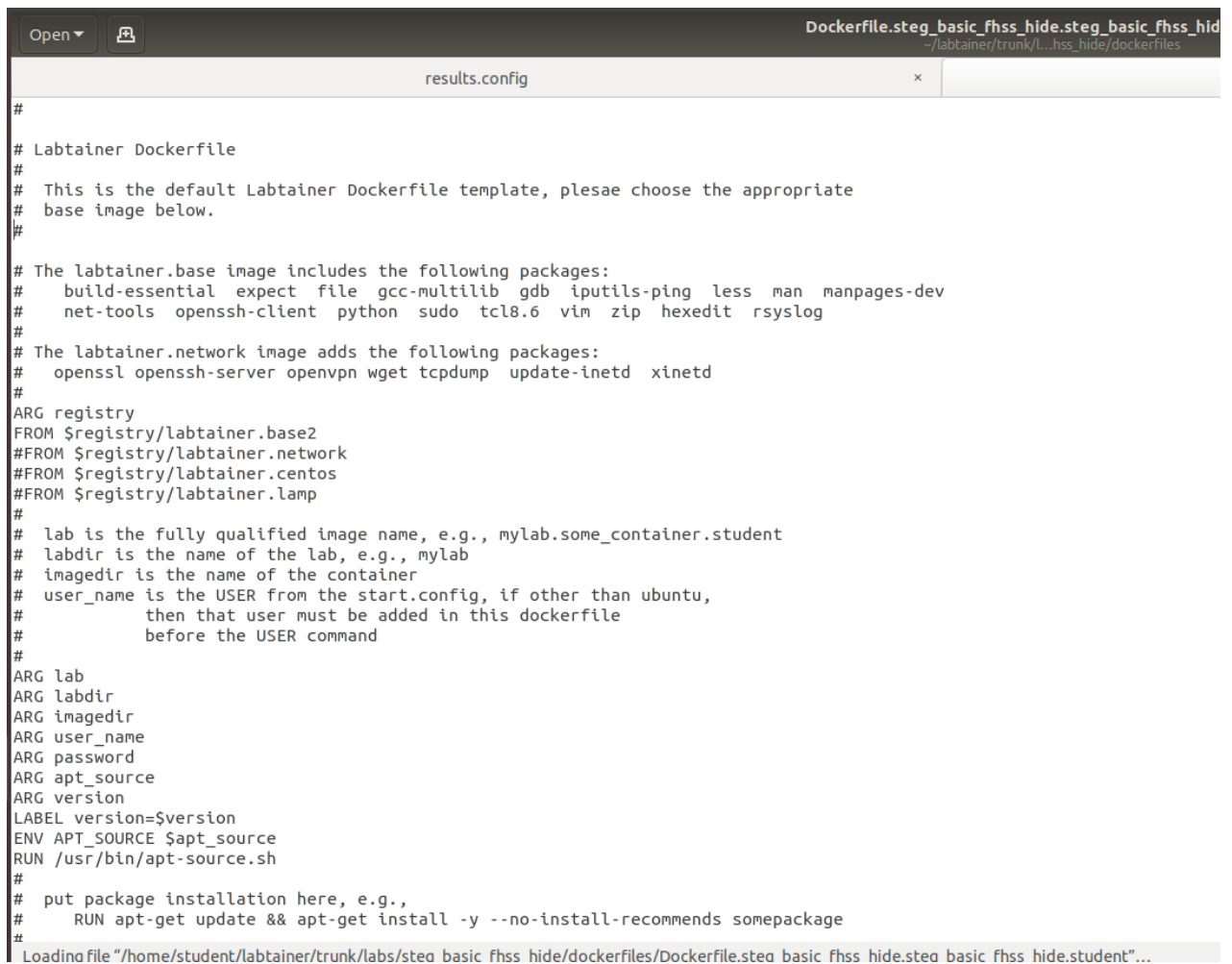
2.2 Cài đặt và cấu hình máy ảo



Hình 2: Giao diện Labedit



Hình 3: Cài đặt phần result

The image shows a code editor window with a dark theme. The title bar at the top reads 'Dockerfile.steg_basic_fhss_hide.steg_basic_fhss_hid' and the file path is '/labtainer/trunk/l...hss_hide/dockerfiles'. The editor displays the content of a file named 'results.config'. The code is a Dockerfile template with various comments and instructions. It starts with a shebang line '#', followed by a comment '# Labtainer Dockerfile'. It then provides instructions on how to use the template, including choosing a base image and installing packages. The code includes several ARG statements for registry, lab, labdir, imagedir, user_name, password, apt_source, and version. It also includes a LABEL statement for version and an ENV statement for APT_SOURCE. The code ends with a RUN statement for apt-get update and install. The status bar at the bottom shows the file path: 'Loading file "/home/student/labtainer/trunk/labs/steg_basic_fhss_hide/dockerfiles/Dockerfile.steg_basic_fhss_hide.steg_basic_fhss_hide.student"...'.

Hình 4: Nội dung dockerfiles

2.3 Tích hợp và triển khai

Bài thực hành được triển khai như sau:

Docker

Đường dẫn: <https://hub.docker.com/repositories/dthuynd>

Thêm registry cho bài thực hành

Truy cập vào thư mục trunk/distrib gõ lệnh: *docker login* đăng nhập tài khoản DockerHub với *Username: dthuynd* và *password: *****

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

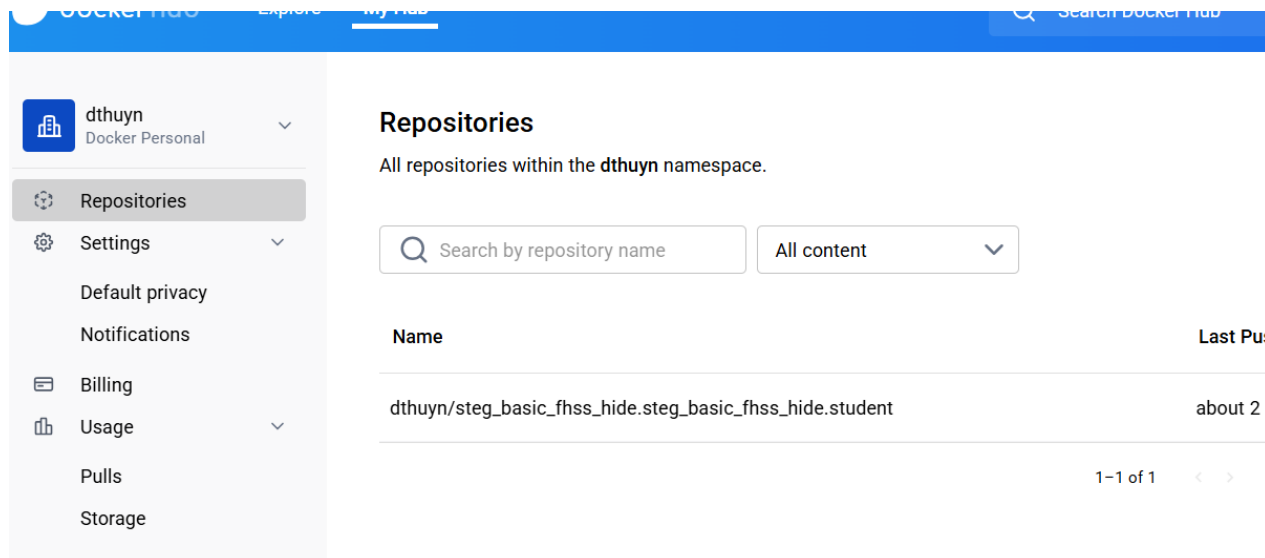
```

student@ubuntu:~/labtainer/trunk/distrib$ docker login
Login with your Docker ID to push and pull images from Docker Hub. If you don't have a
Username: dthuyn
Password:
WARNING! Your password will be stored unencrypted in /home/student/.docker/config.json
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
student@ubuntu:~/labtainer/trunk/distrib$ ./publish.py -d -l steg_basic_fhss_hide
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]

```

Hình 5: Đẩy image bài thực hành lên docker hub



Hình 6: Trạng thái bài thực hành trên Docker Hub

Github

Đường dẫn: https://github.com/DTHuyn/steg_basic_fhss_hide

Ở đường dẫn \$LABTAINER_DIR/distrib, tạo file tar bằng create-imodules.sh hoặc tạo thủ công ở thư mục ~/labs bằng lệnh:

```
tar -cvf steg_basic_fhss_hide.tar steg_basic_fhss_hide
```

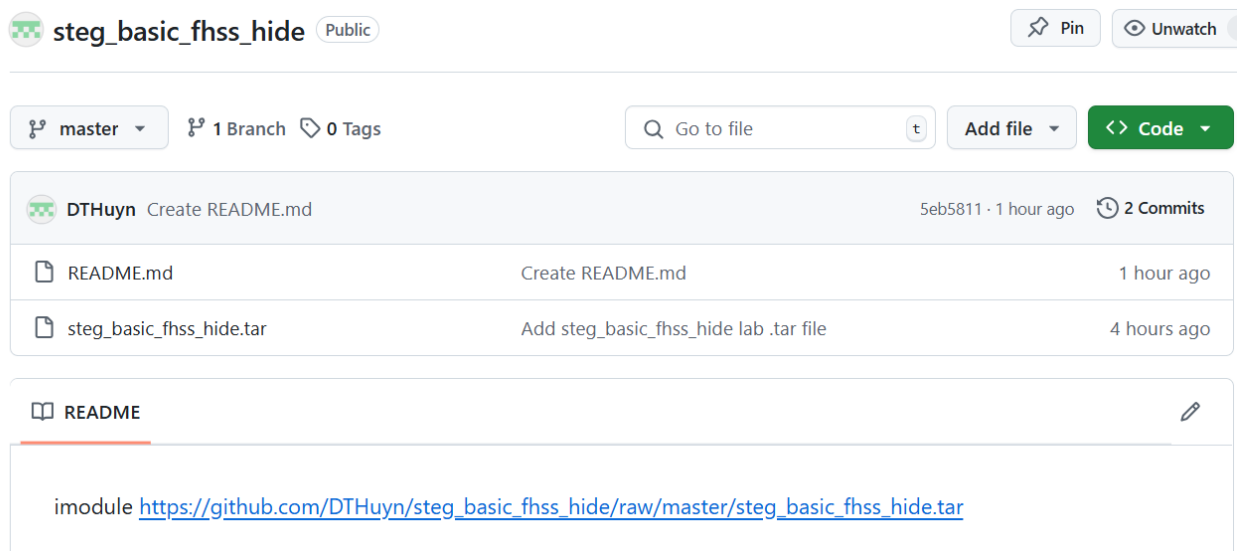
```

student@ubuntu:~/labtainer/trunk/labs$ tar -cvf steg_basic_fhss_hide.tar steg_basic_fhss_hide
steg_basic_fhss_hide/
steg_basic_fhss_hide/config/
steg_basic_fhss_hide/config/start.config
steg_basic_fhss_hide/config/parameter.config
steg_basic_fhss_hide/config/steg_basic_fhss_hide-home_tar.list
steg_basic_fhss_hide/dockerfiles/
steg_basic_fhss_hide/dockerfiles/Dockerfile.steg_basic_fhss_hide.steg_basic_fhss_hide.student
steg_basic_fhss_hide/instr_config/
steg_basic_fhss_hide/instr_config/pregrade.sh
steg_basic_fhss_hide/instr_config/goals.config
steg_basic_fhss_hide/instr_config/results.config
steg_basic_fhss_hide/docs/
steg_basic_fhss_hide/docs/read_first.txt
steg_basic_fhss_hide/steg_basic_fhss_hide/
steg_basic_fhss_hide/steg_basic_fhss_hide/_bin/
steg_basic_fhss_hide/steg_basic_fhss_hide/_bin/fixlocal.sh
steg_basic_fhss_hide/steg_basic_fhss_hide/home_tar/
steg_basic_fhss_hide/steg_basic_fhss_hide/home_tar/home.tar
steg_basic_fhss_hide/steg_basic_fhss_hide/message.txt

```

Hình 7: tạo thủ công ở thư mục ~/labs bằng lệnh

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



Hình 8: thực hiện đẩy lên github bằng git hoặc thủ công

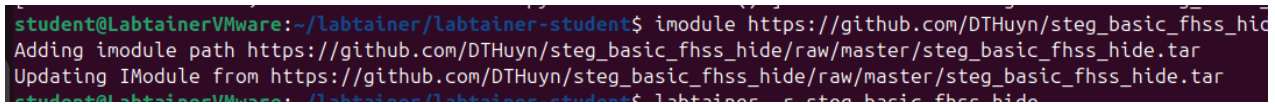
File `steg_basic_fhss_hide.tar` chứa bài thực hành

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:

imodule https://github.com/DTHuyn/steg_basic_fhss_hide/raw/master/steg_basic_fhss_hide.tar



```
student@LabtainerVMware:~/labtainer/labtainer-student$ imodule https://github.com/DTHuyn/steg_basic_fhss_hide/raw/master/steg_basic_fhss_hide.tar
Adding imodule path https://github.com/DTHuyn/steg_basic_fhss_hide/raw/master/steg_basic_fhss_hide.tar
Updating IModule from https://github.com/DTHuyn/steg_basic_fhss_hide/raw/master/steg_basic_fhss_hide.tar
student@LabtainerVMware:~/labtainer/labtainer-student$ labtainer -r steg_basic_fhss_hide
```

Hình 9: tải bài thực hành bằng imodule

Vào terminal, gõ:

labtainer -r steg_basic_fhss_hide



```
student@LabtainerVMware:~/labtainer/labtainer-student$ labtainer -r steg_basic_fhss_hide
latest: Pulling from dthuyn/steg_basic_fhss_hide.steg_basic_fhss_hide.student
e5f011653f0d: Downloading [=====>] 704e5f011653f0d: Downloading [=====>]
f0d: Extracting [=====>] 927Be5f011653f0d: Extracting [=====>]
ing
e3d76562ae23: Waiting
8e7e24552aa8: Downloading [=====>] 719e5f011653f0d: Pull complete
c4ea71364da7: Extracting [=====>] 111Bc4ea71364da7: Extracting [=====>]
da7: Pull complete
8e7e24552aa8: Extracting [=====>] 4.654kB8e7e24552aa8: Extracting [=====>]
aa8: Pull complete
05362e1ec3f9: Downloading [=====>] 72305362e1ec3f9: Downloading [=====>]
3f9: Extracting [=====>] 4.638kB05362e1ec3f9: Extracting [=====>]
complete
e5cc82cdea62: Extracting [=====>] 446Be5cc82cdea62: Extracting [=====>]
a62: Pull complete
```

Hình 10: mở bài lab

Thực hiện bài lab

(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 với người dùng **ubuntu**. Thực hiện lệnh kiểm tra các thư mục có sẵn trong container

Sinh viên thực hiện in ra màn hình nội dung file message.txt

Để tiến hành giấu tin, sinh viên cần phải chuyển message từ text sang dạng nhị phân. Đã có sẵn code chuyển nội dung tin cần giấu sang dạng nhị phân text_to_binary.py, tiến hành đọc file

ls

cat message.txt

cat text_to_binary.py

```
ubuntu@stegbasicfhsshide:~$ ls
hide_message.py  message.txt  original_audio.wav  text_to_binary.py
ubuntu@stegbasicfhsshide:~$ cat message.txt
H
```

Hình 11: Sinh viên thực hiện in ra màn hình nội dung file message.txt

Tiến hành chuyển text sang nhị phân thông qua câu lệnh :

```
python3 text_to_binary.py
```

Màn hình hiển thị đã chuyển nội dung thành công và in mã nhị phân ra màn hình, đồng thời lưu vào file *binary_message.txt*. Tiến hành in ra nội dung file nhị phân này:

```
cat binary_message.txt
```

```
ubuntu@stegbasicfhsshide:~$ python3 text_to_binary.py
Tin cần giấu là: H chuyển sang mã nhị phân là: 01001000
Đã tạo file binary_message.txt
ubuntu@stegbasicfhsshide:~$ ls
binary_message.txt  message.txt  text_to_binary.py
hide_message.py    original_audio.wav
ubuntu@stegbasicfhsshide:~$ cat binary_message.txt
01001000ubuntu@stegbasicfhsshide:~$
```

Hình 12: Tiến hành in ra nội dung file nhị phân

Sinh viên thực hiện đọc chương trình python giấu tin *hide_message.py*

Kiểm tra xem đã có đủ các file để tiến hành giấu tin chưa bằng câu lệnh:

```
ls -l
```

Đảm bảo đầu vào đã có file *original_audio.wav* là file âm thanh đầu vào và file *binary_message.txt* là nội dung tin giấu sau khi chuyển sang mã nhị phân. Và có file *hide_message.py* là file code giấu tin.

Tiến hành giấu tin bằng câu lệnh :

```
python3 hide_message.py
```

```
ubuntu@stegbasicfhsshide:~$ python3 hide_message.py
Đã tạo file stego_audio.wav
```

Hình 13: Tiến hành giấu tin

Thực hiện ls để xem file *stego_audio.wav* đã được sinh ra hay chưa.

Kiểm tra định dạng file đầu ra *stego_audio.wav* xem thỏa mãn yêu cầu không?

file *stego_audio.wav*

```
ubuntu@stegbasicfhsshide:~$ file stego_audio.wav
stego_audio.wav: RIFF (little-endian) data, WAVE audio, Microsoft PCM, 16 bit, mono 44100 Hz
```

Hình 14: Kiểm tra định dạng file đầu ra *stego_audio.wav*

Đảm bảo file có định dạng: RIFF (little-endian) data, WAVE audio, Microsoft PCM, 16 bit, mono 44100 Hz

Sau khi hoàn thành giấu tin, ta tiến hành so sánh dung lượng của file audio trước và sau khi giấu tin.

ls -l

```
ubuntu@stegbasicfhsshide:~$ ls -l
total 880
-rw-rw-r-- 1 ubuntu ubuntu    8 Apr 24 11:52 binary_message.txt
-r-x--x--x 1 ubuntu ubuntu 2092 Apr 24 07:53 hide_message.py
-rw-r--r-- 1 ubuntu ubuntu    2 Apr 24 07:52 message.txt
-r----- 1 ubuntu ubuntu 441044 Apr 24 07:53 original_audio.wav
-rw-rw-r-- 1 ubuntu ubuntu 441044 Apr 24 11:52 stego_audio.wav
-r-x--x--x 1 ubuntu ubuntu   642 Apr 24 07:53 text_to_binary.py
```

Hình 15: tiến hành so sánh dung lượng của file audio trước và sau khi giấu tin.

Với phương pháp này thì nó sẽ không làm thay đổi dung lượng của file audio trước và sau khi giấu tin.

Để kiểm tra sự khác nhau của header, ta dùng lệnh:

xxd original_audio.wav |head -n 10

xxd stego_audio.wav |head -n 10


```

-bash: xxs: command not found
ubuntu@stegbasicfhsshide:~$ xxd original_audio.wav | head -n 10
00000000: 5249 4646 ccba 0600 5741 5645 666d 7420 RIFF....WAVEfmt
00000010: 1000 0000 0100 0100 44ac 0000 8858 0100 .....D....X..
00000020: 0200 1000 6461 7461 a8ba 0600 0000 0204 ....data.....
00000030: 0008 f70b e10f bb13 8217 301b c41e 3822 .....0...8"
00000040: 8b25 b728 ba2b 922e 3a31 b133 f435 0138 .%.(.+:.1.3.5.8
00000050: d639 703b cf3c f03d d33e 773f db3f ff3f .9p;.<.=.>w?..?
00000060: e23f 863f e93e 0d3e f33c 9b3b 073a 3938 .??.>.>.<.;.:98
00000070: 3336 f633 8531 e22e 0f2c 1129 e925 9b22 36.3.1....).%."
00000080: 2a1f 9a1b ee17 2a14 5210 690c 7408 7604 *.....*.R.i.t.v.
00000090: 7400 73fc 74f8 7cf4 90f0 b4ec ebe8 39e5 t.s.t.|.....9.
ubuntu@stegbasicfhsshide:~$ xxd stego_audio.wav | head -n 10
00000000: 5249 4646 ccba 0600 5741 5645 666d 7420 RIFF....WAVEfmt
00000010: 1000 0000 0100 0100 44ac 0000 8858 0100 .....D....X..
00000020: 0200 1000 6461 7461 a8ba 0600 1300 2404 ....data.....$.
00000030: 2f08 2f0c 1f10 fb13 bf17 661b f01e 5722 /. /.....f...W"
00000040: 9b25 b628 a92b 722e 0d31 7933 b635 c037 .%.(.+r..1y3.5.7
00000050: 9739 373b a03c cd3d bf3e 743f e83f 1c40 .97;.<.=.>t?..@
00000060: 0c40 bb3f 253f 4c3e 313d d43b 373a 5d38 .@.??%?L>1=.;7:]8
00000070: 4936 fc33 7a31 c72e e62b dd28 ad25 5b22 I6.3z1...+.(.%["
00000080: ea1e 5e1b bb17 0214 3810 5f0c 7b08 8d04 ..^.....8._.{...
00000090: 9900 a5fc aff8 bcf4 d0f0 f1ec 20e9 63e5 ..... .C.
ubuntu@stegbasicfhsshide:~$ █

```

Hình 16: Để kiểm tra sự khác nhau của header

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@LabtainerVMware:~/labtainer/labtainer-student$ checkwork steg_basic_fhss_hide
Results stored in directory: /home/student/labtainer_xfer/steg_basic_fhss_hide
Successfully copied 59.9kB to steg_basic_fhss_hide-igrader:/home/instructor/B21DCAT105.steg_basic_fhss_hide.lab
Successfully copied 2.05kB to /home/student/labtainer_xfer/steg_basic_fhss_hide
Labname steg_basic_fhss_hide

Student      |      check_text | text_to_binary |      check_binary |      hide_message |      check_output |
===== | ===== | ===== | ===== | ===== | ===== |
B21DCAT105   |      Y |      Y |      Y |      Y |      Y |
What is automatically assessed for this lab:

```

Hình 17: Kết quả checkwork

Khởi động lại bài lab:

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

labtainer -r steg_basic_fhss_hide

TÀI LIỆU THAM KHẢO

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