

**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: TÁCH TIN ẨN TRONG ÂM THANH
BẰNG KỸ THUẬT FHSS**

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: Khởi động bài lab.....	13
Hình 10:Tiến hành kiểm tra ip của 2 máy	13
Hình 11: chỉnh sửa nội dung file message.txt	14
Hình 12: mở file cấu hình ssh và chỉnh sửa	14
Hình 13: Tiến hành gửi file audio chứa thông điệp cho người nhận	14
Hình 14: Bên nhận tạo chuỗi nhảy tần đồng bộ với bên gửi.....	15
Hình 15: : Tiến hành tách trích xuất thông tin đã giấu.....	15
Hình 16: Màn hình checkwork.....	15

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
SSH	Secure Shell	Giao thức kết nối mạng an toàn
IP	Internet Protocol	Giao thức liên mạng
LAN	Local Area Network	Mạng cục bộ
FHSS	Frequency Hopping Spread Spectrum	Phổ trải nhảy tần
SCP	Secure Copy Protocol	Giao thức sao chép bảo mật
GUI	Graphical User Interface	Giao diện đồ họa
venv	Virtual Environment	Môi trường ảo
PyCryptodome	Python Cryptographic Library	Thư viện mã hóa trong Python
GitHub	Git Repository Hosting Service	Dịch vụ lưu trữ mã nguồn sử dụng Git
Docker	Containerization Platform	Nền tảng đóng gói và triển khai ứng dụng bằng container

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). Đây là một phương pháp mã hóa và truyền tin có tính bảo mật cao, thường được sử dụng trong các hệ thống truyền thông quân sự hoặc không dây để tránh bị phát hiện và gây nhiễu.

FHSS (Frequency Hopping Spread Spectrum) là kỹ thuật truyền tín hiệu bằng cách thay đổi tần số mang theo một chuỗi định sẵn gọi là *hop pattern* (mẫu nhảy tần). Trong kỹ thuật giấu tin, thuật toán FHSS được ứng dụng để phân tán dữ liệu bí mật vào các vị trí khác nhau trong tệp âm thanh, theo một dãy nhảy tần giả lập (pseudo-random), giúp tăng tính bảo mật và làm cho dữ liệu khó bị phát hiện.

Nguyên lý hoạt động của giấu và tách tin bằng FHSS

- Bước 1 – Giấu tin:

Trước đó, một thông điệp (thường là dạng văn bản) được mã hóa nhị phân. Sau đó, mỗi bit sẽ được nhúng vào một vị trí cụ thể trong tệp âm thanh dựa trên một chuỗi *hop sequence*. Chuỗi này được tạo từ một khóa bí mật (secret key) và đóng vai trò quyết định vị trí của các bit trong toàn bộ tệp âm thanh.

- Bước 2 – Tách tin:

Để tách được thông điệp, cần biết chính xác chuỗi *hop pattern* đã được sử dụng khi giấu tin. Khi đã có khóa bí mật, chương trình sẽ tạo lại đúng chuỗi nhảy tần để định vị lại chính xác các vị trí chứa dữ liệu trong tệp âm thanh. Sau đó, các bit được trích xuất tuần tự và ghép lại thành thông điệp gốc.

Đặc điểm của kỹ thuật tách tin bằng FHSS

- Tính phân tán cao: dữ liệu được rải đều trên toàn bộ âm thanh nên khó bị phát hiện.
- Chống nhiễu tốt: do việc nhảy tần giúp tránh các đoạn âm bị biến đổi nhẹ.
- Cần có khóa để trích xuất chính xác: người không có khóa sẽ không thể tái tạo được chuỗi nhảy tần, do đó không thể tách được dữ liệu.

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 cách sử dụng các đặc tính của tín hiệu âm thanh và kỹ thuật nhảy tần. Đồng thời làm quen với môi trường thực hành an toàn thông tin qua các lệnh cấu hình và phân tích.

1.2.2 Yêu cầu đối với sinh viên

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

Nắm được nguyên lý hoạt động của thuật toán Frequency Hopping Spread Spectrum (FHSS).

Hiểu cách sử dụng giao thức SSH, SCP để thực hiện kết nối và truyền dữ liệu giữa các máy trong mạng LAN.

1.2.3 Nội dung thực hành

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

imodule <https://github.com/DTHuyn/steg-fhss-extract/raw/master/steg-fhss-extract.tar>

Vào terminal, gõ:

Labtainer -r steg-fhss-extract

(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 2 terminal, một cái là đại diện cho máy gửi: **sender**, một cái là đại diện cho máy nhận: **receiver**. Biết rằng 2 máy nằm cùng mạng LAN 172.20.0.0/24.

Task1: Tiến hành kiểm tra ip của 2 máy

Thực hiện lệnh kiểm tra ip và các thư mục có sẵn trong container sender và receiver.

ip a

ls -l

Sinh viên chỉnh sửa nội dung file message.txt từ sender. “HELLO PTIT!”

Trên container receiver, mở file cấu hình ssh và thay đổi các dòng config hoặc thêm cuối file /etc/ssh/sshd_config trên receiver.

Port 22

PasswordAuthentication yes

PermitRootLogin yes

Cần restart lại dịch vụ ssh-server: *sudo systemctl restart ssh*

Task2: Tiến hành gửi file audio chứa thông điệp cho người nhận.

Tại terminal sender, tiến hành gửi file audio chứa thông điệp đến người nhận.

scp stego_audio.wav ubuntu@<ip_máy_receiver>:/home/ubuntu/

Nhập password là *ubuntu*

Task3: Bên nhận tạo chuỗi nhảy tần đồng bộ với bên gửi

Trên terminal receiver, tạo chuỗi nhảy tần thông qua khóa bí mật *mykey* và độ dài chuỗi nhảy tần *88 bit* mà 2 bên trao đổi.

Tiến hành tạo chuỗi:

python3 generate_hopping_pattern.py

Nhập số bit là 88 và khóa bí mật là *mykey*

Kết quả chuỗi nhảy tần được sinh ra và lưu vào file *hopping_pattern.txt*

Đọc file : *cat hopping_pattern.txt*

Task4: Tiến hành tách trích xuất thông tin đã giấu

Kiểm tra các file có hiện tại, đảm bảo đã có file *hopping_pattern.txt* và file âm thanh được nhận từ sender *stego_audio.wav*

Tiến hành bóc tách tin:

python3 extract_message.py

Thông điệp nhận được được in ra màn hình, đồng thời lưu vào file *message.txt*

Kiểm tra xem thông điệp nhận có giống bên gửi không?

So sánh thông điệp được tách ra với nội dung file gửi xem có trùng khớp không

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-fhss-extract

CHƯƠNG 2: PHÂN TÍCH YÊU CẦU BÀI THỰC HÀNH

2.1 Thiết kế bài thực hành

Bài lab gồm 2 container là sender và receiver. Cấu hình mạng 2 máy nằm cùng mạng LAN 172.20.0.0/24. Setup ip cho sender là 172.20.0.12, máy receiver là 172.20.0.11

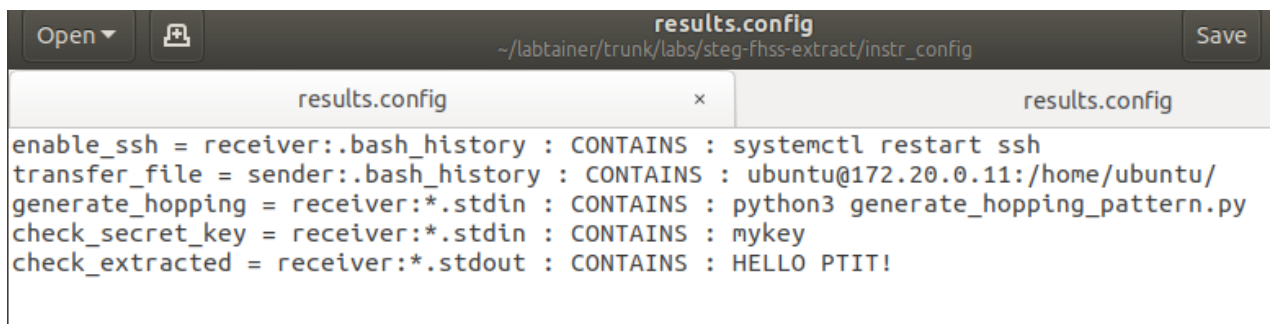
Cấu hình Docker

- Bài lab chạy với image là .network2
- 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:

- Khởi động dịch vụ SSH
- Gửi audio đã giấu tin cho bên nhận
- Tiến hành tách tin 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: Yêu cầu checkwork

enable_ssh : Tiến hành kiểm tra đã khởi động dịch vụ SSH chưa

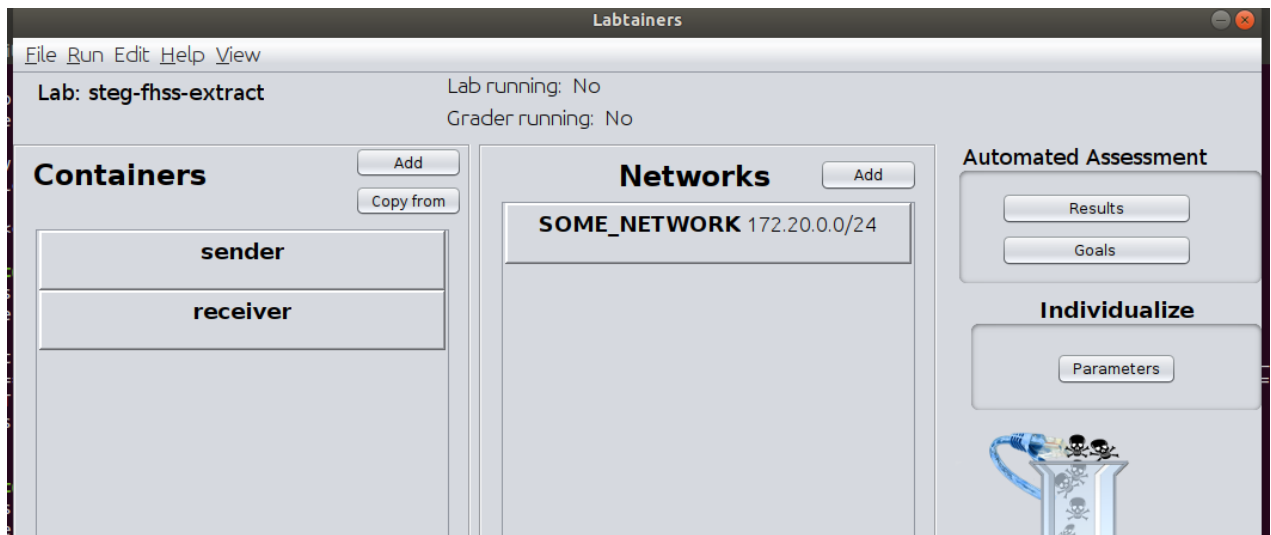
transfer_file : Kiểm tra đã thực hiện đúng câu lệnh gửi file audio cho bên nhận chưa

generate_hopping: Kiểm tra đã thực hiện câu lệnh sinh chuỗi nhảy tần

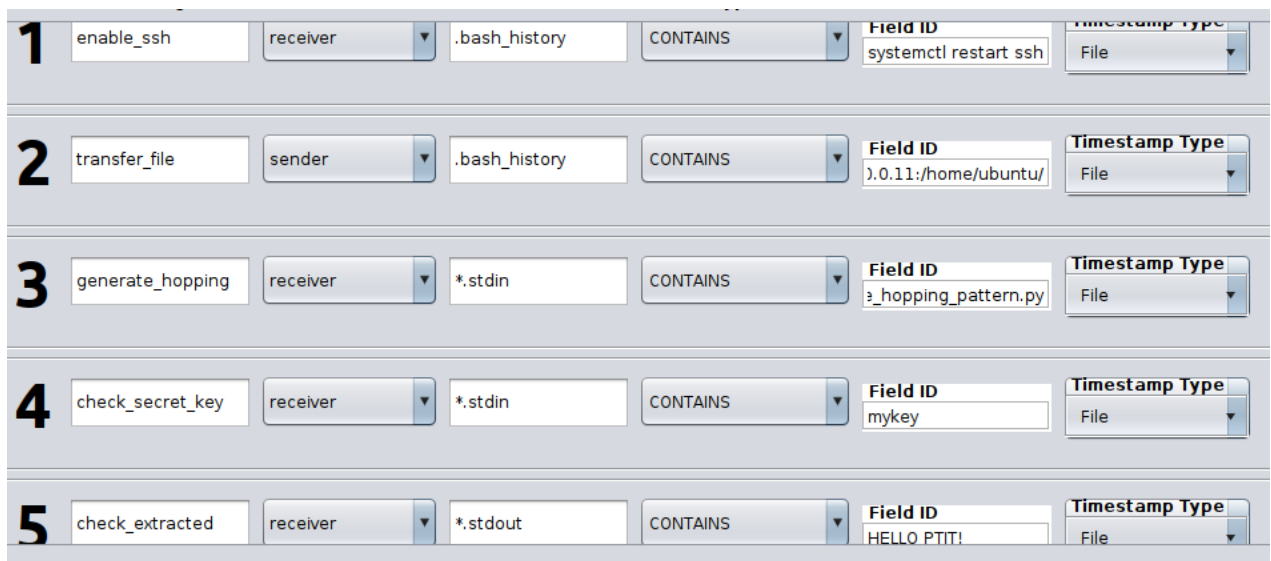
check_secret_key: Kiểm tra khóa bí mật người nhận nhập vào

check_extracted: Kiểm tra thông điệp nhận có chính xác không


2.2 Cài đặt và cấu hình máy ảo



Hình 2: Giao diện Labedit



Hình 3: Cài đặt phần result

```
Open  Dockerfile.steg-basic-fhss-extract.receiver.student
~/labtainer/trunk/labs/s...fhss-extract/dockerfiles

#
# Labtainer Dockerfile
#
# This is the default Labtainer Dockerfile template, please choose the appropriate
# base image below.
#
# The labtainer.base image includes the following packages:
#   build-essential expect file gcc-multilib gdb iputils-ping less man manpages-dev
#   net-tools openssh-client python sudo tcl8.6 vim zip hexedit rsyslog
#
# The labtainer.network image adds the following packages:
#   openssl openssh-server openvpn wget tcpdump update-inetd xinetd
#
ARG registry
FROM $registry/labtainer.network2
#FROM $registry/labtainer.network
#FROM $registry/labtainer.centos
#FROM $registry/labtainer.lamp
#
# lab is the fully qualified image name, e.g., mylab.some_container.student
# labdir is the name of the lab, e.g., mylab
# imagedir is the name of the container
# user_name is the USER from the start.config, if other than ubuntu,
#           then that user must be added in this dockerfile
#           before the USER command
#
ARG lab
ARG labdir
ARG imagedir
ARG user_name
ARG password
ARG apt_source
ARG version
LABEL version=$version
ENV APT_SOURCE $apt_source
RUN /usr/bin/apt-source.sh
#
# put package installation here, e.g.,
#   RUN apt-get update && apt-get install -y --no-install-recommends somepackage
#
#
#
```

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/dthuytn>

Thêm registry cho bài thực hành

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

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

```

Login Succeeded
student@ubuntu:~/labtainer/trunk/distrib$ ./publish.py -d -l steg-fhss-extract
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]
7725eb440c58
ae5410fc88a9

```

Hình 5: Đẩy image bài thực hành lên docker hub

Search by repository name

All content

Create a repository

Name	Last Pushed <div></div>	Contains	Visibility
dthuyn/steg-fhss-extract.receiver....	34 minutes ago	IMAGE	Public
dthuyn/steg-fhss-extract.sender.st...	36 minutes ago	IMAGE	Public

Hình 6: Trạng thái bài thực hành trên Docker Hub

Github

Đường dẫn: <https://github.com/DTHuyn/steg-fhss-extract>

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-fhss-extract.tar steg-fhss-extract
```




```

student@ubuntu:~/labtainer/trunk/labs$ tar -cvf steg-fhss-extract.tar steg-fhss-extract
steg-fhss-extract/
steg-fhss-extract/sender/
steg-fhss-extract/sender/_bin/
steg-fhss-extract/sender/_bin/fixlocal.sh
steg-fhss-extract/sender/stego_audio.wav
steg-fhss-extract/sender/home_tar/
steg-fhss-extract/sender/home_tar/home.tar
steg-fhss-extract/sender/message.txt
steg-fhss-extract/sender/sys_tar/
steg-fhss-extract/sender/sys_tar/sys.tar
steg-fhss-extract/sender/steg-fhss-extract.sender.student.tar.gz
steg-fhss-extract/sender/sys_steg-fhss-extract.sender.student.tar.gz
steg-fhss-extract/sender/_system/
steg-fhss-extract/sender/_system/etc/
steg-fhss-extract/sender/_system/etc/securetty
steg-fhss-extract/sender/_system/etc/login.defs
steg-fhss-extract/config/
steg-fhss-extract/config/start.config
steg-fhss-extract/config/parameter.config
steg-fhss-extract/config/receiver-home_tar.list
steg-fhss-extract/config/sender-home_tar.list
steg-fhss-extract/receiver/
steg-fhss-extract/receiver/sys_steg-fhss-extract.receiver.student.tar.gz
steg-fhss-extract/receiver/_bin/
steg-fhss-extract/receiver/_bin/fixlocal.sh

```

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

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

 DTHuyn Create README.md	
Name	Last commit message
 README.md	Create README.md
 steg-fhss-extract.tar	Add steg-fhss-extract lab .tar file
README.md	

module <https://github.com/DTHuyn/steg-fhss-extract/raw/master/steg-fhss-extract.tar>

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

File steg-basic-fhss-extract.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-fhss-extract/raw/master/steg-fhss-extract.tar

Vào terminal, gõ:

labtainer -r steg-fhss-extract

```
student@LabtainerVMware:~/labtainer/labtainer-student$ imodule https://github.com/DTHuyn/steg-fhss-extract/raw/master/steg-fhss-extract.tar
Adding imodule path https://github.com/DTHuyn/steg-fhss-extract/raw/master/steg-fhss-extract.tar
Updating IModule from https://github.com/DTHuyn/steg-fhss-extract/raw/master/steg-fhss-extract.tar
student@LabtainerVMware:~/labtainer/labtainer-student$ labtainer -r steg-fhss-extract
latest: Pulling from dthuyn/steg-fhss-extract.sender.student
b5a24e1a655e: Pull complete
7651ce4566c8: Pull complete
8e7e24552aa8: Pull complete
df611ceef7a4: Pull complete
f23f6a51bc25: Pull complete
7ee7f1fb60e9: Pull complete
sha256:15236f... 0.33 MB / 0.33 MB
```

Hình 9: Khởi động 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 2 terminal, một cái là đại diện cho máy gửi: **sender**, một cái là đại diện cho máy nhận: **receiver**. Biết rằng 2 máy nằm cùng mạng LAN 172.20.0.0/24.

Task1: Tiến hành kiểm tra ip của 2 máy

Thực hiện lệnh kiểm tra ip và các thư mục có sẵn trong container sender và receiver.

ip a

ls -l

```
ubuntu@sender:~$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.20.0.12 netmask 255.255.255.0 broadcast 172.20.0.255
    ether ee:4e:d3:27:cf:55 txqueuelen 0 (Ethernet)
    RX packets 50 bytes 6687 (6.6 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 3 bytes 126 (126.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 0 bytes 0 (0.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

ubuntu@sender:~$ ls
message.txt  stego_audio.wav

ubuntu@receiver:~$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.20.0.11 netmask 255.255.255.0 broadcast 172.20.0.255
    ether 8e:95:90:c3:28:e3 txqueuelen 0 (Ethernet)
    RX packets 51 bytes 6797 (6.7 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 3 bytes 126 (126.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 0 bytes 0 (0.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

ubuntu@receiver:~$ ls
extract_message.py  generate_hopping_pattern.py
```

Hình 10: Tiến hành kiểm tra ip của 2 máy

Sinh viên chỉnh sửa nội dung file message.txt từ sender. “HELLO PTIT!”

```
ubuntu@sender:~$ cat message.txt
HELLO PTIT!
ubuntu@sender:~$
```

Hình 11: chỉnh sửa nội dung file message.txt

Trên container receiver, mở file cấu hình ssh và thay đổi các dòng config hoặc thêm cuối file /etc/ssh/sshd_config trên receiver.

Port 22

PasswordAuthentication yes

PermitRootLogin yes

Cần restart lại dịch vụ ssh-server: `sudo systemctl restart ssh`

```
ubuntu@receiver:~$ sudo nano /etc/ssh/sshd_config
ubuntu@receiver:~$ sudo systemctl restart ssh
```

Hình 12: mở file cấu hình ssh và chỉnh sửa

Task2: Tiến hành gửi file audio chứa thông điệp cho người nhận.

Tại terminal sender, tiến hành gửi file audio chứa thông điệp đến người nhận.

`scp stego_audio.wav ubuntu@<ip_máy_receiver>:/home/ubuntu/`

Nhập password là *ubuntu*

```
ubuntu@sender:~$ ls
message.txt  stego_audio.wav
ubuntu@sender:~$ scp stego_audio.wav ubuntu@172.20.0.11:/home/ubuntu/
-bash: 172.20.0.11:/home/ubuntu/: No such file or directory
ubuntu@sender:~$ scp stego_audio.wav ubuntu@172.20.0.11:/home/ubuntu/
The authenticity of host '172.20.0.11 (172.20.0.11)' can't be established.
ECDSA key fingerprint is SHA256:ZtE8xi5Y50aUktZ/XtgjIs1c5jxYQB84Vq5ofmlgGng.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '172.20.0.11' (ECDSA) to the list of known hosts.
ubuntu@172.20.0.11's password:
stego_audio.wav                                100% 431KB 27.1MB/s 00:00
```

Hình 13: Tiến hành gửi file audio chứa thông điệp cho người nhận

Task3: Bên nhận tạo chuỗi nhảy tần đồng bộ với bên gửi

Trên terminal receiver, tạo chuỗi nhảy tần thông qua khóa bí mật *mykey* và độ dài chuỗi nhảy tần 88 bit mà 2 bên trao đổi.

Tiến hành tạo chuỗi:

`python3 generate_hopping_pattern.py`

```

ubuntu@receiver:~$ python3 generate_hopping_pattern.py
Nhập số bit của dãy nhảy tần: 88
88
Nhập khóa bí mật: mykey
mykey
Dãy nhảy tần đã được lưu vào hopping_pattern.txt
ubuntu@receiver:~$ cat hopping_pattern.txt
1 3 3 3 3 1 5 3 5 5 2 5 1 2 5 1 5 4 5 4 3 4 5 1 4 2 5 2 3 1 2 1 4 2 4 3 1 5 2 5 2 5 1
3 2 3 1 2 2 4 1 5 3 2 1 2 4 2 3 2 4 3 3 2 4 4 5 2 4 2 5 4 3 4 5 2 5 4 2 3 3ubuntu@re

```

Hình 14: Bên nhận tạo chuỗi nhảy tần đồng bộ với bên gửi

Nhập số bit là 88 và khóa bí mật là mykey

Kết quả chuỗi nhảy tần được sinh ra và lưu vào file *hopping_pattern.txt*

Đọc file : *cat hopping_pattern.txt*

Task4: Tiến hành tách trích xuất thông tin đã giấu

Kiểm tra các file có hiện tại, đảm bảo đã có file *hopping_pattern.txt* và file âm thanh được nhận từ sender *stego_audio.wav*

Tiến hành bóc tách tin:

python3 extract_message.py

```

ubuntu@receiver:~$ python3 extract_message.py
Đã lưu thông điệp tại message.txt: HELLO PTIT!
ubuntu@receiver:~$ ls
extract_message.py  generate_hopping_pattern.py  hopping_pattern.txt  message.txt  stego_audio.wav
ubuntu@receiver:~$ cat message.txt
HELLO PTIT!ubuntu@receiver:~$

```

Hình 15: : Tiến hành tách trích xuất thông tin đã giấu

Thông điệp nhận được được in ra màn hình, đồng thời lưu vào file *message.txt*

Kiểm tra xem thông điệp nhận có giống bên gửi không?

So sánh thông điệp được tách ra với nội dung file gửi xem có trùng khớp không

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*.

Màn hình checkwork:

```

student@LabtainerVMware:~/labtainer/labtainer-student$ checkwork steg-fhss-extract
Results stored in directory: /home/student/labtainer_xfer/steg-fhss-extract
Successfully copied 122kB to steg-fhss-extract-igrader:/home/instructor/B21DCAT105.steg-fhss-extract.lab
Successfully copied 2.05kB to /home/student/labtainer_xfer/steg-fhss-extract
Labname steg-fhss-extract

Student          | enable_ssh | transfer_file | generate_hoppin | check_secret_ke | check_extracted |
=====|=====|=====|=====|=====|=====|
B21DCAT105      | Y          | Y             | Y              | Y              | Y              |
What is automatically assessed for this lab:

```

Hình 16: Màn hình 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-fhss-extract

TÀI LIỆU THAM KHẢO

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