**HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG**

**KHOA AN TOÀN THÔNG TIN**

****

**BÁO CÁO BÀI TẬP LỚN**

**HỌC PHẦN: CÁC KỸ THUẬT GIẤU TIN**

**MÃ HỌC PHẦN: INT14102**

**CHỦ ĐỀ: THỰC HIỆN GIẤU TIN TRONG ÂM THANH BẰNG PHƯƠNG PHÁP FHSS TỪ CODE PYTHON SỬ DỤNG THƯ VIỆN TKINTER ĐỂ TẠO TOOL**

**Lab: giau\_tin\_fhss\_tool**

| Sinh viên thực hiện | Đặng Minh Hiếu |
| --- | --- |
| Mã sinh viên | B21DCAT087 |

Tên nhóm: Nhóm 10

Tên lớp: Nhóm 3

Giảng viên hướng dẫn: TS. Đỗ Xuân Chợ

**HÀ NỘI 2025**

# Bài lab : Thực hiện giấu tin trong âm thanh bằng phương pháp FHSS từ code Python sử dụng thư viện Tkinter

## Mục đích

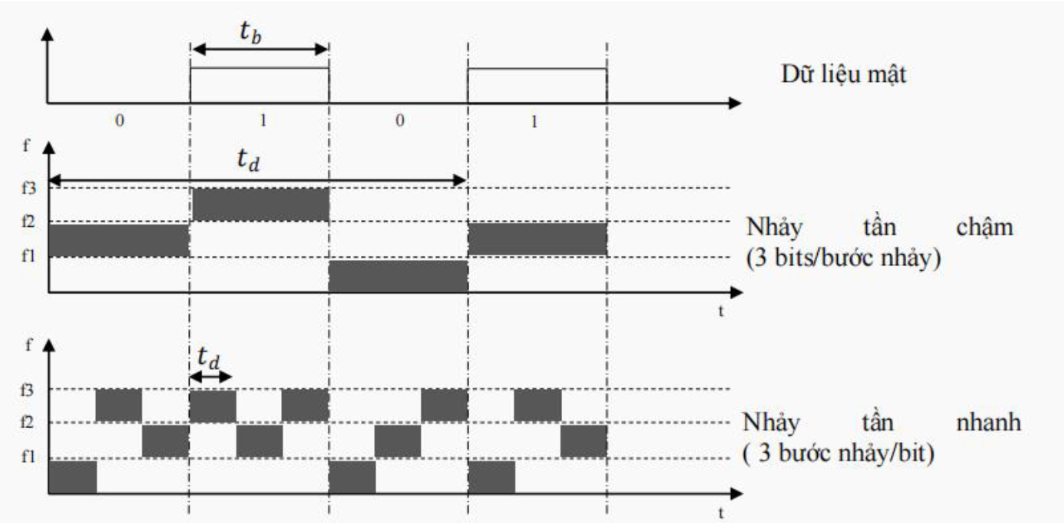
* Giúp sinh viên hiểu được thuật toán giấu tin và tách tin trong âm thanh sử dụng phương pháp FHSS
* Sinh viên làm quen với các hàm trong thư viện Tkinter để thực hiện tạo GUI phục vụ việc tạo tool từ Python

## Yêu cầu đối với sinh viên

* Nắm vững các khái niệm cơ bản về giấu tin (Steganography), đặc biệt là giấu tin trong môi trường âm thanh.
* Hiểu sâu về nguyên lý hoạt động của phương pháp Trải phổ Nhảy tần (FHSS) trong truyền thông và cách nó được ứng dụng để giấu tin trong âm thanh. Điều này bao gồm các thành phần cốt lõi như: chuỗi giả ngẫu nhiên (PN sequence), cơ chế nhảy tần, điều chế FSK (Frequency Shift Keying) cho dữ liệu mật, vai trò của khóa bí mật.
* Hiểu biết về các đặc trưng cơ bản của tín hiệu âm thanh số (tần số lấy mẫu, lượng tử hóa, định dạng file WAV).
* Có kiến thức về cách biểu diễn và xử lý dữ liệu âm thanh trong Python (ví dụ: sử dụng thư viện như scipy.io.wavfile, numpy).
* Thành thạo ngôn ngữ lập trình Python.
* Hiểu rõ cách sử dụng thư viện Tkinter để xây dựng giao diện người dùng đồ họa (GUI), bao gồm việc tạo các widget, quản lý layout, và xử lý sự kiện.

## Nội dung lý thuyết

Trải phổ nhảy tần là một công nghệ sử dụng bộ phát tần số và có thể thay đổi tần số truyền một cách đột ngột trong dãy băng tần sử dụng. Trong trải phổ nhảy tần, độ rộng băng thông sẵn có sẽ được chia thành một số lớn các khe tần không lấn lên nhau. Tại bất kì khoảng thời gian nào, tín hiệu truyền đi đều chiếm một hay nhiều hơn một khe tần số nói trên. Việc chọn một khe hay nhiều khe tần số trong một khoảng thời gian truyền tín hiệu đều được thực hiện một cách giả ngẫu nhiên theo tín hiệu ra của một bộ tạo chuỗi giả ngẫu nhiên. Dựa trên tốc độ nhảy của tần số thì phương pháp trải phổ nhảy tần được chia làm 2 loại đó là trải phổ nhảy tần nhanh và trải phổ nhảy tầm chậm. Trong đó, trải phổ nhảy tần nhanh có đặc điểm là tốc độ nhảy nhanh hơn tốc độ dữ liệu. Còn trải phổ nhảy tần chậm thì có tốc độ nhảy chậm hơn tốc độ dữ liệu. Hình 3.17 mô tả ví dụ của trải phổ nhảy tần nhanh và trải phổ nhảy tần chậm.



#### Quy trình Giấu tin (Phía Máy phát):

***Bước 1: Chuẩn bị thông tin và Mã hóa (Tùy chọn):***

* Thông tin cần giấu (Chuỗi thông tin mật) được đưa vào.
* Thông tin này có thể được đưa qua **Bộ mã hóa**, nơi nó có thể được mã hóa bằng một khóa riêng để tăng tính bảo mật. Bước này là tùy chọn.

***Bước 2: Điều chế FSK:***

* Tín hiệu (đã mã hóa hoặc chưa) được đưa vào **Bộ điều chế FSK (Frequency Shift Keying)**.
* Bộ điều chế FSK chuyển đổi thông tin thành tín hiệu nhị phân x(t), trong đó mỗi bit (0 hoặc 1) tương ứng với một trong hai tần số cụ thể (ví dụ: f′ cho bit 0 và f′+Δf cho bit 1).

***Bước 3: Tạo chuỗi nhảy tần:***

* Một **Bộ tạo chuỗi giả ngẫu nhiên (PN - Pseudorandom Number)** tạo ra một đoạn mã (ví dụ: m bit).
* Đoạn mã này điều khiển một **Bộ tổ hợp tần số (Frequency Synthesizer)** để tạo ra các giá trị tần số nhảy tần y(t) cho sóng mang. Tần số y(t) thay đổi sau mỗi khoảng thời gian Tc​ giây theo giá trị m bit từ bộ tạo chuỗi PN.

***Bước 4: Trộn tín hiệu:***

* Tín hiệu x(t) (từ bộ điều chế FSK) và tín hiệu tần số nhảy y(t) (từ bộ tổ hợp tần số) được đưa vào một **Bộ trộn (Mixer)**.
* Bộ trộn kết hợp hai tín hiệu này, tạo ra các tần số tổng và hiệu. Một trong hai tần số này sẽ được lọc ra (thường bằng bộ lọc băng thông BPF, không thể hiện rõ trong sơ đồ nhưng là một phần của quá trình trộn chuẩn) trước khi đưa vào môi trường chứa (ví dụ: file âm thanh gốc).
* Về bản chất, thông tin mật sẽ được "trải" trên nhiều tần số khác nhau theo một trình tự giả ngẫu nhiên.

#### Quy trình Tách tin (Phía Máy thu):

***Bước 1: Đồng bộ hóa:***

* Đây là bước cực kỳ quan trọng. **Bộ đồng bộ định thời** đảm bảo rằng bộ tạo chuỗi PN ở phía máy thu hoạt động đồng bộ với phía máy phát.

***Bước 2: Tái tạo chuỗi nhảy tần:***

* **Bộ tạo chuỗi PN** ở máy thu (đã được đồng bộ và sử dụng cùng khóa/hạt giống như máy phát) tạo ra lại cùng một chuỗi giả ngẫu nhiên.
* Chuỗi này điều khiển **Bộ tổ hợp tần số** để tạo ra chính xác cùng một chuỗi tần số nhảy y(t) đã được sử dụng ở máy phát.

***Bước 3: Giải trải phổ (De-hopping):***

* Tín hiệu thu được từ môi trường (file âm thanh chứa tin) được đưa vào **Bộ trộn**.
* Tại đây, tín hiệu thu được sẽ được trộn với tín hiệu tần số nhảy y(t) (tái tạo ở bước 2). Nếu đồng bộ chính xác, quá trình này sẽ loại bỏ thành phần nhảy tần, khôi phục lại tín hiệu FSK x(t) ban đầu (sau khi qua bộ lọc BPF thích hợp).

***Bước 4: Giải điều chế FSK:***

* Tín hiệu x(t) được đưa vào **Bộ giải điều chế FSK**.
* Bộ giải điều chế FSK sẽ chuyển đổi tín hiệu tần số trở lại thành chuỗi bit dữ liệu (đã mã hóa nếu có).

***Bước 5: Giải mã (Tùy chọn):***

* Nếu thông tin đã được mã hóa ở phía máy phát, chuỗi bit dữ liệu thu được sẽ được đưa qua **Bộ giải mã** (sử dụng cùng khóa riêng) để khôi phục lại thông tin mật ban đầu.

#### Xử lý Âm thanh số cơ bản và Lập trình Ứng dụng với Python & Tkinter

* Biểu diễn tín hiệu âm thanh: Trong miền thời gian, tín hiệu âm thanh số là một dãy các giá trị biên độ rời rạc được gọi là mẫu (samples).
* Đọc/Ghi file âm thanh: Các file âm thanh (ví dụ, định dạng WAV) lưu trữ thông tin về tần số lấy mẫu, độ sâu bit, số kênh và dữ liệu mẫu. Trong Python, thư viện như scipy.io.wavfile (hàm read, write) thường được sử dụng để làm việc với file WAV.
* Xử lý dữ liệu mẫu:
  + Chuyển đổi kiểu dữ liệu: Dữ liệu đọc từ file WAV thường là số nguyên. Việc chuyển đổi sang kiểu số thực (float) giúp thực hiện các phép toán chính xác hơn (như trong đoạn code bạn cung cấp: astype(float)).
  + Xử lý kênh (Mono/Stereo): File âm thanh có thể có một kênh (mono) hoặc nhiều kênh (ví dụ, stereo có 2 kênh). Nếu là stereo, việc chuyển về mono bằng cách lấy trung bình các kênh có thể được thực hiện để đơn giản hóa xử lý (như trong code: np.mean(audio\_data, axis=1)).
  + Chuẩn hóa biên độ: Đôi khi cần chuẩn hóa biên độ của tín hiệu âm thanh (ví dụ, về khoảng [-1, 1]) trước khi xử lý.

#### Xây dựng Giao diện Người dùng (GUI) với Thư viện Tkinter

* Giới thiệu Tkinter: Tkinter là thư viện GUI chuẩn, tích hợp sẵn trong Python. Nó cung cấp một bộ công cụ để tạo các cửa sổ, nút bấm, hộp văn bản, và các thành phần giao diện khác.
* Các thành phần (Widgets) cơ bản thường dùng:
  + Tk(): Cửa sổ chính của ứng dụng.
  + Label: Hiển thị văn bản hoặc hình ảnh tĩnh.
  + Button: Cho phép người dùng kích hoạt một hành động.
  + Entry: Cho phép người dùng nhập một dòng văn bản ngắn.
  + Text: Cho phép người dùng nhập hoặc hiển thị nhiều dòng văn bản.
  + Frame: Một vùng chứa để tổ chức các widget khác.
  + Các hộp thoại (Dialogs): filedialog (để mở/lưu file), messagebox (để hiển thị thông báo).
* Quản lý layout: Tkinter cung cấp các trình quản lý layout như pack(), grid(), place() để sắp xếp vị trí các widget trong cửa sổ.
* Xử lý sự kiện (Event Handling): Gán các hàm (callbacks) cho các sự kiện của widget (ví dụ, khi nút được nhấn).
* Vai trò trong đề tài: Tkinter sẽ được sử dụng để tạo một ứng dụng có giao diện đồ họa, giúp người dùng dễ dàng tương tác với chương trình giấu tin/tách tin FHSS: chọn file âm thanh, nhập thông điệp/khóa, bắt đầu quá trình xử lý, và xem kết quả.

## Nội dung thực hành

Khởi động bài lab:

* Vào terminal, gõ lệnh:

rebuild giau\_tin\_fhss\_tool

### Nhiệm vụ 1: Đọc tin nhắn cần giấu vào âm thanh

Trước khi thực hiện giấu tin, ta cần đọc trước thông tin sẽ giấu. Thông tin được để trong file message.txt:

***cat message.txt***

### Nhiệm vụ 2: Thực hiện chạy tool và giấu tin

Mở tool có sẵn python

***python3 giau\_tin\_tool.py***

Ở trong tool, ta sẽ chọn những file cần thiết cho quá trình giấu tin

1. File âm thanh gốc (.wav):Là file âm thanh chưa được giấu tin: input.wav
2. Thông điệp bí mật: Có thể chọn viết trực tiếp hoặc chọn file .txt để giấu tin trong đó vào âm thanh
3. Khóa bí mật (quan trọng): Ở đây ta cần đặt 1 khóa bí mật để khi tách tin, ta phải nhập đúng khóa bí mật đó để tách tin chuẩn xác
4. Lưu File Stego (.wav): Ta chọn đường dẫn để lưu file wav đã giấu tin, ở đâu ta lưu file là output.txt
5. Tham số FHSS (tùy chỉnh):

* Thời gian Bit (T\_bit,s) - là khoảng thời gian được sử dụng để truyền (hoặc nhúng) một bit đơn lẻ của thông điệp bí mật: Ta sẽ đặt là 0.01s
* Số lượng tần số (Number of Frequencies) - là tổng số các tần số sóng mang riêng biệt mà hệ thống FHSS có thể lựa chọn để "nhảy" tới trong quá trình hoạt động. Các tần số này nằm trong một dải tần số (bandwidth) đã được xác định trước. Ví dụ, nếu bộ tạo chuỗi PN đưa ra một đoạn mã m bit để chọn tần số, thì bộ tổng hợp tần số có thể tạo ra 2^m giá trị tần số khác nhau: Ta đặt là 6
* Tần số min (Minimum Frequency) - Là tần số thấp nhất trong dải tần mà hệ thống FHSS sử dụng để nhảy: Ta đặt là 1000 Hz
* Tần số max (Maximum Frequency) - Là tần số cao nhất trong dải tần mà hệ thống FHSS sử dụng để nhảy: Ta đặt là 5000 Hz

Việc cuối cùng ta ấn nút “Bắt đầu giấu tin”

### 

### Nhiệm vụ 3: So sánh kích thước của file âm thanh gốc và file âm thanh sau khi giấu tin

Để thực hiện so sánh kích thước của 2 file âm thanh, ta sẽ sử dụng file python có sẵn để so sánh sự khác biệt giữa kích thước

***python3 compare\_file\_size.py input.wav output.wav***

Sau khi chạy python để so sánh, hệ thống sẽ in ra kích thước của 2 file và so sánh 2 file lớn hơn/bé hơn bao nhiêu so với file còn lại

### Nhiệm vụ 4: So sánh BER (Bit Error Rate) của file âm thanh gốc và file âm thanh sau khi giấu tin

Để thực hiện so sánh BER của 2 file âm thanh, ta sẽ sử dụng file python có sẵn để so sánh sự khác biệt bit error rate

***python3 compare\_file\_ber.py input.wav output.wav***

Sau khi chạy python, hệ thống sẽ in ra thông tin frequency samples, kiểu dữ liệu và số kênh của 2 file âm thanh

Sau đó hệ thống sẽ chỉ ra số bit lỗi giữa âm thanh gốc và âm thanh đã giấu tin. Và cuối cùng là tỷ lệ lỗi bit của dữ liệu âm thanh (bit)

## Kết quả cần đạt được

* Chạy được tất cả các bước như yêu cầu.

## Kết thúc bài lab:

Sử dụng lệnh: “checkwork” để kiểm tra số checkwork đã làm đúng.

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

***stoplab giau\_tin\_fhss\_tool***

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

Để kiểm tra kết quả khi trong khi làm bài thực hành sử dụng lệnh:

***checkwork giau\_tin\_fhss\_tool***

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

***labtainer –r giau\_tin\_fhss\_tool***