**BỘ THÔNG TIN TRUYỀN THÔNG**



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

----------

**Kỹ thuật giấu tin trong văn bản**

**Lab: gnf-stega**

**Sinh viên:** Lê Văn Tuấn

**Mã sinh viên:** B21DCAT020

**Bộ môn:** Kỹ thuật giấu tin

**Giảng viên hướng dẫn:** PGS.TS. Đỗ Xuân Chợ

**Hà Nội – 2025**

1. **Giới thiệu chung về bài thực hành**

Ở bài thực hành này, sinh viên sẽ được làm quen với phương pháp sử dụng văn phạm phi ngữ cảnh. Sử dụng một văn phạm phi ngữ cảnh (CFG-Context Free Grammar) để sinh ra các câu tạo thành văn bản nhân tạo chứa thông điệp bí mật và văn bản này có thể bắt chước văn bản thực tế. Văn phạm phi ngữ cảnh là một tập hợp hữ hạn các biến (còn được gọi là các ký hiệu chưa kết thúc), mỗi biến biểu diễn một ngôn ngữ. Ngôn ngữ được biểu diễn bởi các biến được mô tả một cách đệ quy theo thuật ngữ của một khái niệm khác gọi là ký hiệu kết thúc. Quy tắc quan hệ giữa các biến gọi là luật sinh. Mỗi luật sinh. Mỗi luật sinh có dạng một biết, ở vế trái sinh ra một chuối có thể gồm biến lẫn các ký hiệu là văn phạm G (V, T, P, S), trong đó:

* V là tập hợp hữu hạn các biến
* T là tập hợp hữu hạn các ký tự kết thúc V ∩ T = ∅
* P là tập hợp hữu hạn các luật sinh mà mỗi luật sinh có dạng A → α (với A là biến và α là chuỗi các ký hiệu ∈ (V∪T)\*)
* S là một biến đặc biệt gọi là ký hiệu bắt đầu văn phạm.

Một CFG ở dạng Greibach normal form (GNF) nếu kí hiệu chưa kết thúc luôn là sự lựa chọn cuối cùng trong các lựa chọn của luật sinh. Ví dụ, luật sinh có dạng something → A B | C D thuộc dạng GNF nhưng luật sinh dạng “blah → the size sum | a size bell” thì không thuộc dạng GNF. Tuy nhiên có thể sửa một luật không thuộc dạng GNF thành GNF bằng cách thêm các luật như sau:

**adjective** → the **sizesum** | a **sizebell**

**sizesum** → tiny sum | small sum | large sum | big sum

**sizebell** → tiny bell | small bell | large bell | big bell

1. **Nội dung và hưỡng dẫn thực hành**
   1. **Mục đích**

Thực hành mã hóa và giải mã thông tin nhị phân bằng văn bản phi ngữ pháp dựa trên luật sinh GNF.

* 1. **Yêu cầu người thực hành**

Sinh viên có kiến thức cơ bản về mật mã cơ sở, và có khả năng sử dụng lệnh linux cơ bản

* 1. **Nội dung thực hành**

Sinh viên khởi động bài lab

Chạy lệnh

*labtainer -r gnf-stega*

*(Chú ý: Sinh viên sử dụng <Tên\_tài\_khoản> của mình để nhập thông tin người thực hiện bài lab khi có yêu cầu, để sử dụng khi chấm điểm)*

Sau khi khởi dộng lab xong, 1 container hiện lên và sinh viên thực hiện làm theo các nhiệm vụ:

**Task 1:**

Mục tiêu: Chuyển từ 1–4 ký tự thành chuỗi nhị phân và lưu vào binary.txt

Thao tác:

*Python3 gnf\_utils.py*

Sau đó nhập chuỗi “jack” kết quả sinh ra file binary.txt

**Task 2:**

Đọc chuỗi nhị phân từ binary.txt, sử dụng luật gnf\_rules.cfg để mã hóa thành câu văn tự nhiên.

Thao tác:

*Python3 hide\_message.py*

File message.txt được tạo

**Task 3:**

Đọc message.txt và khôi phục lại dữ liệu nhị phân -> chuyển thành kí tự gốc.

Thao tác:

*Python3 extract\_message.py*

Kết quả hiển thị file decoded.txt chứa “jack”

Kết thúc lab:

* Trên terminal khởi động lab, sinh viên sử dụng lệnh:

*stoplab*

* Khi bài lab kết thúc, một tệp lưu kết quả đuọc tạo và lưu vào một vị trí được hiển thị bên dưới stoplab. Sinh viên cần nộp file .lab để chấm điểm.
* Để kiểm tra kết quả trong khi làm bài thực hành sử dụng lệnh:

*checkwork <tên bài lab>*

* Sinh viên cần nộp file .lab để chấm điểm.
* Kiểm tra kết quả trong quá trình làm bài:

*checkwork <tên bài lab>*

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

*labtainer -r gnf-stega*