

**UNIVERSIDAD DEL VALLE DE GUATEMALA**

**CC3094 - SECURITY DATA SCIENCE**

**Sección 10**

**Ing. Jorge Yass**



## **Proyecto: Fase 1**

Carlos Alberto Raxtum Ramos, 19721

Javier Alejandro Cotto Argueta, 19324

Juan Manuel Marroquin Alfaro, 19845

Jose Abraham Gutierrez Corado, 19111

Walter Danilo Saldaña Salguero, 19897

**GUATEMALA, 23 de febrero de 2023**

# Motivación


Desarrollar un simulador de middleware para correo electrónico que analice las entradas para detectar posibles stego-malwares en PDF, PNG, JPEG, TXT, y entre otros formatos (se investigará cuáles son los formatos más enviados por email para enfocarnos en esos).

El motivo para realizar este simulador de middleware es para que las personas estén al tanto si los archivos que les mandan son maliciosos o no. Además ser conscientes de las posibles brechas de seguridad a las cuales estos sufren al recibir y descargar cualquier archivo sin previo aviso.

## Preguntas Clave

- ¿Qué tipos de stego-malware existen, sus características y su respectiva detección?
- ¿Qué tan avanzado y qué impacto puede ser/tener un stego-malware?
- ¿Cuáles son los indicios que pueden indicar la presencia de un stego-malware en un documento?
- ¿Existen posibles indicadores que desencadenan un FN o FP en la predicción a los que nos podamos anticipar para preprocesar los datos?

## Revisión de Literatura

- Github: [https://github.com/ncanceill/pdf\\_hide](https://github.com/ncanceill/pdf_hide)
  - Este repositorio da una explicación detallada de una librería que nos ayudará a incrustar *stegomalware* dentro de un PDF. El motivo de esta librería es para contemplar el caso de que no hayamos una base de datos que tuviera pdf alterados con malware para poder entrenar nuestro modelo.
- Passi, H. (2013). Top 10 Must-Have tools to perform Steganography. GreyCampus. <https://www.greycampus.com/blog/information-security/top-must-have-tools-to-perform-steganography>
- NullByte. (2018). Conceal Secret Messages or Data Through Steganography with Steghide [Tutorial]. [Www.youtube.com](https://www.youtube.com/watch?v=...).  
 Conceal Secret Messages or Data Through Steganography with Steghide [Tutorial]
  - Estas referencias nos servirán para alterar imágenes tipo JPG, PNG, BMP con *stegomalware* en su estructura. Basado en esto podemos utilizarlos para probar nuestro modelo luego de haberlo entrenado.
- En la siguiente referencia se abordan los diferentes tipos de stegomalware y sus características considerando técnicas avanzadas de “invisibilización” de los mismos

- Suarez-Tangil, G., Tapiador, J. E., & Peris-Lopez, P. (2015). Stegomalware: Playing Hide and Seek with Malicious Components in Smartphone Apps. *Information Security and Cryptology*, 496-515. [https://doi.org/10.1007/978-3-319-16745-9\\_27](https://doi.org/10.1007/978-3-319-16745-9_27)
- Servidor de correos:
  - Esta herramienta nos permite simular un servidor de transmisión de correos, donde con ayuda de esto podemos ir enviando correo al modelo de python y de esa forma el modelo determina si el correo que estamos enviando tiene en su contenido/archivos adjuntos un stegomalware.

## Recolección de Datos

La data que necesitamos para realizar este proyecto se basa en archivos de los 5 formatos principales que definimos alterados con un *stegomalware* “dentro” de ellos, para así poder realizar nuestro detector de correo malicioso. Kaggle es una plataforma gratuita que brinda a los usuarios varias series de problemas con su respectivo *dataset*, dentro de nuestra búsqueda encontramos soluciones de modelos que detectan *stegomalware* en imágenes en blanco y negro. También tenemos pensado utilizar herramientas que permitan esconder texto dentro de la estructura del archivo sin que este cambie. Pdf-hide es una librería de python que permite esconder texto dentro de la estructura del pdf para simular el *stegomalware*. Herramientas como *steghide*, *invisible secret 4* permitiran obtener esta data.

Principalmente planeamos alterar la estructura del documento para “emular” un stegomalware dentro de un archivo para que el modelo detecte que el documento original ha sido alterado de alguna manera, que en este caso, ha sido inyectado de un malware en su estructura.