



udp UNIVERSIDAD
DIEGO PORTALES

FACULTAD DE INGENIERÍA
ESCUELA DE INFORMÁTICA Y TELECOMUNICACIONES

LAB 1 CRIPTOGRAFÍA Y SEGURIDAD EN REDES

SECCION 1

Alumno:

Kevin Cabrera

Correo:

Kevin.cabrera@mail.udp.cl

agosto - 2024

Índice

| | |
|--------------------------------------|----------|
| 1. Desarrollo Actividades | 2 |
| 1.1. Paso 1 | 2 |
| 1.2. Captura N°1: | 2 |
| 1.3. Paso 2 | 2 |
| 1.4. Captura N°2: | 2 |
| 1.5. Paso 3 | 4 |
| 1.6. Captura N°3: | 4 |
| 2. Issues | 4 |
| 2.1. I1 | 4 |
| 2.2. I2 | 5 |
| 2.3. I3 | 5 |
| 2.4. I4 | 5 |
| 3. Conclusiones y Comentarios | 5 |
| 4. Bibliografía: | 5 |

1. Desarrollo Actividades

1.1. Paso 1

Para comenzar, se procede a generar un programa, en python3 utilizando chatGPT, que permita cifrar texto utilizando el algoritmo Cesar. Como parámetros de su programa deberá ingresar el string a cifrar y luego el corrimiento.

1.2. Captura N°1:

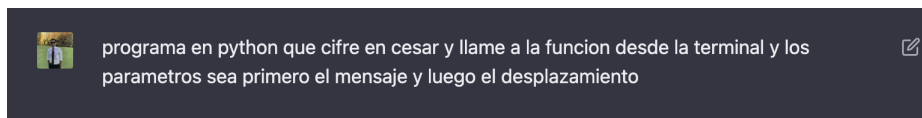


Figura 1: Aqui se le solicita a ChatGPT que haga el programa según lo indicado

Luego de obtener el código en python en ChatGPT y ejecutarlo, se obtiene lo siguiente:

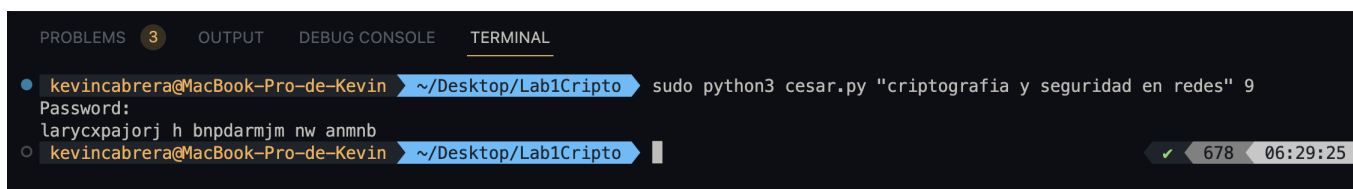
A screenshot of a terminal window. At the top, there are tabs: "PROBLEMS" (with a count of 3), "OUTPUT", "DEBUG CONSOLE", and "TERMINAL" (which is selected). Below the tabs, the terminal shows a command prompt for a user named "kevincabrera@MacBook-Pro-de-Kevin" in the directory "~/Desktop/Lab1Cripto". The command entered is "sudo python3 cesar.py 'criptografia y seguridad en redes' 9". The output of the command is "Password: larycxpajorj h bnpdarmjm nw anmnb". Below the output, there is another prompt line for the same user and directory. In the bottom right corner of the terminal window, there is a status bar showing a green checkmark, the number "678", and the time "06:29:25".

Figura 2: Resultado de la encriptacion por cesar

1.3. Paso 2

Una vez terminado el **Paso 1** Generar un programa, en python3 utilizando ChatGPT, que permita enviar los caracteres del string (el del paso 1) en varios paquetes ICMP request (un caracter por paquete en el campo data de ICMP) para de esta forma no gatillar sospechas sobre la filtración de datos. Deberá mostrar los campos de un ping real previo y posterior al suyo y demostrar que su tráfico consideró todos los aspectos para pasar desapercibido.

1.4. Captura N°2:

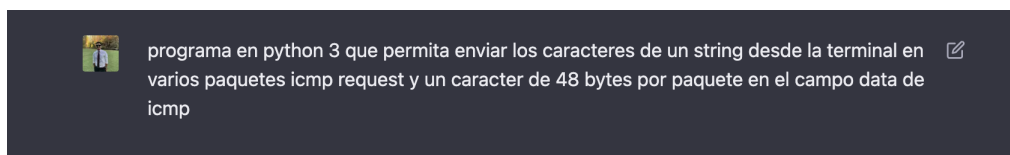


Figura 3: Aqui se le solicita a ChatGPT que haga el programa según lo indicado

Al de obtener el código en python en ChatGPT y ejecutarlo, se obtiene lo siguiente:

```
kevincabrera@MacBook-Pro-de-Kevin: ~/Desktop/Lab1Cripto$ sudo python3 ping.py "larycxpajorj h bnpdarmjm nw anmnb"
Password:
Sent 1 packets.
.
Sent 1 packets.
.
```

Figura 4: Resultado al ejecutar el programa

Junto a la ejecución del programa anterior, se obtiene la siguiente captura del trafico generado en wireshark:

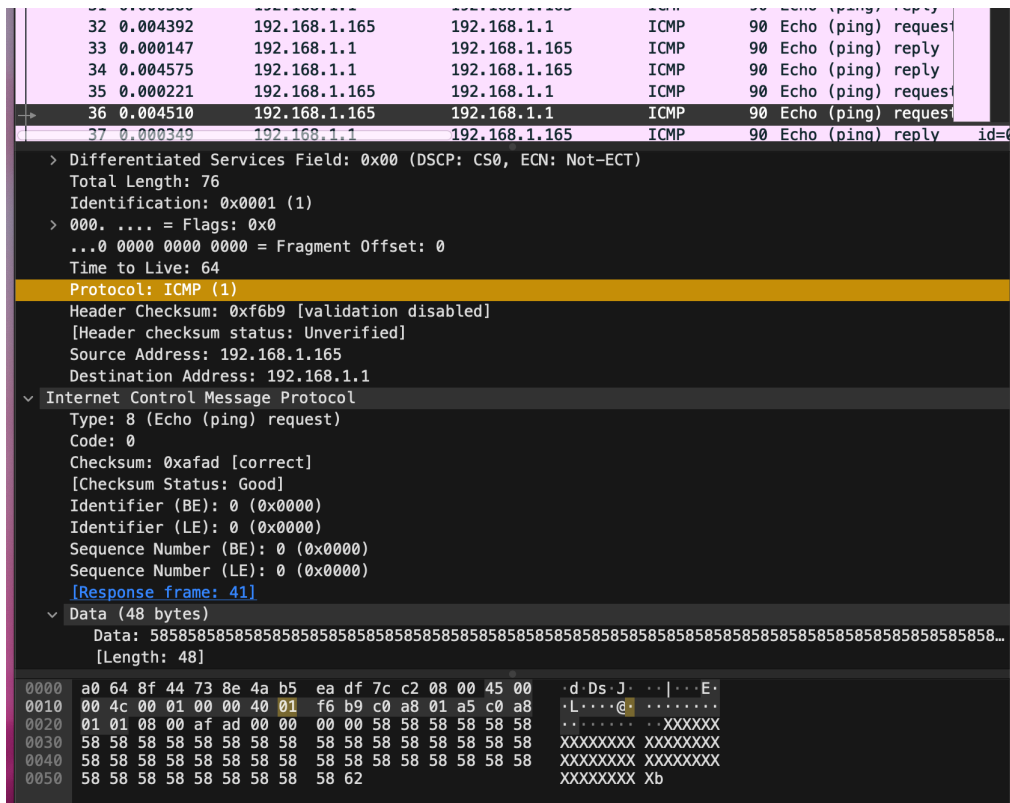


Figura 5: Se muestra los paquetes ICMP transmitidos según lo pedido

Por tanto, en la imagen anterior se muestra claramente que el mensaje encriptado se transmitió de forma correcta mediante el protocolo ICMP y con un data length de 48 bytes. Donde se le añadió un padding con la letra X para que cumpliera con el largo y se muestra que la ultima letra fue la letra "b".

1.5. Paso 3

Generar un programa, en python3 utilizando ChatGPT, que permita obtener el mensaje transmitido en el paso2. Como no se sabe cual es el corrimiento utilizado, genere todas las combinaciones posibles e imprímalas, indicando en verde la opción más probable de ser el mensaje en claro.

1.6. Captura N°3:

```
ls
cesar.pcapng cesar.py pingv4.py readv2.py

~/desktop/Criptografia/Lab1 Cripto (4.174s)
sudo python3 readv2.py cesar.pcapng
Password:
WARNING: No IPv4 address found on anpi1 !
WARNING: No IPv4 address found on anpi0 !
WARNING: more No IPv4 address found on en3 !
Mensaje cifrado extraído:
larycxpajorj h bnpdarmjm nw anmnb

Todas las posibles combinaciones del mensaje:
0 larycxpajorj h bnpdarmjm nw anmnb
1 kzqxbwozinqi g amoczqlil mv zmlma
2 jypwavnymph f zlnbypkhh lu yklkz
3 ixovzumxglog e ykmaxojgj kt xkjky
4 hwnuytlwfknd d xjlzwnifi js wjijx
5 gvmtxskvejme c wikymveh ir vihiw
6 fulswrjudild b vhjxulgdg hq uhghv
7 etkrvqitchkc a ugiwtkfcf gp tgfgu
8 dsjquphsbgjb z tfhvsjebe fo sfeft
9 criptografia y seguridad en redes
10 bqhosnfqzehz x rdftqhczc dm qdcdr
11 apgnrmepdygy w qcespgbyb cl pbcq
12 zofmqldoxcfx v pbdrofaxa bk obabp
13 ynelpkcnwbew u oacqnezvz aj nazao
14 xmdkojbmvadv t nzbpmdyvy zi mzyzn
15 wlcjniauzcu s myaolcxux yh lyxym
16 vkbinhzktybt r lxznkbwtw xg kxwxl
17 ujahlgysxas q kwymjavsv wf jvwvk
18 tizgkfxirwzr p jvxlizuru ve ivuvj
19 shyfjewhqvyq o iuwkhytqt ud hutui
20 rgxeidvgpuxp n htvjgxspz tc gtsth
21 qfwdhucufotwo m gsuifwrwr sb fsrsg
22 pevcgbtensvn l frthevqng ra erqrf
23 odubfasdmrum k eqsgdupmp qz dqpqe
24 nctaezrcqltl j dprfctolo py cpopd
25 mbszdyqbkpsk i coqebnskn ox bonoc
```

Figura 6: Se muestra el mensaje original en verde según lo pedido

En la imagen anteriormente expuesta, se muestra la ejecución de un programa en python 3 según lo especificado en las instrucciones, el cual lee una captura de trafico .pcapng, obtiene el ultimo carácter de cada paquete ICMP transmitido en el paso anterior, hasta llegar al mensaje cifrado, y luego procede a descifrar el mensaje probando con todos los corrimientos del algoritmo cesar y coloreando en verde el mensaje mas probable de ser el original.

2. Issues

2.1. I1

Que haga el programa es relativamente simple pero cuesta que el formato con el que lo entregue sea el deseado.

2.2. I2

El tema de las librerías es que no se especifica la versión del lenguaje con el que se va a trabajar puede estar des-actualizado.

2.3. I3

A veces el programa generado hace lo que se pide según los parámetros pero no de forma correcta.

2.4. I4

En ocasiones hay que corregir el código porque no viene con todas las librerías necesarias por ejemplo.

3. Conclusiones y Comentarios

En este laboratorio, exploramos conceptos esenciales de criptografía y seguridad en redes mediante la implementación de varios programas en Python. Las actividades incluyeron la creación de un cifrado César, la transmisión de datos cifrados a través de paquetes ICMP para evitar su detección, y la posterior decodificación del mensaje sin conocer previamente el corrimiento utilizado.

Se destacó la importancia de comprender no solo los conceptos teóricos de la criptografía, sino también su aplicación práctica en situaciones reales. Implementar el cifrado César y manipular el tráfico de red con ICMP permitió ver de manera tangible cómo se pueden proteger los datos y cómo evitar que sean detectados en la red. Aunque enfrentamos algunos desafíos, como el manejo de librerías y ajustes necesarios en el código, logramos completar las tareas exitosamente.

Este laboratorio también se ve la importancia de un enfoque cuidadoso y crítico al desarrollar soluciones de seguridad, garantizando que cumplan con los requisitos y estándares necesarios.

4. Bibliografía:

<https://chat.openai.com/chat/fb1376b7-7ced-4d5b-bcbc-8b1e2a66aab8>