Universidad San Francisco de Quito

Ingeniería en Ciencias de la Computación

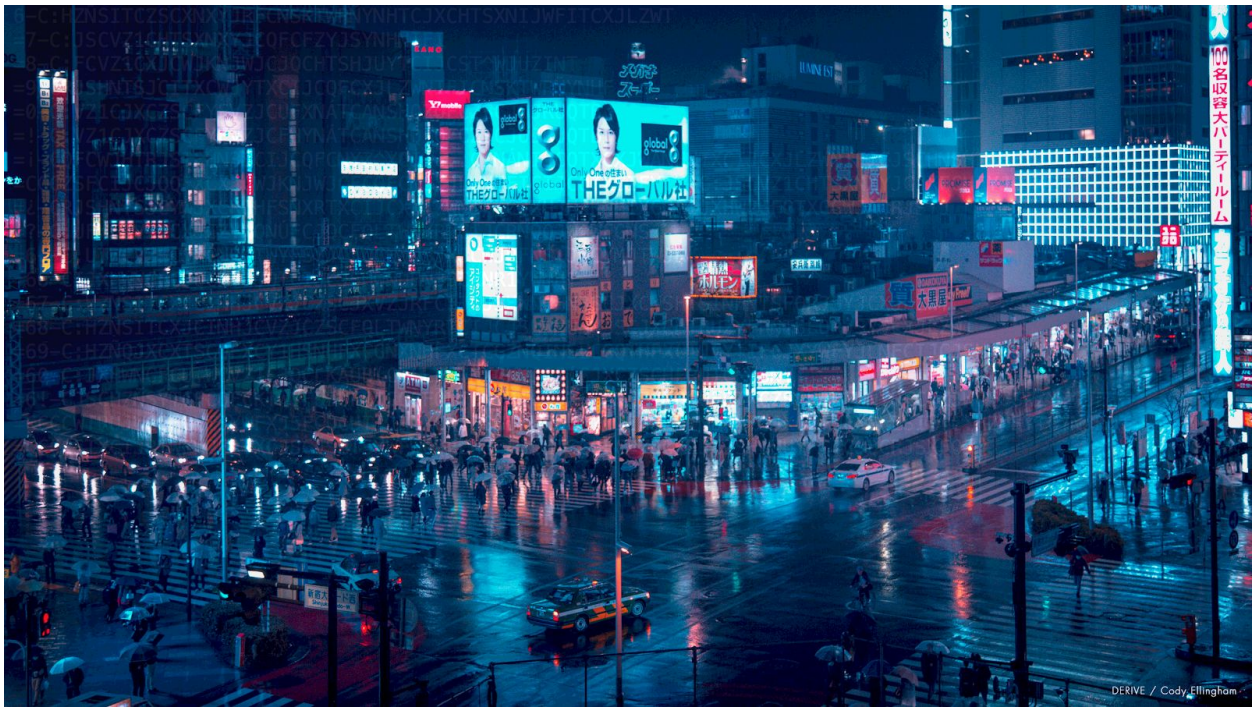
CMP 5006: Seguridad Informática

Examen Parcial

Pablo Llanes - 00133203

Quito, 25 de Octubre del 2020

Parte 1: Procesamiento de imagen y obtención del mensaje



La primera parte del examen consistía en obtener un mensaje oculto en la siguiente imagen, para ilustrar el concepto de esteganografía.
Al ampliar la imagen y al mostrarla en una pantalla con mejor resolución a simple vista se puede observar que hay un texto oculto en la parte izquierda de la imagen, el cual se pierde en el fondo de la imagen. Con el fin de leer y procesar la imagen, y el texto oculto se utilizaron 3 módulos de Python: OpenCV, Pillow y Tesseract.

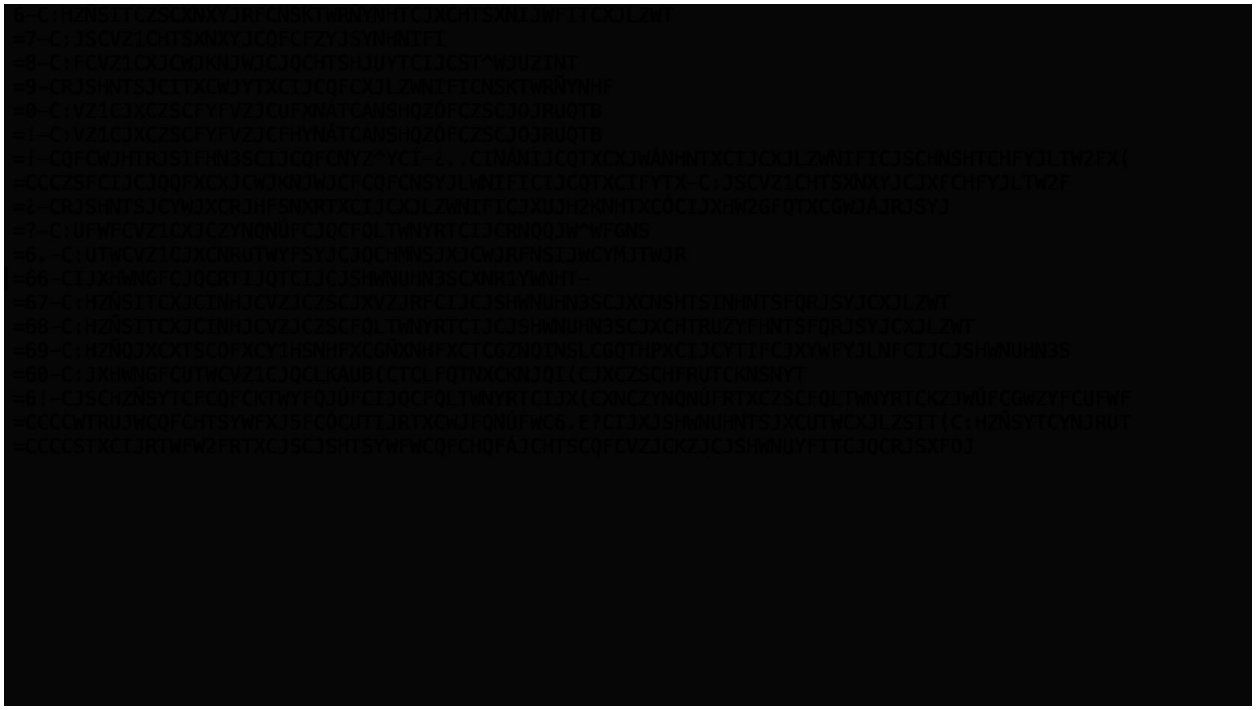
Resumen de los módulos utilizados

OpenCV es un módulo especial que permite leer y procesar imágenes, y las almacena en arreglos bidimensionales que están relacionados con la librería numpy. Este es un módulo extenso que tiene una amplia funcionalidad, pero que no es tan simple de usar (Radečić, 2019).

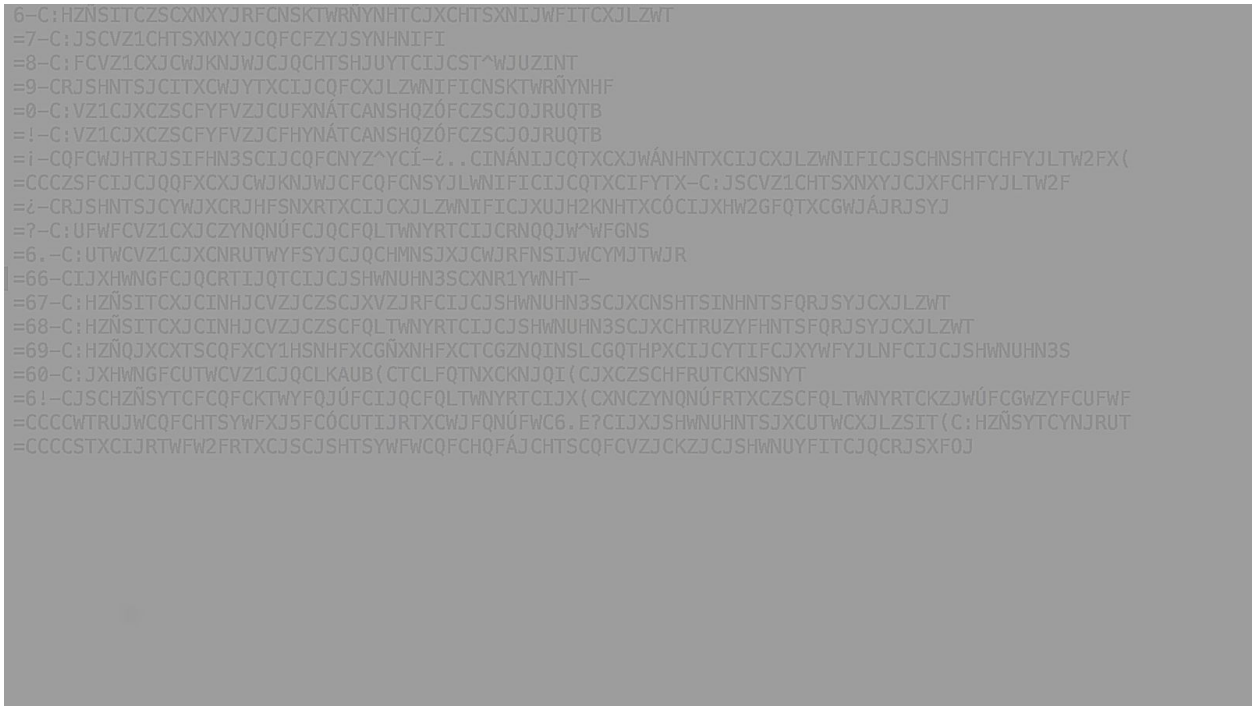
Pillow es una interfaz amigable para un módulo de procesamiento de imágenes llamado PIL, este es un módulo sumamente intuitivo y facilita la escritura de código por lo que va a ser utilizado en este proyecto.

Tesseract es un motor de reconocimiento de caracteres (OCR en inglés) auspiciado por google, el cual captura texto de imágenes y permite procesarlos en el código (Radečić, 2019). A pesar de que Tesseract es un módulo extensamente utilizado tiene sus limitaciones y no siempre entrega los mejores resultados. Para esto la imagen debería ser preprocesada de la mejor manera y tomar en cuenta las siguientes acciones. Para obtener mejores resultados se puede invertir el color de las imágenes, aumentar el tamaño del texto, binarizar el texto y convertirlo en blanco y negro, remover ruido en la imagen, entre otras (Anónimo, s.f.).

Como Tesseract entrega mejores resultados cuando la imagen tiene un fondo claro para obtener el texto se va a proceder a convertir el archivo a blanco y negro (SHI, 2020). Como una imagen es una matriz de píxeles, esta va a ser leída de píxel en píxel y por cada pixel de **tres bytes en formato (RGB)** se va a proceder hacer la siguiente conversión: **si el color tiene un valor par entonces se va a convertir en blanco y si tiene un valor impar se lo transforma a negro.** El resultado de hacer esta conversión es la siguiente:



En este caso el texto de la imagen aún no es muy reconocible, por ese motivo se la procesó utilizando un editor de texto online y el resultado es el siguiente:



Tal como se puede apreciar en la figura, el texto ya tiene un mejor contraste con el fondo y a simple vista ya es reconocible, por lo que con esto ya se pueden obtener mejores resultados. Una vez obtenida esta imagen se la incluye como input en el método **image_to_string** del módulo Tesseract y con esto se obtiene el mensaje cifrado.

Mensaje Cifrado

6-C:HZNSTTCZSCXNXYJIRFCNSKTWRNYNHTCIXCHTSXNIJWFIICXILZWT
=7-C:JSCVZ1CHTSXNXYJCQFCFZYISYNHNIFI
=8—C:FCVZ1CXICWIKNIWICIQCHTSHIUYTCLICST“WIUZINT
=9-CRISHNTSJCLITXCWIYTXCLICQFCXILZWNIFICNSKTWRNYNHF
=Q-C:VZ1CJXCZSCFYFVZICUFXNATCANSHQZOFCZSCJOJRUOTB
=!—C:VZ1CJXCZSCFYFVZICFHYNATCANSHQZOFCZSCJOJRUOTB
=i-COFCWIHTRISIFHN3SCLICQFCNYZ*YCI—é..CINANIJCQTXCXJWANHNTXCIIJCXI
LZWNIFICISCHNSHTCHFYJLTW2FX(

=CCCZSFCLICIOOFXCXICWIKNIWICFCQFCNSYILWNIFICLIJICQTXCLFYTX—C:JSC
VZICHTSXNXYICIXFCHFYILTW2F
=é-CRISHNTSJICYWIXCRIHFSNXRTXCLICXILZWNIFICIXUJH2KNHTXCOCLIXHW2G
FOTXCGWIAIRISYIJ
=?-C:UFWFCVZ1CXJICZYNONUFCIOCFOLTWNYRTCIJCRNQQJW“WFGNS
=§.—C:UTWCVZ1CIXCNRUTWYFSYJCJIQCHMNSIXICWIRFNSIIWCYMITWIR
|=66—-CLIXHWNGFCIQCRTIIQTCLICISHWNUHN3SCXNR1YWNHT—
=67-C:HZNSITCXICINHIJCVZICZSCIXVZIRFCIICISHWNUHN3SCIXCNSHTSINHNTSF
ORISYJCXILZWT
=68-C:HZNSITCXICINHJCVZJICZSCFOLTWNYRTCIJCJISHWNUHN3SCIXCHTRUZYF
HNTSFORISYJCXILZWT
=69-C:HZNOJIXCXTSCOFXCYLHSNHFEXCGNXNHFXCTCGZNGINSLCGOTHPXCLIC
YTLFCIXYWFYILNFCIICISHWNUHN3S
=60-C:JXHWNGFCUTWCVZ1CJOCLKAUB(CTCLFQTNXCKNJOQI(CIXCZSCHFRUTCK
NSNYT
=6!-CJSCHZNSYTCFCOFCKTWYFQJUFCIJOQCFOLTWNYRTCLIX(CXNCZYNONUFRT
XCZSCFOLTWNYRTCKZIWUFCGWZYFCUFWF
=CCCCWIRUIJWCOFCHTSYWFXISFCOCUTIIRIXCWIFONUFWCS.E?CLIXISHWNUHN
TSIXCUTWCXILZSIT(C:HZNSYTCYNJRUT
=CCCCSTXCLIIRTWFW2FRTIXCISCISHTSYWFWCOFCHOFAJCHTSCOFCVZICKZICIS
HWNUYFITCIQCRISXFOJ

En este caso el mensaje cifrado es completamente irreconocible, pero se va a proceder a utilizar el cifrado césar para ver si hay alguna llave con la cual se está cifrando el mensaje.

Parte 2: Desencriptación del mensaje

Para el cifrado César, se va a realizar un ataque de fuerza bruta utilizando 50 llaves distintas, un valor aleatorio que se determinó para un número de archivos razonable que no sea muy difícil de analizar manualmente o utilizando otros algoritmos. Algunos scripts del código se muestran a continuación.

decrypter.py

```
'''
    PARTE II: PROCESO DE DESENCRIPCION DEL MENSAJE

'''

# text = caesar.decrypt("EXXEGOexsrgi", 4)
# print("Decripted message: ", text)

for key in range(1, 51):
    # print("Key: ", key)
    with open("decoded-message" + str(key) + ".txt", "w") as dFile:
        for line in ciphered_text_lines:
            text = caesar.decrypt(line, key)
            dFile.write(text + "\n")

deciphered_text = []
```

caesar.py

```
1  ALPHABET_LOWER = "abcdefghijklmnopqrstuvwxyz"
2  ALPHABET_UPPER = "ABCDEFGHIJKLMNOPQRSTUVWXYZ"
3  ALPHABET_LENGTH = 26
```

```
5   def decrypt(text, key):
6       length = len(text)
7       ciphered_val = 0
8       original_char = ""
9       original_text = ""
10
11      for i in range(length):
12          if(text[i].isalpha()):
13              if (text[i].isupper()):
14                  index = ALPHABET_UPPER.find(text[i])
15
16                  if(index != -1):
17                      ciphered_val = (index - key) % ALPHABET_LENGTH
18                      original_char = ALPHABET_UPPER[ciphered_val]
19
20              else:
21                  index = ALPHABET_LOWER.find(text[i])
22
23                  if(index != -1):
24                      ciphered_val = (index - key) % ALPHABET_LENGTH
25                      original_char = ALPHABET_LOWER[ciphered_val]
26          else:
27              original_char = text[i]
28
29          original_text += original_char
30
31      return original_text
```

Tras hacer el proceso de descripción con las 50 llaves y analizarlas una por una, se determinó que el archivo que lograba mostrar un mensaje legible fue el archivo 31, el mensaje obtenido se muestra a continuación:

Mensaje Descifrado (archivo decoded-message31.txt)

6-X:CUINOOXUNXSISTEDMAXINFORMITICOXDSXCONSIDERADDXSDGURO
=7-X:ENXQU1XCONSISTEXLAXAUTDNTICIDAD
=8—X:AXQU1XSDXRDFIDRDXDLXCONCDPTOXGDXNO“RDPUDIO
=9-XMDNCIONEXGDOSXRDTOSXGDXLAXSDGURIDADXINFORMITICA
=L-X:QU1XESXUNXATAQUDXPASIVOXVINCLUJAXUNXEJEMPJOW
=!—X:QU1XESXUNXATAQUDXACTIVOXVINCLUJAXUNXEJEMPJOW
=d-XJAXRDCOMDNDACI3NXGDXLAXITU*TXD——..XDIVIDEXLOSXSERVICIOSXD
DEXSDGURIDADXDNXCINCOXCATEGOR2AS(
=XXXUNAXGDXDJJASXSDXRDFIDRDXAXLAXINTDGRIDADXGDEDXLOSXGATOS
—X:ENXQUDXCONSISTDXDSAXCATDGOR2A
==-XMDNCIONEDXTRDSXMDCANISMOSXGDXSDGURIDADXDSPEC2FICOSXJXGDS
CR2BAJOSXBRDVDMDNTDE
=?-X:PARAXQU1XSEDXUTIJIPAXDJXAJGORITMOXDEXMILLER“RABIN
=§.—X:PORXQU1XDSXIMPORTANTEXEDLXCHINDSDXRDMAINDDRXTHDORDM
|=66—-XGDSCRIBAXDLXMODDLOXGDXDNCRIPCI3NXSIM1TRICO—
=67-X:CUINDOXSDXDICDEXQUDXUNXDSQUDMAXDDXDNCRIPCI3NXDSXINCOND
ICIONAJMDNTEXSDGURO
=68-X:CUINDOXSDXDICEXQUEDXUNXAJGORITMOXDEXEDNCRIPCI3NXDSXCOMP
UTACIONAJMDNTEXSDGURO
=69-X:CUIJEDSXSONXJASXTGCNICAZSXBISICASXOXBUIBDINGXBJOCKSXGDXTO
GAXDSTRATDGIAXDDXDNCRIPCI3N
=60-X:ESCRIBAXPORXQU1XEJXGFVPW(XOXGALOISXFIEJLD(XDSXUNXCAMPOXFI
NITO
=6!-XENXCUINTOXAXJAXFORTALEPAXDEJLXAJGORITMOXGDS(XSIXUTIJIPAMOS
XUNXAJGORITMOXFUDRPAXBRUTAXPARA

=XXXXRDMPDERXJAXCONTRASDNAXJXPODDMDSXRDAJIPARXN.Z?XGDSDNCRIPCIONDSXPORXSDGUNDO(X:CUINTOXTIEMPO
=XXXXNOSXGDDMORAR2AMODSXDNXDNCONTRARXJAXCJAVEXCONXJAXQUDXFUDXDNCRIPTADOXDLXMDNSAJE

decrypter.py

```
78      with open("decoded-message31.txt", "r") as rFile:
79          line = rFile.readline()
80          while line:
81              deciphered_text.append(line)
82              line = rFile.readline()
83
84      length = len(deciphered_text)
85      for i in range(length):
86          deciphered_text[i] = deciphered_text[i].replace("X", " ")
87          deciphered_text[i] = deciphered_text[i].replace("=", "")
88          print(deciphered_text[i] + "\n")
```

Finalmente, para hacer que el mensaje sea más legible se procedió a eliminar las equis (x) y los signos de igualdad (=), ya que estos estaban estorbando la lectura de la información original y no aportaban a su contenido. Con esto ya se pudo descifrar el cuestionario para el examen y responder las preguntas planteadas.

Parte 3: Resolución de cuestionario

1. ¿CUANDO UN SISTEMA INFORMÁTICO ES CONSIDERADO SEGURO?

Un sistema se considera seguro si mantiene la **integridad** de los datos almacenados y los mantiene **disponibles** para los agentes autorizados. Asimismo se tiene que mantener la **confidencialidad** de los datos, pues no todo el mundo debería tener acceso a esa información.

2. ¿EN QUÉ CONSISTE LA AUTENTICIDAD?

La autenticidad consiste en un mecanismo de verificación, en el cual los usuarios tienen propiedad de sus datos privados. En este contexto los datos deben ser verificados, ya que se necesita determinar la identidad de los usuarios y tener la seguridad de que estos vienen de una fuente confiable.

3. ¿A QUE SE REFIERE EL CONCEPTO DE NO REPUDIO?

El concepto de no repudio asigna responsabilidad a los agentes participantes en la comunicación, tanto de **origen** como **destino,** y busca generar un registro de todas las actividades realizadas.

Con respecto al **origen** no se puede negar el envío de información, ya que el receptor tiene pruebas sobre su envío, y en el caso del **destino,** el receptor no puede negar que no recibió la información, pues también el emisor también tiene esas pruebas desde su lado.

4. MENCIONE DOS DATOS DE LA SEGURIDAD INFORMÁTICA

La seguridad informática es una disciplina que busca proteger un conjunto de activos o recursos de vital importancia, y se enfoca en implementar políticas de seguridad utilizando mecanismos robustos.

5. ¿QUÉ ES UN ATAQUE PASIVO INCLUYA UN EJEMPLO?

Un **ataque pasivo** está relacionado a la información que fluye en la red, violando la privacidad de los usuarios que transitan por ella. Un ejemplo práctico de este ataque es el clasico **man in the middle**, pues mientras dos individuos están comunicándose en la red existe un tercer individuo quien tiene **acceso** a toda esa información. En este caso este último individuo puede divulgar o analizar la información capturada.

6. ¿QUÉ ES UN ATAQUE ACTIVO INCLUYA UN EJEMPLO?

Un **ataque activo** es aquel que causa algún **cambio** en la red o en los datos que fluyen por ella. Un ejemplo de este tipo de ataque es la **negación de servicio**, el cual colapsa todo el tráfico que trata de acceder a este.

7. LA RECOMENDACIÓN DE LA ISO X.800 DIVIDE LOS SERVICIOS DE SEGURIDAD EN CINCO CATEGORÍAS. UNA DE ELLAS SE REFIERE A LA INTEGRIDAD DE LOS DATOS ¿EN QUÉ CONSISTE ESA CATEGORÍA?

Esta categoría se basa en la idea de que los datos deben ser modificados únicamente por sus propietarios y de manera autorizada.

8. MENCIONE TRES MECANISMOS DE SEGURIDAD ESPECÍFICOS Y DESCRÍBALOS BREVEMENTE.

1. **Cifrado:** El cifrado es un mecanismo básico de seguridad, en el cual los datos originales son modificados de tal manera que no puedan ser entendidos por agentes externos en caso de ser recuperados. Es el equivalente a utilizar un mensaje secreto y la manera más simple de lograr la obtención de información es mediante llaves secretas y canales seguros.
2. **Firma digital:** La firma digital es un mecanismo que permite determinar si un individuo dice quien es, y se usa principalmente con fines de autorización de documentos o información sensible. Ejemplos de esto serían las facturas electrónicas o los contratos, los cuales hoy en día muy comúnmente se realizan de manera digital.
3. **Control de rutas:** El control de rutas es un mecanismo de seguridad que resguarda la información para que no todos los usuarios que acceden a un sistema puedan ver lo mismo, o tengan acceso a los mismos datos. Por ejemplo, un empleado común no debería tener acceso a los recursos financieros de la empresa, ni tener la capacidad de modificar la información de otros empleados.

9. ¿PARA QUE SE UTILIZA EL ALGORITMO DE MILLER RABIN?

El algoritmo miller rabin se utiliza para verificar si un número es primo o no, y para esto se recomienda que se haga una cantidad extensiva de iteraciones, para tener un nivel de confiabilidad más alto.

10. ¿POR QUÉ ES IMPORTANTE EL CHINESE REMAINDER THEOREM?

Es importante porque este algoritmo indica que cualquier número puede ser reconstruido como el producto de primos relativos, y esta es una de las bases más importantes en la criptografía.

11. ¿DESCRIBA EL MODELO DE ENCRIPCIÓN SIMÉTRICO?

El modelo de encripción simétrico, es un modelo que se basa la idea de mantener un canal compartido entre dos individuos, el emisor y el receptor, tener dos procesos inversos los cuales se pueden realizar mediante una llave compartida.

En este contexto se requiere un algoritmo fuerte de encripción y tener un canal seguro de comunicación en el cual ambos agentes puedan compartir la llave secreta. Una vez compartida la clave se pueden realizar los procesos inversos sin problema. El emisor encripta el texto utilizando la llave secreta, la comparte y una vez recibido el receptor puede descifrar el mensaje sin problema. Cabe recalcar que en este caso, como es un algoritmo robusto, sería muy difícil descifrar el mensaje sin poseer la llave secreta.

12. ¿CUANDO SE DICE QUE UN SISTEMA DE ENCRIPCIÓN ES INCONDICIONALMENTE SEGURO?

Un sistema de encriptación es incondicionalmente seguro cuando el texto cifrado no proporciona información alguna sobre el texto plano original y aun disponiendo los recursos no es posible descifrar el mensaje.

13. ¿CUÁNDO SE DICE QUE UN ALGORITMO DE ENCRIPCIÓN ES COMPUTACIONALMENTE SEGURO?

Se dice que un sistema de encriptación es computacionalmente seguro si el atacante puede resolver la tarea de desencriptar el mensaje sólo si cuenta con suficiente poder de cálculo y recursos de almacenamiento en su sistema. Sin embargo, en este caso el costo es tan elevado que supera el valor de resolver el mensaje, por lo que no resulta práctico.

14. ¿CUALES SON LAS TÉCNICAS BÁSICAS O BUILDING BLOCKS DE TODA ESTRATEGIA DE ENCRIPCIÓN?

Las técnicas básicas o building blocks de toda estrategia son la **sustitución** y la **transposición** de los caracteres en el texto plano.

15. ESCRIBA POR QUÉ EL GRUPO O GALOIS FIELD ES UN CAMPO FINITO

Un Grupo es un campo finito ya que la cantidad de elementos pertenecientes o la cardinalidad es un valor fijo o finito. El número de elementos en el campo se le conoce como su orden, y un campo solo existe si su orden es una potencia de un número primo p.

16. ¿EN CUANTO A LA FORTALEZA DEL ALGORITMO DES SI UTILIZAMOS UN ALGORITMO FUERZA BRUTA PARA ROMPER LA CONTRASEÑA Y PODEMOS REALIZAR N DESENCRIPCIONES POR SEGUNDO. ¿CUÁNTO TIEMPO NOS DEMORAREMOS EN ENCONTRAR LA CLAVE CON LA QUE FUE ENCRIPTADO EL MENSAJE?

En base a datos referenciales se sabe que un ataque fuerza bruta con 10^13 desencripciones por segundo toma alrededor de una hora en obtener la clave, por lo que N desencripciones por segundo tomaría (10^13/N) horas. Si se pueden hacer 100.000 iteraciones por segundo entonces tomaría alrededor de 1145 años.

Referencias

Anónimo. (Sin fecha). *Improving the quality of the output.* Recuperado el 24 de Octubre del 2020 de https://tesseract-ocr.github.io/tessdoc/ImproveQuality

GeeksforGeeks. (10/08/2020). *Reading images in Python.* Recuperado el 24 de Octubre del 2020 de https://www.geeksforgeeks.org/reading-images-in-python/

Radečić, D. (28/10/2019). *Read Text from Image with One Line of Python Code*. Towards Data Science. Recuperado el 24 de Octubre del 2020 de https://towardsdatascience.com/read-text-from-image-with-one-line-of-python-code-c22ede074cac

SHI, M. (24/06/2020). *Character recognition (OCR) without training model: Use better-preprocessed images in Tesseract without training your own character recognition model.* Towards Data Science. Recuperado el 24 de Octubre del 2020 de https://towardsdatascience.com/character-recognition-without-training-model-d03f213f1aee