

UNIVERSIDAD CENTRAL DEL ECUADOR

Algoritmos Simétricos, Asimétricos y Funciones Hash

Integrantes:

- ❖ Cen Michael
- ❖ Chiguano Kevin
- ❖ Maldonado Ariel
- ❖ Solano Erick



Integrantes: Cen Michael, Chiguano Kevin, Maldonado Ariel, Solano Erick.

Documentación

Todos los algoritmos fueron implementados en Google Colaboratory, un compilador en línea cuyo funcionamiento es similar a Jupyter Notebooks.

El lenguaje de programación utilizado para escribir cada uno de los algoritmos fue Python, y ninguno de los algoritmos fue creado por nosotros, todos fueron instalados y/o importados de librerías que se manejan en el lenguaje.

Cada uno de los algoritmos solicitará que se ingrese la cantidad de palabras que se desea encriptar. Para generar palabras creamos el método “`generar_lorem`”, a este método debemos enviarle un número que será el numero de palabras que generaremos. El método interno “`lorem.words`” nos permite generar texto en forma Lorem Ipsum.

Lorem Ipsum: es un texto ficticio utilizado comúnmente para representar el contenido de un texto cuando el contenido real no está disponible o no es relevante en el contexto en el que se está diseñando.

Algoritmos Simétricos

1. Fernet

Fernet es un sistema de criptografía simétrica diseñado para garantizar la confidencialidad y la integridad de los datos. Se utiliza comúnmente en aplicaciones web y servicios para proteger información sensible, como contraseñas, tokens de autenticación y otros datos confidenciales.

No ahondaremos mucho en detalles pues funciona como cualquier otro algoritmo simétrico, es decir, una misma clave para cifrar y descifrar, la clave es secreta y únicamente se comparte entre los participantes de la comunicación, entre otros datos únicos de la criptografía simétrica.

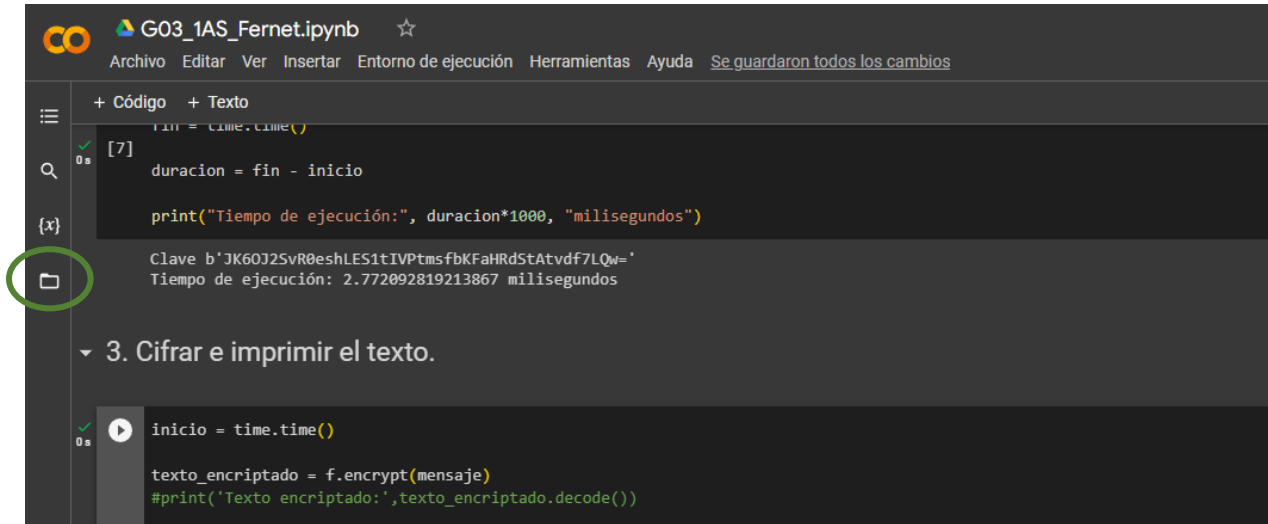
Este algoritmo se encuentra implementado en el archivo “`G03_1AS_Fernet.ipynb`”. Para abrir de forma sencilla el archivo, se recomienda presionar el siguiente botón que se encuentra en la previsualización del archivo en GitHub:



Debido a que es un compilador en línea, no es posible que realicemos consumos excesivos en impresiones de datos en consola, por lo que hemos decidido implementar herramientas que nos permitan generar archivos en donde estará la información encriptada y desencriptada, es por ello que recomendamos abrir los archivos en la herramienta Google Colaboratory.

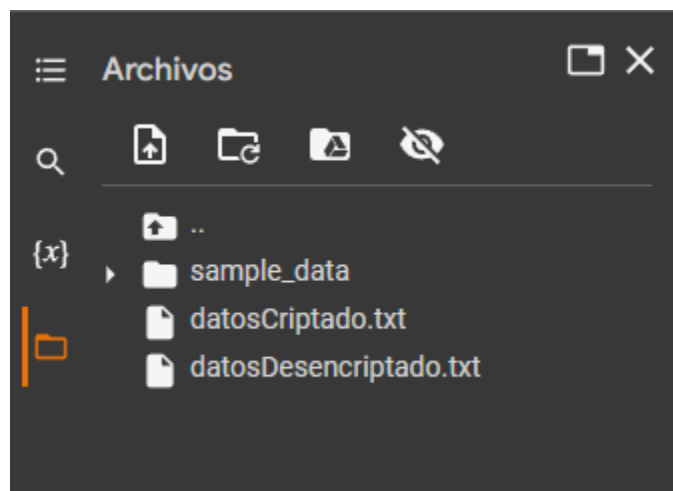
Para acceder a los archivos, una vez ejecutados los distintos bloques del algoritmo, se deberá presionar el siguiente botón que se encuentra a la izquierda de la página web:

Integrantes: Cen Michael, Chiguano Kevin, Maldonado Ariel, Solano Erick.



The screenshot shows a Jupyter Notebook titled "G03_1AS_Fernet.ipynb". The interface includes a menu bar with options like "Archivo", "Editar", "Ver", "Insertar", "Entorno de ejecución", "Herramientas", "Ayuda", and a link "Se guardaron todos los cambios". On the left sidebar, a file icon is circled in green. The main area displays code cells. The first cell contains code for timing an operation: `fin = time.time()`, `duracion = fin - inicio`, and `print("Tiempo de ejecución:", duracion*1000, "milisegundos")`. The output shows a key and the execution time: `Clave b'JK60J2SvR0eshLES1tIVPtmsfbKFaHRdStAtvdf7LQw='` and `Tiempo de ejecución: 2.772092819213867 milisegundos`. The second cell is titled "3. Cifrar e imprimir el texto." and contains code for encryption: `inicio = time.time()`, `texto_encryptado = f.encrypt(mensaje)`, and `#print('Texto encryptado:', texto_encryptado.decode())`.

De esta forma aparecerá lo siguiente:



El archivo "datosCriptado.txt" corresponde a los datos encriptados y el archivo "datosDescriptado.txt", los datos descriptados.

2. ARC4

El algoritmo de cifrado ARC4, también conocido como Alleged RC4, es un algoritmo de cifrado de flujo simétrico que genera un flujo de bytes pseudoaleatorios para cifrar y descifrar datos. Aunque ha sido ampliamente utilizado en el pasado, presenta vulnerabilidades de seguridad y se recomienda utilizar algoritmos más modernos y seguros en aplicaciones actuales.

Este algoritmo se encuentra implementado en el archivo "G03_2AS_ARC4.ipynb". Tal como se mencionó con el algoritmo anterior, se recomienda abrir el archivo por medio del botón "Open in Colab" para facilitar la ejecución del algoritmo, y observar con mayor facilidad los archivos de



Integrantes: Cen Michael, Chiguano Kevin, Maldonado Ariel, Solano Erick.

texto generados al cifrar o descifrar el texto. Es importante destacar que estos archivos se generan de la misma forma en que son generados en el algoritmo anterior.

Algoritmos Asimétricos

3. DSA

DSA (Digital Signature Algorithm) es un algoritmo de firma digital utilizado para garantizar la autenticidad, integridad y no repudio de datos electrónicos. El proceso de firma con DSA implica el cálculo de una función hash del mensaje a firmar y luego aplicar una serie de operaciones matemáticas utilizando la clave privada (generada internamente por el algoritmo, es decir, nosotros no podemos saber cuál es). El resultado de estas operaciones es la firma digital del mensaje, y justamente debido a la aplicación de la función hash, y a que se obtiene una firma digital podemos decir que DSA es un algoritmo de una sola dirección, en otras palabras, es un algoritmo que no descifra datos. Lo que sí puede hacer este algoritmo, y de donde hemos generado los tiempos de la tercera etapa, es verificar la firma.

El algoritmo fue implementado en el archivo “G03_3AA_DSA.ipynb”, y tal como se menciona con los algoritmos anteriores, se recomienda abrirlo en Google Colaboratory por medio del mismo botón. A diferencia de los algoritmos anteriores, como este no firma los datos con una cadena larga de texto, no hemos implementado la creación de documentos externos.

4. HKDF

HKDF (HMAC-based Extract-and-Expand Key Derivation Function) es un algoritmo de derivación de claves basado en HMAC (Hash-based Message Authentication Code). Fue diseñado para ser utilizado en aplicaciones que requieren la generación de claves criptográficas a partir de una clave maestra o de un material de clave inicial. HKDF es flexible y permite adaptarse a diferentes necesidades de aplicaciones. Permite la generación de múltiples claves a partir de una clave maestra, lo que facilita su uso en diferentes contextos criptográficos. Además, permite la incorporación de información contextual adicional, conocida como "información suplementaria", para garantizar la seguridad y la diversificación de las claves generadas.

La decisión de implementar un algoritmo más en nuestra tarea fue debido a que el algoritmo anterior, DSA, no descifra los datos. Este algoritmo fue implementado en el archivo “G03_4AA_HKDF.ipynb”, y se recomienda abrirlo en Google Colaboratory debido a que este algoritmo encripta y desencripta los datos en archivos externos.

Función Hash

5. SHA256

SHA-256 es un algoritmo de función hash criptográfica que toma un mensaje de cualquier longitud y produce un hash de salida fijo de 256 bits. Se utiliza ampliamente en la criptografía para garantizar la integridad de los datos y la autenticación de mensajes en diversas aplicaciones y



Integrantes: Cen Michael, Chiguano Kevin, Maldonado Ariel, Solano Erick.

protocolos. Las funciones hash son de una sola dirección, y por ello no se podrá descryptar los datos que se envíen para generar este hash, sin embargo, se debe destacar que la salida generada en distintas aplicaciones de la función hash a distintos textos difícilmente será la misma.

Esta función fue implementada en el archivo “G03_5FH_SHA256.ipynb”, no cuenta con archivos externos que presenten la información, pero de todos modos se recomienda ejecutarlo en el entorno de Google Colaboratory.

Tabla de Resultados

Columnas de nuestra tabla:

- Algoritmo
- #Palabras, aquí se detalla el número de palabras utilizadas para la aplicación de los algoritmos.
- T-E1 (mili-seg), corresponde al tiempo de la primera etapa “Leer un archivo con el texto del mensaje a cifrar”.
- T-E2 (mili-seg), corresponde al tiempo de la segunda etapa “Generar e imprimir la(las) claves de cifrado y/o descifrado”.
- T-E3 (mili-seg), corresponde al tiempo de la tercera etapa “Cifrar e imprimir el texto”.
- T-E4 (mili-seg), corresponde al tiempo de la cuarta etapa “Descifrar e imprimir el texto”.
- T-Total (mili-seg), corresponde al tiempo total, la suma de los tiempos obtenidos en las etapas anteriores.

UNIVERSIDAD CENTRAL DEL ECUADOR
Facultad de Ingeniería y Ciencias Aplicadas



Integrantes: Cen Michael, Chiguano Kevin, Maldonado Ariel, Solano Erick.

| Algoritmo | #Palabras | T-E1 (mili-seg) | T-E2 (mili-seg) | T-E3 (mili-seg) | T-E4 (mili-seg) | T-Total (mili-seg) |
|-----------|-----------|-----------------|-----------------|-----------------|-----------------|--------------------|
| Fernet | 10 | 0,16093 | 3,02195 | 10,15663 | 2,82239 | 16,1619 |
| | 100 | 0,28133 | 1,51395 | 6,58154 | 2,16746 | 10,54428 |
| | 1000 | 0,86236 | 2,46024 | 8,44430 | 2,21276 | 13,97966 |
| | 10000 | 9,06252 | 3,04961 | 7,61342 | 3,01265 | 22,7382 |
| | 100000 | 6,07393 | 1,05595 | 15,41280 | 10,27941 | 32,822087 |
| | 1000000 | 642,49753 | 1,70564 | 144,41013 | 102,53572 | 891,14902 |
| ARC4 | 10 | 2,78997 | 29,75845 | 4,83155 | 0,66232 | 38,04229 |
| | 100 | 0,23937 | 6,46257 | 2,73609 | 2,78949 | 12,22752 |
| | 1000 | 0,99444 | 5,22685 | 3,26395 | 1,11341 | 10,59865 |
| | 10000 | 9,17601 | 5,03993 | 6,40416 | 2,61259 | 23,23269 |
| | 100000 | 63,54069 | 2,54368 | 28,21850 | 9,31668 | 103,61955 |
| | 1000000 | 498,23474 | 10,07318 | 188,29178 | 85,06155 | 781,66125 |
| DSA | 10 | 0,150919 | 49,920082 | 1,634836 | 0,57435 | 52,280187 |
| | 100 | 0,262022 | 135,853052 | 2,253771 | 1,970291 | 140,339136 |
| | 1000 | 0,941992 | 113,153219 | 62,07466 | 3,122091 | 179,291962 |
| | 10000 | 7,983446 | 112,848043 | 2,421379 | 10,992765 | 134,245633 |
| | 100000 | 267,023802 | 127,464771 | 15,727997 | 19,722223 | 429,938793 |
| | 1000000 | 476,813078 | 80,542564 | 26,770592 | 30,819893 | 614,946127 |
| HKDF | 10 | 0,22387 | 19,35124 | 5,44190 | 1,77645 | 26,79346 |
| | 100 | 0,23865 | 30,13324 | 6,39486 | 3,93986 | 40,70661 |
| | 1000 | 0,82516 | 21,20923 | 6,32309 | 2,27713 | 30,63461 |
| | 10000 | 23,62322 | 19,79589 | 7,50136 | 2,70271 | 53,62318 |
| | 100000 | 120,80574 | 17,18449 | 15,97595 | 8,18538 | 162,15156 |
| | 1000000 | 510,42079 | 11,15822 | 58,82859 | 57,66725 | 638,07485 |
| Sha256 | 10 | 0,16164 | 1,62935 | 0,21362 | | 2,00461 |
| | 100 | 0,24223 | 0,22101 | 0,38623 | | 0,84947 |
| | 1000 | 1,08361 | 6,49523 | 0,34117 | | 7,92001 |
| | 10000 | 11,48080 | 0,29635 | 0,75483 | | 12,53198 |
| | 100000 | 52,59513 | 0,25010 | 6,63638 | | 59,48161 |
| | 1000000 | 443,82762 | 0,99396 | 37,72592 | | 482,5475 |

El color rojo significa que no se obtuvo ningún tiempo en esa etapa.