Informe Laboratorio 4

Sección 2

Cristóbal Barra
cristobal.barra1@mail.udp.cl

Junio de 2024

Índice

1. Descripción de actividades

Para este laboratorio, deberá utilizar Tampermonkey y la librería CryptoJS (con SRI) para lograr obtener los mensajes que le está comunicando su informante. En esta ocasión, su informante fue más osado y se comunicó con usted a través de un sitio web abierto a todo el público https://cripto.tiiny.site/.

Sólo un ojo entrenado como el suyo logrará descifrar cuál es el algoritmo de cifrado utilizado y cuál es la contraseña utilizada para lograr obtener la información que está oculta.

1. Desarrolle un plugin para tampermonkey que permita obtener la llave para el descifrado de los mensajes ocultos en la página web. La llave debe ser impresa por la consola de su navegador al momento de cargar el sitio web. Utilizar la siguiente estructura:

   - La llave es: KEY

2. En el mismo plugin, se debe detectar el patrón que permite identificar la cantidad de mensajes cifrados. Debe imprimir por la consola la cantidad de mensajes cifrados. Utilizar la siguiente estructura: Los mensajes cifrados son: NUMBER

3. En el mismo plugin debe obtener cada mensaje cifrado y descifrarlo. Ambos mensajes deben ser informados por la consola (cifrado espacio descifrado) y además cada mensaje en texto plano debe ser impreso en la página web.

   El script desarrollado debe ser capaz de obtener toda la información del sitio web (llave, cantidad de mensajes, mensajes cifrados) sin ningún valor forzado. Para verificar el correcto funcionamiento de su script se utilizará un sitio web con otro texto y una cantidad distinta de mensajes cifrados. Deberá indicar la url donde se podrá descargar su script.

   Un ejemplo de lo que se debe visualizar en la consola, al ejecutar automáticamente el script, es lo siguiente:

2. Desarrollo (Parte 1)

2.1. Detecta el cifrado utilizado por el informante

Lo primero que se nota al ingresar en la página web es un parráfo grande de texto, éste está escrito de tal manera que lleva una llave de descifrado escondida dentro, pero que solamente analizando el texto de manera detenida se podrá encontrar.

Sin el conocimiento de informaciÃ³n secreta, el criptoanÃ¡lisis se dedica al estudio de sistemas criptogrÃ¡ficos con el fin de encontrar debilidades en los sistemas y romper su seguridad. El criptoanÃ¡lisis es un componente importante del proceso de creaciÃ³n de criptosistemas sÃ³lidos. Gracias al criptoanÃ¡lisis, podemos comprender los criptosistemas y mejorarlos identificando los puntos dÃ©biles. Un criptoanalista puede ayudarnos a trabajar en el algoritmo para crear un cÃ³digo secreto mÃ¡s seguro y protegido. Resultado del criptoanÃ¡lisis es la protecciÃ³n de la informaciÃ³n crÃtica para que no sea interceptada, copiada, modificada o eliminada. Otras tareas de las que pueden ser responsables los criptoanalistas incluyen evaluar, analizar y localizar las debilidades en los sistemas y algoritmos de seguridad criptogrÃ¡fica. Sin el conocimiento de informaciÃ³n secreta, el criptoanÃ¡lisis se dedica al estudio de sistemas criptogrÃ¡ficos con el fin de encontrar debilidades en los sistemas y romper su seguridad. El criptoanÃ¡lisis es un componente importante del proceso de creaciÃ³n de criptosistemas sÃ³lidos. Gracias al criptoanÃ¡lisis, podemos comprender los criptosistemas y mejorarlos identificando los puntos dÃ©biles. Un criptoanalista puede ayudarnos a trabajar en el algoritmo para crear un cÃ³digo secreto mÃ¡s seguro y protegido. Resultado del criptoanÃ¡lisis es la protecciÃ³n de la informaciÃ³n crÃtica para que no sea interceptada, copiada, modificada o eliminada. Otras tareas de las que pueden ser responsables los criptoanalistas incluyen evaluar, analizar y localizar las debilidades en los sistemas y algoritmos de seguridad criptogrÃ¡fica. Sin el conocimiento de informaciÃ³n secreta, el criptoanÃ¡lisis se dedica al estudio de sistemas criptogrÃ¡ficos con el fin de encontrar debilidades en los sistemas y romper su seguridad. El criptoanÃ¡lisis es un componente importante del proceso de creaciÃ³n de criptosistemas sÃ³lidos. Gracias al criptoanÃ¡lisis, podemos comprender los criptosistemas y mejorarlos identificando los puntos dÃ©biles. Un criptoanalista puede ayudarnos a trabajar en el algoritmo para crear un cÃ³digo secreto mÃ¡s seguro y protegido. Resultado del criptoanÃ¡lisis es la protecciÃ³n de la informaciÃ³n crÃtica para que no sea interceptada, copiada, modificada o eliminada. Otras tareas de las que pueden ser responsables los criptoanalistas incluyen evaluar, analizar y localizar las debilidades en los sistemas y algoritmos de seguridad criptogrÃ¡fica. Sin el conocimiento de informaciÃ³n secreta, el criptoanÃ¡lisis se dedica al estudio de sistemas criptogrÃ¡ficos con el fin de encontrar debilidades en los sistemas y romper su seguridad. El criptoanÃ¡lisis es un componente importante del proceso de creaciÃ³n de criptosistemas sÃ³lidos. Gracias al criptoanÃ¡lisis, podemos comprender los criptosistemas y mejorarlos identificando los puntos dÃ©biles. Un criptoanalista puede ayudarnos a trabajar en el algoritmo para crear un cÃ³digo secreto mÃ¡s seguro y protegido. Resultado del criptoanÃ¡lisis es la protecciÃ³n de la informaciÃ³n crÃtica para que no sea interceptada, copiada, modificada o eliminada. Otras tareas de las que pueden ser responsables los criptoanalistas incluyen evaluar, analizar y localizar las debilidades en los sistemas y algoritmos de seguridad criptogrÃ¡fica.

Figura 1: Texto página web.

El texto mostrado en la figura 1 está escrito de una manera extraña, cada oración empieza de tal manera que se puede ver un patrón que se repite cada seis oraciones. Es de esta manera que si se extraen las mayúsculas del texto se puede obtener la llave. Esto es mas una corazonada que cualquier otra cosa, por lo tanto, no se tiene total certeza de que esa sea la llave, pero por algo se debe empezar. La potencial llave es **SEGUROSEGUROSEGUROSEGURO**.

2.2. Logra que el script solo se gatille en el sitio usado por el informante

Para lograr que el script solo se gatille en la página web que se desea, se debe pegar su URL (https://cripto.tiiny.site/) en la sección de UserScript *@match*. Tal como se verá en la figura 2 a continuación.

```
// ==UserScript==
// @name         Lab4 | Cristobal Barra
// @namespace    http://tampermonkey.net/
// @version      2024-06-04
// @description  try to take over the world!
// @author       You
// @match        https://cripto.tiiny.site/
```

Figura 2: Establecimiento del script para la página web.

2.3. Define función que obtiene automáticamente el password del documento

Ahora se procede a escribir el script para obtener la llave, script el cual extraerá todas las letras mayúsculas del texto en cuestión. La función accede al documento HTML, y en la

sección body busca todas las mayúsculas y las extrae para colocarlar en un array de caracteres. Luego, este array se imprime a través de la consola de la página web, y por último se guarda la variable para utilizarse más adelante. El script se podrá apreciar en la figura 3 más abajo.

```
// Función para obtener llave
function getKey() {
    // Extraer todas las letras mayúsculas del contenido de la página
    let uppercaseLetters = document.body.innerText.match(/[A-Z]/g) || [];
    let key = uppercaseLetters.join('');
    console.log(`La llave es: ${key}`);
    return key;
}

// Ejecutar función de obtención de llave
const key = getKey();
```

Figura 3: Función de obtención de llave.

2.4. Muestra la llave por consola

Tal como se muestra en la figura 4, la llave resultó ser la que se esperaba. La llave encontrada se utilizará más adelante para descifrar los mensajes.

```
La llave es: SEGUROSEGUROSEGUROSEGURO                    Lab4-|-Cristobal-Barra.user.js:22:17
```

Figura 4: Impresión de llave por consola.

3. Desarrollo (Parte 2)

3.1. Reconoce automáticamente la cantidad de mensajes cifrados

Lo siguiente es reconocer la cantidad de mensajes cifrados que contiene esta página web, para ello será necesario entrar en el modo de *Inspección de elemento*. Dentro de este modo se notará que hay mensajes cifrados según la figura 5, éstos tienen un *id* y una *clase*, y cada mensaje tiene por clase la letra "M"seguido de un número, por lo que se puede usar esta característica dentro del script para contar los mensajes cifrados.

```
<div class="M1" id="xEopI5pGBCQ="> </div>
<div class="M2" id="vGECWVrA2f4="> </div>
<div class="M3" id="MBDQE5ZiDsQ="> </div>
<div class="M4" id="c8zxLt/4Iuk="> </div>
<div class="M5" id="tPciMWg0pmg="> </div>
<div class="M6" id="/XZw4C/lGEk="> </div>
```

Figura 5: Modo de inspección de elemento.

El script entra en el documento HTML y busca todos los elementos con la query especifíca que se muestra en la figura 6 a continuación, llevando la cuenta de los mensajes que tengan esa misma query en su descripción a través del núumero de elementos que fueron seleccionados. Luego se imprime el resultado por consola y se guarda la variable.

```
// Función para contar mensajes cifrados
function countEncryptedMessages() {
    let encryptedElements = document.querySelectorAll('[class^="M"]');
    let count = encryptedElements.length;
    console.log(`Los mensajes cifrados son: ${count}`);
    return count;
}

// Ejecutar funciñón de conteo de mensajes
const encryptedCount = countEncryptedMessages();
```

Figura 6: Función de conteo de mensajes cifrados.

3.2. Muestra la cantidad de mensajes por consola

Luego de ejecutar la función, el resultado se imprime en la consola, mostrando que hay 6 mensajes cifrados. El resultado se puede ver en la figura 7

```
Los mensajes cifrados son: 6                    Lab4-|-Cristobal-Barra.user.js:33:17
```

Figura 7: Impresión de conteo de mensajes cifrados por consola.

4. Desarrollo (Parte 3)

4.1. Importa la librería cryptoJS

Para importar la librería CryptoJS se debe acceder a la siguiente página: cdnjs.com/libraries/crypto-js. En ésta se encuentra la URL de la librería que se utilizará para descifrar los mensajes, acompañado de su respectivo SRI.

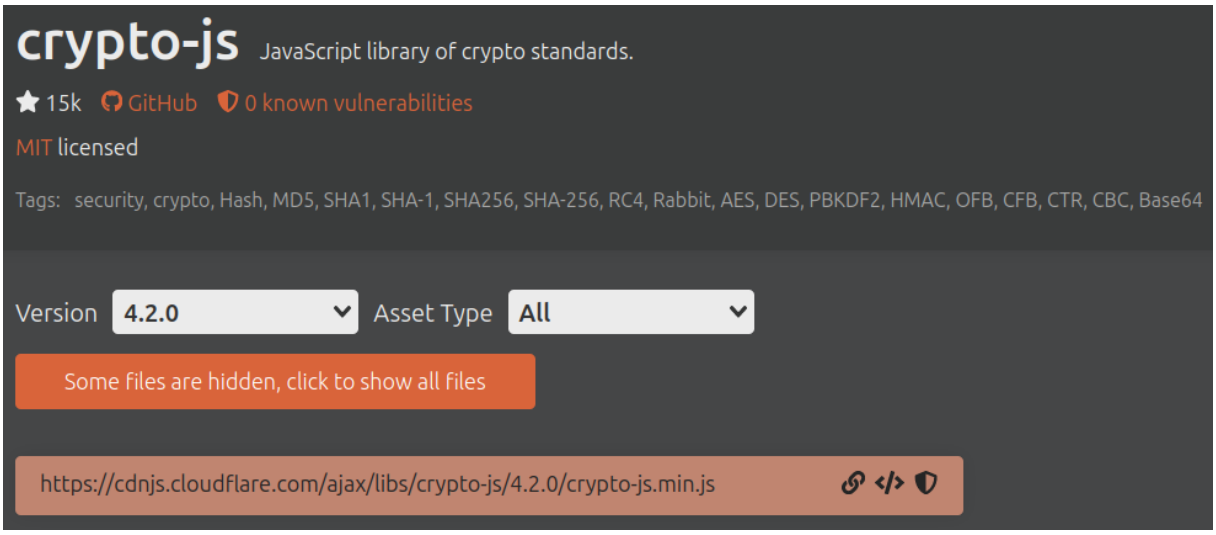

Figura 8: Página cdnjs.com

Para importar dentro del script, solo se debe copiar la URL que se muestra en la parte baja de la figura 8, y pegar en la sección UserScript *@require* tal como se muestra en la figura 9 a continuación.

```
// ==UserScript==
// @name         Lab4 | Cristobal Barra
// @namespace    http://tampermonkey.net/
// @version      2024-06-04
// @description  try to take over the world!
// @author       You
// @match        https://cripto.tiiny.site/
// @icon         https://www.google.com/s2/favicons?sz=64&domain=tiiny.site
// @grant        none
// @require      https://cdnjs.cloudflare.com/ajax/libs/crypto-js/4.2.0/crypto-js.min.js
// ==/UserScript==
```

Figura 9: Importación de librería CryptoJS.

4.2. Utiliza SRI en la librería CryptoJS

El SRI es una característica de seguridad que permite a los desarrolladores asegurarse de que sus recursos (scripts, hojas de estilos, documentos, etc.) no han sido alterados por terceros al momento de cargar la página web. Resulta útil al momento de cargar recursos desde CDNs (Content Delivery Network), que es justamente lo que se está haciendo aquí, proporcionando una capa adicional de seguridad al sistema.
En este caso, el SRI proporcionado asegura que la librería CryptoJS que se está cargando desde un servidor externo no haya sido alterada desde su origen, al colocar el SRI después del URL de la librería, se está asegurando que la librería que se está utilizando sea exactamente la misma que se espera que sea.

En la figura 10 se muestra como se implementa el SRI dentro del código, justamente después del URL de la librería CryptoJS. EL SRI se obtiene de la página que se muestra en la figura 8, haciendo click en ícono de escudo, se copía el texto que hace referencia al SRI.

```
// ==UserScript==
// @name         Lab4 | Cristobal Barra
// @namespace    http://tampermonkey.net/
// @version      2024-06-04
// @description  try to take over the world!
// @author       You
// @match        https://cripto.tiiny.site/
// @icon         https://www.google.com/s2/favicons?sz=64&domain=tiiny.site
// @grant        none
// @require      https://cdnjs.cloudflare.com/ajax/libs/crypto-js/4.2.0/crypto-js.min.js#sha512-a+SUDuwNzXDvz4XrIcXHuCf089/iJAoN4lmrXJg18XnduKK6YlDHNRalv4yd1N40OKI80tFidF+rqTFKGPoWFQ==
// ==/UserScript==
```

Figura 10: SRI.

4.3. Repercusiones de SRI inválido

Para este caso, se inhabilita la librería CryptoJS, ya que ésta necesita de un SRI específico para funcionar, no funcionará con otra SRI que no sea la que se copió en la figura 8, por lo tanto tampoco podrá hacerse uso de ésta en el script. En la figura 11 se puede apreciar la advertencia que lanza la consola al momento de ejecutar el script.

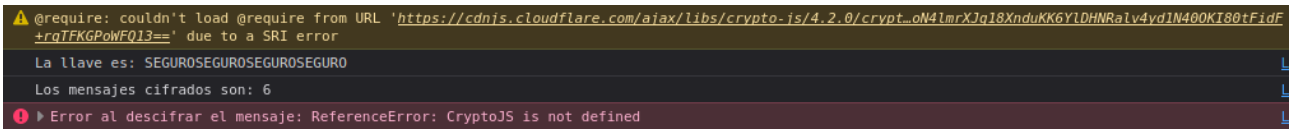

Figura 11: Advertencias SRI inválido.

Pero pueden existir casos donde las páginas web carguen los datos sin la verificación del SRI, por lo que pueden volverse vulnerables al cargar recursos de fuentes externas de dudosa procedencia.

4.4. Logra decifrar uno de los mensajes

Esta función recibe como parámetro la llave anteriormente obtenida, guarda todos los mensajes cifrados y luego itera sobre cada uno. Decodifica el texto utilizando Base64 y convierte la llave en bytes para posterior realizar el descifrado, establece las configuraciones de descifrado. Estas opciones definen el modo de operación y el esquema de padding que se utilizará. Utilizando Triple DES para descifrar, convierte el texto descifrado en un string y lo imprime en la consola, también actualiza el contenido que hay dentro del documento HTML para que lo pueda mostrar en la página web. Esta descripción del código se ve acompañada por la captura de pantalla del mismo, en la figura 12.

```
// Función para descifrar usando CryptoJS
function decryptMessages(key) {
    let encryptedElements = document.querySelectorAll('[class^="M"]');

    encryptedElements.forEach(function(element) {
        let encryptedTextBase64 = element.getAttribute('id');

        try {
            // Decodificar el texto cifrado desde Base64
            let encryptedText = CryptoJS.enc.Base64.parse(encryptedTextBase64);

            // Convertir la clave a bytes
            let keyBytes = CryptoJS.enc.Utf8.parse(key);

            // Configurar las opciones de descifrado
            const options = { mode: CryptoJS.mode.ECB, padding: CryptoJS.pad.Pkcs7 };

            // Descifrar el texto cifrado
            const decrypted = CryptoJS.TripleDES.decrypt(
                { ciphertext: encryptedText },
                keyBytes,
                options
            );

            let decryptedText = decrypted.toString(CryptoJS.enc.Utf8);

            if (decryptedText) {
                console.log(`${encryptedTextBase64} ${decryptedText}`);
                // Actualizar el contenido del elemento con el texto descifrado
                element.textContent = decryptedText;
            } else {
                console.log(`No se pudo descifrar el texto de ${element.className}.`);
            }
        } catch (e) {
            console.error("Error al descifrar el mensaje:", e);
        }
    });
}

// Ejecutar función de descifrado de mensajes
decryptMessages(key);
```

Figura 12: Función de descifrado de mensajes.

Al cargar la página, se ejecuta automáticamente la función de descifrado, mostrando por consola los mensajes desencriptados. En la figura 13 se haya un ejemplo de un mensaje que ha sido descifrado acompañado del mismo mensaje cifrado.

```
xEopI5pGBCQ= este                                    Lab4-|-Cristobal-Barra.user.js:67:29
```

Figura 13: Descifrado de mensaje.

4.5. Imprime todos los mensajes por consola

El código de la figura 12 imprime todos los mensajes descifrados que se han encontrado, a través de la consola se aprecian los seis mensajes que se encontraron anteriormente.

```
xEopI5pGBCQ= este            Lab4-|-Cristobal-Barra.user.js:67:29
vGECWVrA2f4= es              Lab4-|-Cristobal-Barra.user.js:67:29
MBDQE5ZiDsQ= un              Lab4-|-Cristobal-Barra.user.js:67:29
c8zxLt/4Iuk= mensaje         Lab4-|-Cristobal-Barra.user.js:67:29
tPciMWg0pmg= de              Lab4-|-Cristobal-Barra.user.js:67:29
/XZw4C/lGEk= prueba          Lab4-|-Cristobal-Barra.user.js:67:29
```

Figura 14: Impresión de mensajes descifrados por consola.

4.6. Muestra los mensajes en texto plano en el sitio web

El mismo código de la figura 12 actualiza el contenido que hay en el HTML, logrando que también se muestren los mensajes descifrados en la página web, justo abajo del texto para encontrar la llave. Dichos mensajes se aprecian en la figura 15.

Sin el conocimiento de informaciÃ³n secreta, el criptoanÃ¡lisis se dedica al estudio de sistemas criptogrÃ¡ficos con el fin de encontrar debilidades en los sistemas y romper su seguridad. El criptoanÃ¡lisis es un componente importante del proceso de creaciÃ³n de criptosistemas sÃ³lidos. Gracias al criptoanÃ¡lisis, podemos comprender los criptosistemas y mejorarlos identificando los puntos dÃ©biles. Un criptoanalista puede ayudarnos a trabajar en el algoritmo para crear un cÃ³digo secreto mÃ¡s seguro y protegido. Resultado del criptoanÃ¡lisis es la protecciÃ³n de la informaciÃ³n crÃtica para que no sea interceptada, copiada, modificada o eliminada. Otras tareas de las que pueden ser responsables los criptoanalistas incluyen evaluar, analizar y localizar las debilidades en los sistemas y algoritmos de seguridad criptogrÃ¡fica. Sin el conocimiento de informaciÃ³n secreta, el criptoanÃ¡lisis se dedica al estudio de sistemas criptogrÃ¡ficos con el fin de encontrar debilidades en los sistemas y romper su seguridad. El criptoanÃ¡lisis es un componente importante del proceso de creaciÃ³n de criptosistemas sÃ³lidos. Gracias al criptoanÃ¡lisis, podemos comprender los criptosistemas y mejorarlos identificando los puntos dÃ©biles. Un criptoanalista puede ayudarnos a trabajar en el algoritmo para crear un cÃ³digo secreto mÃ¡s seguro y protegido. Resultado del criptoanÃ¡lisis es la protecciÃ³n de la informaciÃ³n crÃtica para que no sea interceptada, copiada, modificada o eliminada. Otras tareas de las que pueden ser responsables los criptoanalistas incluyen evaluar, analizar y localizar las debilidades en los sistemas y algoritmos de seguridad criptogrÃ¡fica. Sin el conocimiento de informaciÃ³n secreta, el criptoanÃ¡lisis se dedica al estudio de sistemas criptogrÃ¡ficos con el fin de encontrar debilidades en los sistemas y romper su seguridad. El criptoanÃ¡lisis es un componente importante del proceso de creaciÃ³n de criptosistemas sÃ³lidos. Gracias al criptoanÃ¡lisis, podemos comprender los criptosistemas y mejorarlos identificando los puntos dÃ©biles. Un criptoanalista puede ayudarnos a trabajar en el algoritmo para crear un cÃ³digo secreto mÃ¡s seguro y protegido. Resultado del criptoanÃ¡lisis es la protecciÃ³n de la informaciÃ³n crÃtica para que no sea interceptada, copiada, modificada o eliminada. Otras tareas de las que pueden ser responsables los criptoanalistas incluyen evaluar, analizar y localizar las debilidades en los sistemas y algoritmos de seguridad criptogrÃ¡fica. Sin el conocimiento de informaciÃ³n secreta, el criptoanÃ¡lisis se dedica al estudio de sistemas criptogrÃ¡ficos con el fin de encontrar debilidades en los sistemas y romper su seguridad. El criptoanÃ¡lisis es un componente importante del proceso de creaciÃ³n de criptosistemas sÃ³lidos. Gracias al criptoanÃ¡lisis, podemos comprender los criptosistemas y mejorarlos identificando los puntos dÃ©biles. Un criptoanalista puede ayudarnos a trabajar en el algoritmo para crear un cÃ³digo secreto mÃ¡s seguro y protegido. Resultado del criptoanÃ¡lisis es la protecciÃ³n de la informaciÃ³n crÃtica para que no sea interceptada, copiada, modificada o eliminada. Otras tareas de las que pueden ser responsables los criptoanalistas incluyen evaluar, analizar y localizar las debilidades en los sistemas y algoritmos de seguridad criptogrÃ¡fica.

este
es
un
mensaje
de
prueba

Figura 15: Mensajes en texto plano en el sitio web.

4.7. El script logra funcionar con otro texto y otra cantidad de mensajes

El script en sí, demuestra que no tiene valores forzados al momento de ejecutar sus funciones, por lo cual debería funcionar correctamente en cualquier otro texto y con diferentes cantidades de mensajes cifrados (siempre y cuando estos mensajes lleven por **class="M#"**

para que puedan ser identificados como mensajes cifrados). Este es un algoritmo que detecta la llave en base al patrón definido, como también la cantidad de mensajes cifrados.

4.8. Indica url al código .js implementado para su validación

El archivo del script se puede encontrar tanto en el Github, dentro de la carpeta **Script Tampermonkey**. Como también se encuentra contenido dentro del Drive con el link que se encuentra a continuación, éste debe ser abierto con una cuenta institucional.
https://drive.google.com/file/d/12JNx8a4pzEkwg7_RaWNf_VamrSyBi3tb/view?usp=sharing

Conclusiones y comentarios

Para terminar la experiencia, el script desarrollado durante esta actividad demuestra ser una solución efectiva para el descrifrado de mensajes ocultos en páginas web, junto al uso de la librería CryptoJS y el análisis de parte de la propia persona para encontrar patrones en el texto. Las tres fases, obtención de llave, conteo de mensajes cifrados y y descifrado de éstos, permiten que este código se encuentre en completitud en base a las diversas labores de desencriptado.
La ejecución del script se muestra por partes y con ejemplos visuales que permiten apreciar como funciona este mismo. Demostrando como es que opera el script al momento de identificar automáticamente parámetros importantes para el correcto funcionando. Se valida también que el script no tiene valores forzados, por lo que puede trabajar sin problemas en cualquier otra página web con características similares, esta flexibilidad destaca la robustez del algoritmo y su precisa implementación. El código fuente de este script se encuentra tanto en el Github como en el URL del Drive presentado con anterioridad, para que pueda ser puesto a prueba en otras páginas web.

Issues

Uno de los primeros issues que se hallaron en esta actividad, fue al momento de entrar en la página web por primera vez y encontrar este texto que parece no tener sentido en un principio. Pero hay que detenerse a analizar el texto, leer y usar un poco la imaginación de cómo se pudo haber guardado la llave dentro de este extenso párrafo, para ello hubo que encontrar patrones que tuvieran sentido al momento de identificar la llave, esto se pudo lograr y se prosiguió con la actividad.
La detección de los mensajes cifrados para su conteo no fue sencilla, ya que estos mensajes no se encuentran impresos en la página web, lo que dificultó en primera instancia el proceso de conteo de mensajes. Hay que ahondar más en la página web y entrar en el modo de inspección de elemento, aquí es donde se encuentra el HTML de la página y es donde también se encuentran los mensajes ocultos y encriptados, luego de esto, se busca el patrón para automatizar el conteo a través del script de Tampermonkey.
La implementación del SRI para la librería CryptoJS parece simple al solo tener que copiar el

hash a continuación del URL de la biblioteca, pero en verdad no es algo tan obvio, ya que no hay documentación de la misma página de donde se obtiene la librería de cómo implementar el SRI. Es por esto que se accede a la documentación de Tampermonkey, donde se encuentra cómo implementar el SRI a la librería, documentación la cual se encuentra en el siguiente link: https://www.tampermonkey.net/documentation.php

El descifrado de los mensajes vendría a ser lo más complicado del laboratorio, ya que requieren de diferentes métodos y de librerías especiales (CryptoJS). Para que el código pueda ser robusto y automatizado, requiere que se identifiquen y se extraigan patrones, lo contrario a forzar valores para descifrar los mensajes, lo que significa un esfuerzo extra al momento de codear. Aplicando dentro de éste querys específicas y validando que cada mensaje pueda ser descifrado de manera correcta.