Informe Laboratorio 4

Sección 3

Alan Toro
e-mail: alan.toro@mail.udp.cl

Noviembre de 2023

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

Se conoce de antemano la llave como “SEGUROSEGUROSEGUROSEGURO”, Por inspección simple se puede notar que las letras son todas mayúsculas y que además, son las únicas mayúsculas dentro del documento.

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

Para gatillar el script se puede usar la directiva `@match` con la URL a la cuál se espera usar para gatillar el script.

```
@match          https://cripto.tiiny.site/
```

2.3. Define función que obtiene automáticamente el password del documento

Para buscar el mensaje dentro del texto, primero se selecciona la etiqueta "p", se aplica una expresión regular buscando letras mayúsculas lo que retorna un arreglo de los caractéres en mayúsculas que se encuentran y se le aplica join() para transformar de array a string.

```
1 let key = document.querySelector('p').textContent.match(/[A-Z]/g).
    join("")
2 console.log("La llave es: ", key)
```

2.4. Muestra la llave por consola

Al ejecutarse el script muestra por consola la llave encontrada con el formato "La llave es: KEY".

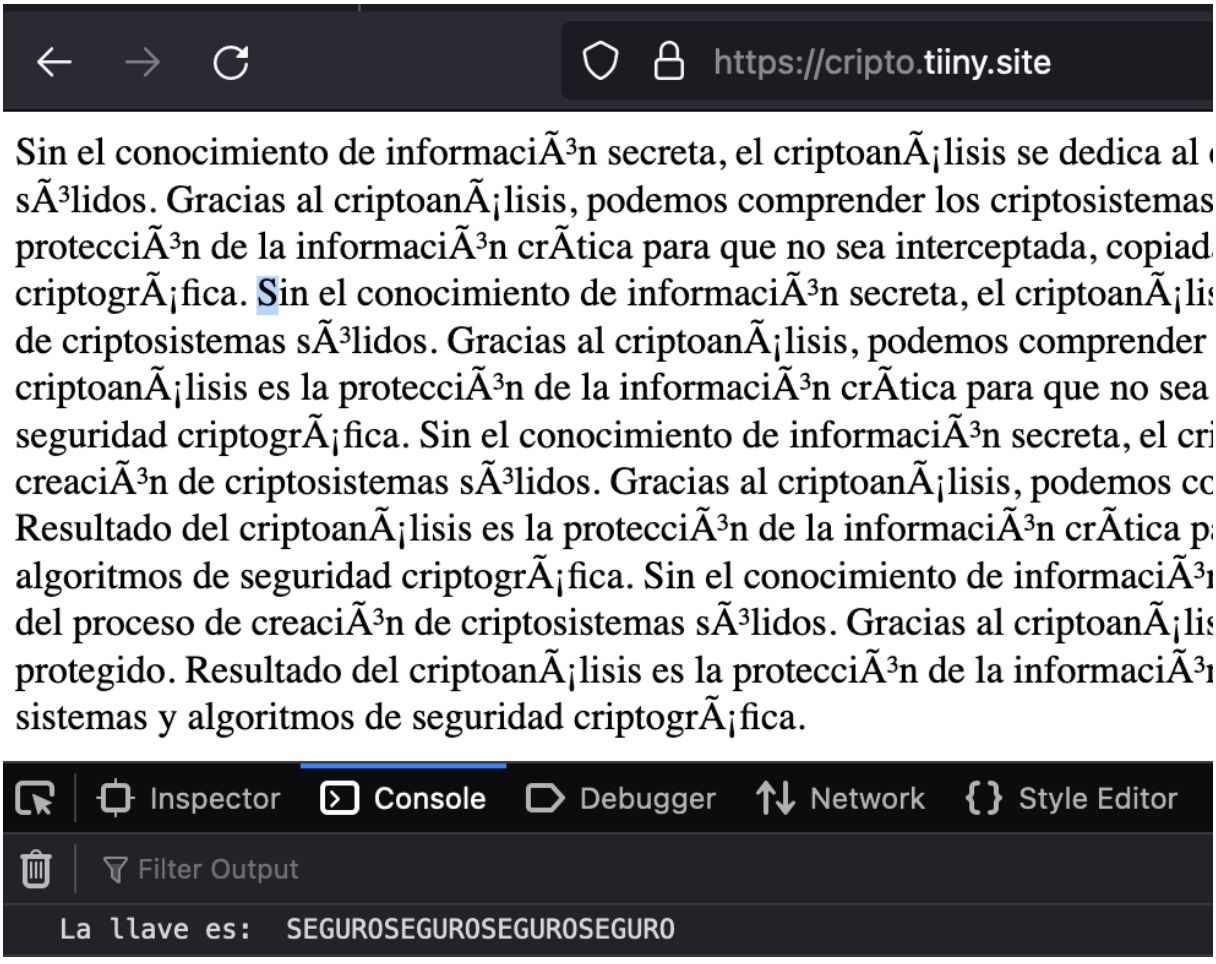

Figura 1: Ejecución del script y llave mostrada en consola.

3. Desarrollo (Parte 2)

3.1. Reconoce automáticamente la cantidad de mensajes cifrados

Además de la etiqueta "p"se encuentran etiquetas "div"sin contenido en donde el id terminar en "=" por lo que se puede intuir que son mensajes en base64. Se genera entonces un script que cuente las etiquetas "div.[en] el documento.

```
let msgs = document.querySelectorAll('div')
console.log("Los mensajes cifrados son: ", msgs.length)
```

3.2. Muestra la cantidad de mensajes por consola

Se muestran entonces la cantidad de mensajes cifrados dentro de la página.

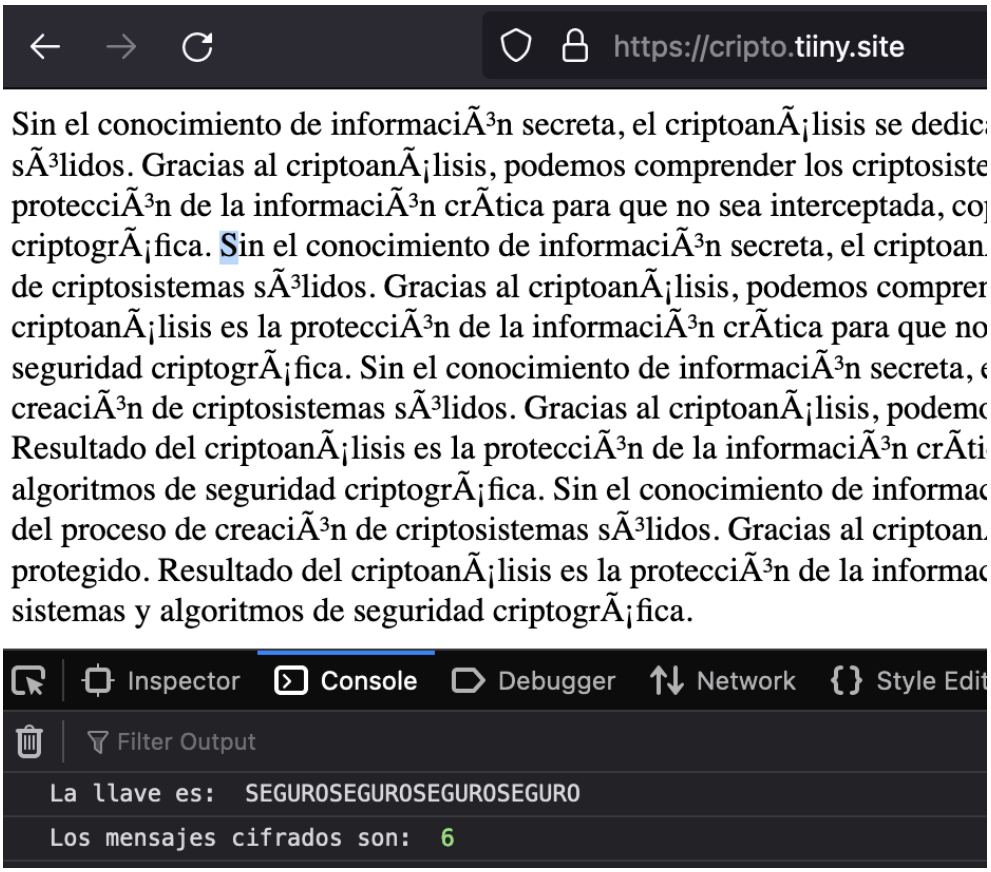

Figura 2: Cantidad de mensajes cifrados.

4. Desarrollo (Parte 3)

4.1. Importa la librería cryptoJS

Para importar librerías con Tampermonkey se utiliza la directiva `@require`, con esta basta con entregarle la URL de la librería para tener disponible la librería.

4.2. Utiliza SRI en la librería CryptoJS

Además, la directiva `@require` permite de forma nativa utilizar una url en el formato url#hash para el SRI, en donde el hash puede estar en MD5 o SHA-512 (que es el utilizado para la revisión de integridad en cdnjs que es el utilizado en este informe).

Para visualizar el uso de la directiva se puede ver en el script adjunto a este trabajo.

4.3. Logra decifrar uno de los mensajes

De la imagen entregada en el enunciado se sabe que el método para decifrar debe ser 3DES. Para esto, la librería CryptoJS exige recibir el texto encriptado y la llave ambos en base64, además de definir el método de padding y el modo.

El mensaje cifrado ya se encuentra en base64 por lo que basta con parsearlo. Para la llave (que se encuentra en texto plano) es necesario transformarla a base64 utilizando

```
1 let keyBase64 = CryptoJS.enc.Base64.stringify(CryptoJS.enc.Utf8.
    parse(key))
```

Para luego parsearla de la misma manera que el mensaje. Esto se realiza usando

```
1 CryptoJS.enc.Base64.parse(<textToParse>)
```

Con lo anterior se puede decifrar el mensaje usando:

```
1 CryptoJS.TripleDES.decrypt(msg.id, parsedKey, {mode: CryptoJS.mode.
    ECB, padding: CryptoJS.pad.Pkcs7})
```

4.4. Imprime todos los mensajes por consola

Se repite el proceso por cada uno de los mensajes guardados por cada uno de los mensajes cifrados.

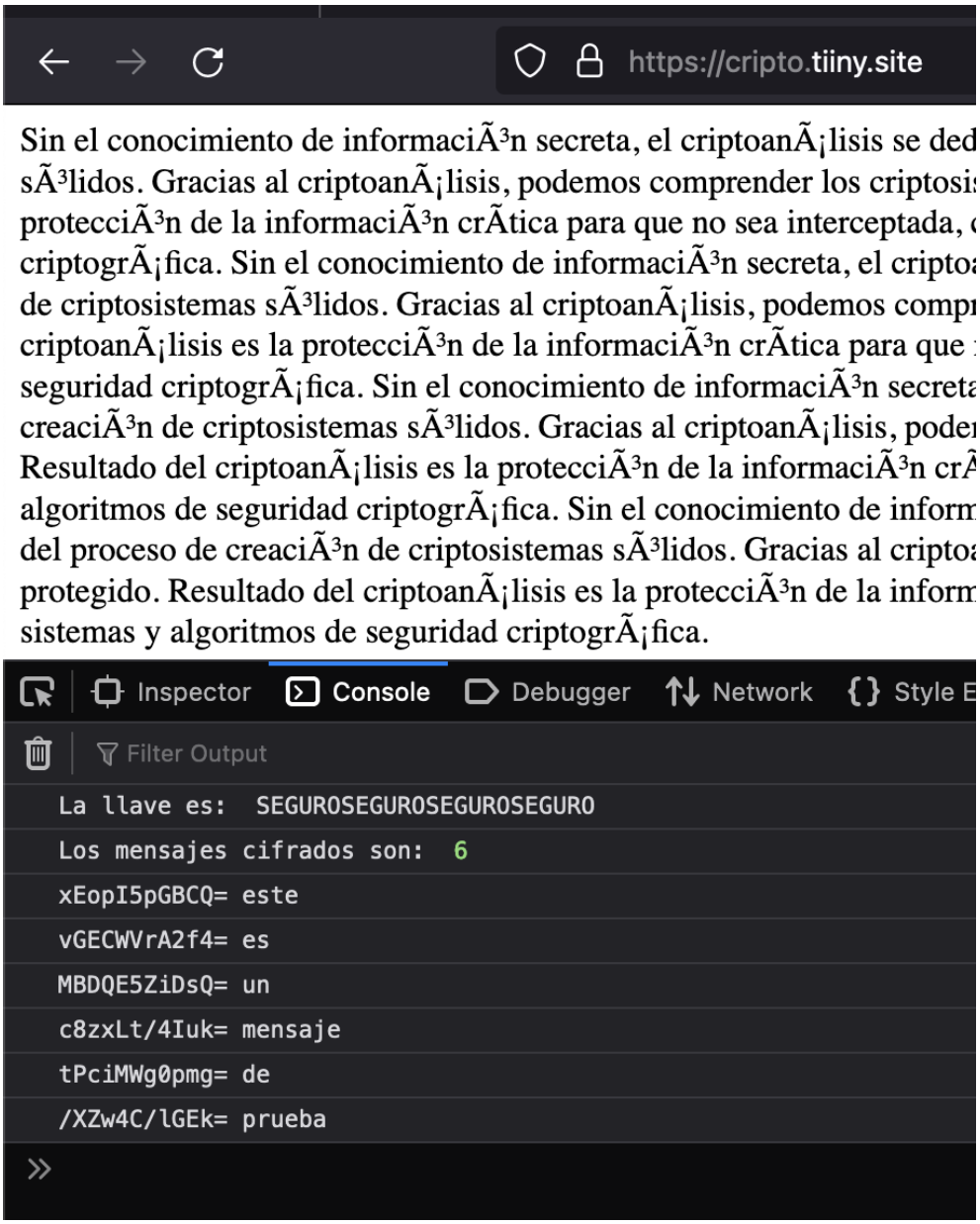

Figura 3: Mensajes mostrados en la consola en texto plano.

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

Finalmente, los mensajes cifrados además de mostrarse en consola se agregan a la etiqueta "pçada uno de los mensajes decifrados en texto plano.

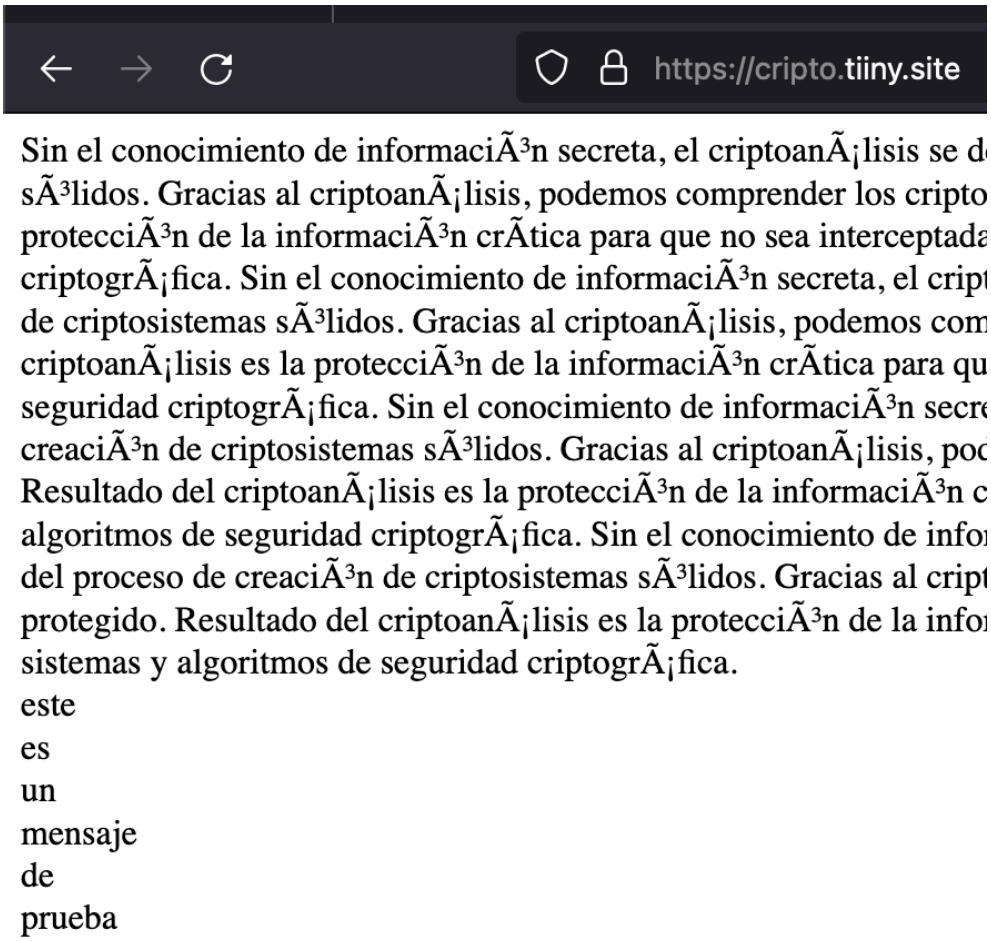

Figura 4: Mensajes mostrados en la consola en texto plano.

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

Las invariantes que considera el script son que el texto tenga una llave escondida entre las mayúsculas y la cantidad de mensajes son tantos como etiquetas div existan. Mientras esto se respete el script sigue funcionando.

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

El archivo lab4/Lab4.js del repositorio `https://github.com/wzrdd/cripto` contiene el script utilizado.

Conclusiones y comentarios

En la pasada experiencia se puede mostrar una manera de transmitir un mensaje dentro de un medio público con un poco de ingenio (usar las mayúsculas de un texto) y la utilización de métodos de cifrado para ofuscar o dificultar su decifrado.