Ataques tecnológicos de ingeniería social en sistemas de red

"Anti-phishing Web con Machine Learning"

Trabajo Fin de Máster

Máster Universitario en Ingeniería Informática

Autor: Pablo Castellanos Santamaría.
Director: Antonio Robles Gómez.
Codirector: Roberto Hernández Berlinches.

ESCUELA TÉCNICA SUPERIOR DE INGENIERIA INFORMÁTICA
UNIVERSIDAD NACIONAL DE EDUCACIÓN A DISTANCIA
MÁSTER EN INGENIERÍA INFORMÁTICA
Madrid, octubre 2020

Resumen

Este Trabajo de Fin de Máster contiene una descripción del problema actual que representan los ataques tecnológicos de ingeniería social en los sistemas de red poniendo el foco en el más extendido de estos ataques, el phishing. A parte de describir el problema y sus repercusiones, para la realización de este trabajo se han analizado las principales técnicas que existen dentro de la literatura científica actual para el desarrollo de sistemas de prevención e identificación de estos ataques. Por último, se ha desarrollado un algoritmo de *Machine Learning* para la implementación de un sistema automático de detección de phishing en tiempo real donde se exploran algunas de las medidas de seguridad estudiadas durante la fase teórica del presente trabajo de fin de máster con el objetivo de estudiar su eficacia en la lucha contra los ataques de ingeniería social en los sistemas de red mediante la comparación de los resultados experimentales obtenidos.

Palabras clave: *Ingeniería social, Phishing, Ataques de red, Algoritmo, Machine Learning, Tiempo Real*.

Abstract

This Master's Thesis contains a description of the current problem posed by social engineering technological attacks on network systems, focusing on the most widespread of these attacks, phishing. In addition to describing the problem and its repercussions, the main techniques that exist within the current scientific literature for the development of systems for the prevention and identification of these attacks have been analysed to carry out this work. Finally, a *Machine Learning* algorithm has been developed for the implementation of an automatic phishing detection system in real time. In this algorithm some of the security measures studied during the theoretical phase of this master's thesis are explored with the objective of review its effectiveness in the fight against social engineering attacks on network systems by comparing the experimental results obtained.

Keywords: *Social Engineering, Phishing, Network Attacks, Algorithm, Machine Learning, Real Time*.

Glosario de términos

TFM: Trabajo de Fin de Máster.

PAN: Personal Area Network o Red de Área Personal, es un tipo de red de computadoras de corto alcance con el objetivo de interconectar los dispositivos electrónicos de una persona.

Salt: En castellano (sal), consiste en un conjunto de bits aleatorios que se utilizan a la vez que una contraseña dentro de una función de generación de una clave con el objetivo de cifrar la contraseña utilizada durante el proceso. La principal utilidad que tiene es dificultar los ataques por fuerza bruta basa llegando a duplicar la cantidad de almacenamiento y de computación requerida en estos ataques por cada bit de Salt utilizado.

NIST: National Institute of Standards and Technology o Instituto Nacional de Estándares y Tecnología hace referencia a las siglas de la agencia de Estados Unidos encargada de promover la innovación y la competencia industrial en ese país mediante la elaboración de normas y métricas aplicadas a la tecnología.

OTP: One Time Password, se trata de una contraseña destinada a ser utilizada una única vez durante un protocolo de autenticación.

TF-IDF: Term Frequency – Inverse Document Frequency o Frecuencia de Término – Frecuencia Inversa de Documento, es un valor numérico que indica la importancia de una palabra dentro de un documento.

HTTPS: Hypertext Transfer Protocol Secure es una extensión del protocolo HTTP (Hypertext Transfer Protocol). La principal diferencia con el protocolo HTTP es que aquí las comunicaciones en envían cifradas mediante el uso del mecanismo de capa de transporte conocido como Transport Layer Security o TLS.

SSL/TLS: Secure Sockets Layer/ Transport Layer Security son dos protocolos de cifrado de la capa de transporte del modelo OSI y que proporcionan una comunicación segura a través de una red de ordenadores.

SVM: Support Vector Machines o Máquinas de Vectores de Soporte son un conjunto de algoritmos de aprendizaje supervisado que está relacionados con los problemas de clasificación y regresión. A partir de un conjunto de datos de entrenamiento permite obtener la clase de los datos y construir un modelo de predicción de clase para una nueva muestra.

RBF: Radial Basis Function o Función de Base Radial es un tipo de función de núcleo utilizada en varios algoritmos de aprendizaje automático y que se utiliza frecuentemente en clasificación utilizando las máquinas de vectores de soporte.

SVC: Support Vector Clustering es un método similar al de SVM también basado en el uso de funciones de núcleo pero que está destinado a problemas de aprendizaje no supervisado.

URL: Uniform Resource Locator o Localizador de Recursos Uniforme es un texto que referencia a un recurso web especificando su localización dentro de una red de ordenadores y el mecanismo para recuperarlo.

TLD: Top Level Domain o Dominio de Nivel Superior es el dominio de más alto nivel dentro del sistema de nombres de dominio de Internet.

IP: La dirección IP es un conjunto de números que identifica a una Interfaz de un dispositivo (ordenador, teléfono, etc.) en la red que utilice el protocolo del mismo nombre (Internet Protocol correspondiente a la capa de red del modelo TCP/IP.

INCIBE: Instituto Nacional de Ciberseguridad de España (INCIBE), es la entidad de referencia destinada al desarrollo de la ciberseguridad y al fortalecimiento de la confianza digital de ciudadanos, profesionales, empresas de todo el territorio español.

OSI: Oficina de Seguridad del Internauta (OSI) es un organismo perteneciente al INCIBE y cuya finalidad es proporcionar información y soporte para evitar y resolver los problemas de seguridad que pueden existir al navegar por Internet.

DOM: Document Object Model o Modelo de Objetos del Documento es el conjunto estándar de objetos para representar documentos HTML, XHTML y XML.

HTML: Hyper Text Mark-up Language o Lenguaje de Marcas de Hipertexto, es el lenguaje estándar para la creación de páginas web a través del cual se define mediante etiquetas el contenido de una página web.

DNS: Domain Name Server o Servidor de Nombres de Dominio, es el sistema de nombres para dispositivos electrónicos que se conectan a una red de tipo IP y que permite acceder a sus recursos a partir de un nombre en lugar de una dirección IP.

PageRank: Conjunto de algoritmos creados por Google para clasificar en función de su relevancia a las páginas web que son indexadas por su motor de búsquedas.

Agradecimientos

A mi tutor, por la ayuda y la guía para la realización de este TFM.

A mis padres, por enseñarme la importancia de la educación universitaria y por ser mi ejemplo y mi apoyo.

A Mayka, por animarme a seguir estudiando y por todos los ratos que ha cuidado de nuestra pequeña para que yo pudiera sacar tiempo para la realización de este trabajo.

A Celia, por alegrarme la vida con su sonrisa.

Índice de figuras

Índice de contenidos

1. Introducción

Este TFM pretende realizar una descripción de problema que suponen los ataques de ingeniería social en los sistemas de red centrándose sobre todo en el tipo de ataque más repetido entre los delincuentes como son los basados en técnicas de phishing. Este tipo de ataques suponen una grave amenaza tanto a nivel personal como a nivel empresarial.

En un primer lugar, en el trabajo se realiza una breve introducción al concepto de seguridad informática, así como a los principales objetivos y amenazas a la seguridad en las tecnologías de la información existentes en la actualidad.

En los siguientes capítulos, el tema central del trabajo se centra en el principal tipo de ataque de ingeniería social, el Phishing. Esta técnica es una de las más antiguas que utilizan los cibercriminales y hoy en día sigue siendo una de las que más tasa de éxito tiene entre los delincuentes de ahí que use de manera intensiva para robar los datos de acceso de los usuarios a los servicios web que utilizan.

En el estado del arte se van a definir los distintos tipos de ataques de phishing que se conocen en la actualidad, así como se detallarán las técnicas tradicionales utilizadas para la detección de este tipo de amenazas digitales. Por último, en este capítulo se van a presentar los enfoques más modernos que los investigadores y los expertos en ciberseguridad han propuesto para desarrollar sistemas seguros de detección de phishing basados principalmente en dos conceptos, la autenticación en varios factores y el empleo de algoritmos de para la clasificación en tiempo real de páginas web en legítimas o fraudulentas.

Para acompañar esta presentación teórica se propondrán varios ejemplos de detección de phishing utilizando entornos controlados de laboratorio donde se comprobará la eficacia de las más modernas soluciones existentes para la defensa de estos ataques de ingeniería social.

1.1. Motivación

Como se ha descrito en el capítulo anterior, los ataques de Ingeniería social son muy variados y difíciles de detectar porque para su éxito dependen en gran medida del factor humano. Las principales técnicas de defensa para los ataques de este tipo como el phishing pasan en gran medida por la formación de los usuarios y su entrenamiento con campañas de simulación de ataques. Por este motivo considero que es de vital importancia poder generar mecanismos de defensa automáticos para apoyar a la detección de estos ataques antes de que se consuman para evitar los graves problemas que llevan asociados.

En caso de que logren su objetivo, este tipo de ataques puede suponer la imposibilidad de acceso a algún servicio online como el correo electrónico o incluso puede suponer pérdidas económicas sustanciales en los casos más graves si se han revelado datos de tarjetas de crédito. Además, los ciberdelincuentes pueden utilizar la información personal obtenida durante un ataque para realzar cargos o compras por internet.

Algunos estudios realizados [CYBRIANT20] determinan que más del 90 % de los ciber ataques de cualquier tipo comienzan con el envío de correos de phishing. El FBI reportó [FBI20] que entre el 2013 y el 2018 las empresas podrían haber perdido aproximadamente 12,500 millones de

dólares relacionados con ciber ataques de phishing. Solo en Estados Unidos se estiman unas pérdidas anuales de 500 millones de dólares asociados a fraudes de phishing [FORBES20].

La consultora tecnológica Accenture revela en su informe anual sobre el estudio de los costes de los delitos relacionados con la ciberseguridad [ACCENTURE20], que solo en 2018, los delitos cibernéticos supusieron unas pérdidas de 8,16 millones de dólares en España, 9,72 millones de dólares en Francia, 13,12 millones de dólares en Alemania o 13,57 millones de dólares en Japón. Todas las mediciones que se realizan no dejan de aumentar cada año debido principalmente a transformación digital que está teniendo lugar en la mayoría de las empresas a nivel mundial, así como en el perfeccionamiento de las técnicas de ataque de los cibercriminales. Es por este último motivo por el que resulta de vital importancia desarrollar nuevas técnicas de defensa que dificulten la labor de los delincuentes.

1.2. Planificación

Para la realización de este TFM se han destinado los siguientes recursos tanto temporales como a nivel de hardware y software.

Desde un punto de vista temporal, han sido necesarios 4 meses de dedicación a tiempo parcial compatibilizándolo con mi actividad laboral. El tiempo se ha destinado de la siguiente manera:

Una primera fase de aproximadamente un mes de duración durante el cual se ha realizado una investigación del estado del arte, así como de las principales publicaciones científicas existentes relacionadas con el problema de los ataques de ingeniería social que se analizan en el presente TFM. Una vez superada esta fase inicial, el tiempo se ha dedicado a profundizar el estudio de los temas más complejos sobre los que se discute en la literatura existente y a comenzar la escritura de documento de la memoria para poder ir recibiendo los comentarios y el *feedback* del equipo docente encargado de guiar a buen puerto este TFM. Durante la fase final de la elaboración de este TFM y una vez que la memoria ha estado encaminada hacia su versión final, se ha dedicado el tiempo a la realización de la parte experimental donde se ponen en práctica algunas de las contramedidas que han sido mencionadas por los autores de las publicaciones científicas consultadas para la realización de este TFM.

La figura 1 contiene una tabla con la descripción detallada en horas de la evolución temporal del tiempo de dedicación requerido para la realización de este TFM:

Tarea realizada	Horas de dedicación	Total de horas acumuladas
Investigación sobre el estado del arte.	50	50
Lectura de artículos científicos relacionados.	30	80
Búsqueda de información sobre algoritmos de ML.	30	110
Aprendizaje sobre el uso de las principales librerías de ML en Python.	20	130
Elaboración del documento con la memoria del TFM.	100	230
Elaboración de la parte experimental del TFM. Diseño del algoritmo utilizado.	50	280
Evaluación y comprobación de los resultados obtenidos.	20	300

Figura 1. Evolución temporal del proyecto realizado.

En lo que respecta a los recursos materiales para la realización del presente TFM se ha contado con el siguiente equipo informático:

Ordenador de sobremesa con un procesador Intel Core i5-4690K @ 3.50GHz, 20 GB de memoria RAM, Tarjeta Gráfica Radeon RX 580 con 8 GB de memoria RAM y un sistema operativo Windows 10 Pro de 64 bits. Actualmente este equipo tendría un coste de mercado aproximado de unos 600 €.

El software utilizado ha sido el siguiente:

- Suite ofimática Microsoft Office 365.
- Editor de texto Visual Code 1.48.2.
- Anaconda Navigator 1.9.12.
- Python 3.7.
- Oracle VM VirtualBox 6.1.

En cuanto a los recursos humanos requeridos, si este TFM hubiese sido un proyecto de ingeniería realizado en un marco laboral habría requerido del siguiente personal:

- Un jefe de proyecto encargado de la organización del trabajo y la gestión de los recursos materiales, tanto del tiempo como del equipo informático y el software utilizado. Este tipo de trabajador tiene un coste aproximado por hora de 180 €. Para la realización de este TFM habría sido necesario contratar 200 horas por lo que el coste total seria de 36.000 €.
- Un programador encargado de codificar el algoritmo del caso de estudio según las especificaciones dictadas por el jefe de proyecto en algún tipo de especificación formal de requisitos. El coste hora de un trabajador de esta categoría es de 150 €. Para la realización de este TFM habría sido necesario contratar 100 horas por lo que el coste total seria de 15.000 €.

La figura 2 contiene el desglose de los gastos asociados a los recursos necesarios para la realización de este TFM como si de un proyecto de ingeniería se tratara:

Recurso	Coste total en Euros
Ordenador personal	600 €
Windows 10 Pro-64 Bits (Coste de la licencia)	259 €
Suite ofimática Microsoft Office 365 (Coste licencia anual)	69 €
Editor de texto Visual Code	Sin coste
Anaconda Navigator	Sin coste
Python	Sin coste
Oracle VM VirtualBox	Sin coste
Ordenador (valor aproximado)	600 €
Coste Personal (Jefe de Proyecto)	36000 €
Coste Personal (Programador)	15000 €
Costes totales	**52528 €**

Figura 2. Desglose de los costes por cada recurso necesarios para la realización de este TFM como un proyecto de ingeniería.

2. Estado del arte

El problema del phishing supone una amenaza para todos los usuarios de Internet. Actualmente la gran mayoría de la información que manejamos se encuentra replicada en la red y la protección de dicha información como de las credenciales de acceso a la misma es una tarea de vital importancia y que por desgracia aún no contamos con contramedidas efectivas contra estas amenazas.

2.1. Seguridad de las tecnologías de la información

La seguridad en el área de las tecnologías de la información o ciberseguridad hace referencia a la protección de las infraestructuras computacionales (hardware), así como en la información contenida en los ordenadores tanto el software como los datos, así como la seguridad de las redes de ordenadores.

La ciberseguridad es un asunto de vital importancia en la época actual y cuyo objetivo principal es el de minimizar los riesgos de que la información o la infraestructura caigan en las manos equivocadas protegiendo los activos ante las posibles amenazas. Las amenazas nunca se eliminan, se mitigan (reducción del riesgo). Podemos entender la seguridad como una cualidad relativa que expresa el balance entre el riesgo (amenazas, vulnerabilidades e impacto sobre activos) y las medidas adoptadas para paliarlo.

A continuación, se van a presentar algunos de los principales conceptos clave dentro de la seguridad en las tecnologías de la información.

Los activos

Los activos son los componentes de un Sistema de información susceptibles de ser atacados deliberada o accidentalmente con consecuencias negativas. Los activos de un sistema de tecnologías de la información deberán cumplir con las siguientes tres características: la confidencialidad (revelarse sólo a usuarios autorizados), la integridad (que solo se puedan modificar por usuarios autorizados) y la disponibilidad (que estén siempre accesibles para usuarios legítimos).

Las vulnerabilidades

Una vulnerabilidad, es la exposición a una amenaza, y cuya explotación viola la política de seguridad del sistema de información. El siguiente gráfico muestra el total de vulnerabilidades que el Instituto Nacional de Estándares y Tecnología del Departamento de Comercio de los Estados Unidos de América conocido por sus siglas en ingles NIST lleva identificando y publicando desde el año 1988. Hasta la fecha se han reportado y publicado más de 130000 vulnerabilidades como puede observarse en la figura 3 donde vemos un gráfico con la evolución del número de vulnerabilidades registradas por año desde 1988.

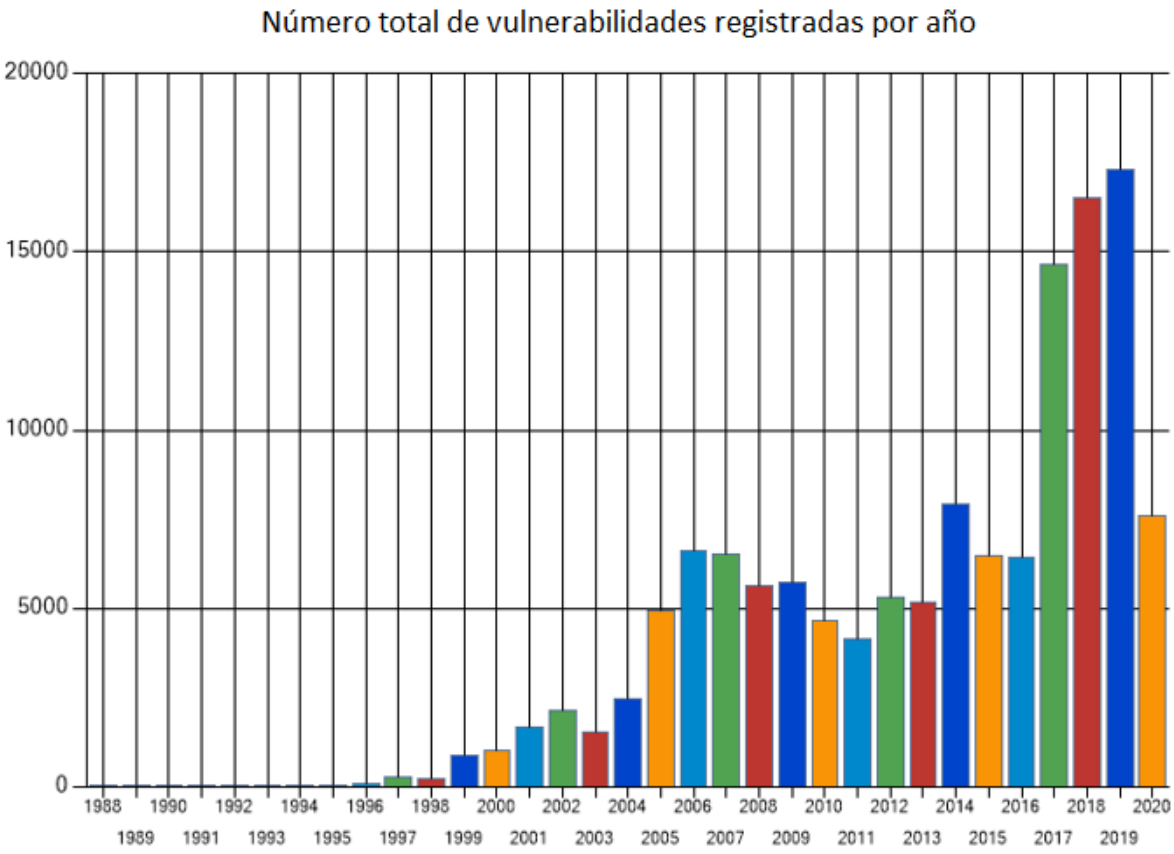

Figura 3. Evolución del número de vulnerabilidades registradas por el NIST cada año desde el 1988 al 2020.

En el mundo de la seguridad informática, los atacantes siempre tienen ventaja frente al defensor ya que un atacante sólo necesita encontrar una vulnerabilidad, mientras que el defensor debe eliminarlas todas.

Las amenazas

Las amenazas de los sistemas de información son las potenciales violaciones de seguridad accidentales o intencionadas que pueden sufrir dichos sistemas y representan cualquier acción que ponga en peligro los objetivos de seguridad prefijados. Muchas de estas amenazas son imprevisibles de modo que la única contingencia posible son la redundancia y la descentralización. Podemos clasificar las amenazas según su origen en:

- Accidentales (fallos en los equipos, anomalías en el suministro de energía, mal funcionamiento de programas, etc.).
- Naturales (fuego, inundaciones, etc.).
- Intencionadas (fraudes, sabotaje, etc.).

Pero también las podemos clasificar en función de su actuación en:

- Pasivas (interceptación).
- Activas (generación, modificación o interrupción).

Los ataques

Son las acciones maliciosas encaminadas a explotar una o varias vulnerabilidades con el fin de materializar una amenaza de manera intencionada.

Con el fin de minimizar los riesgos asociados a la seguridad informática relativos a los activos en un tiempo y un lugar determinado es necesario que se lleven a cabo medidas suministradas por un sistema o un producto. Estas medidas se pueden clasificar en función de su forma de actuación como son las medidas de prevención, de detección, de corrección y de recuperación o bien según su naturaleza, medidas legales, administrativas, físicas o técnicas.

Según la norma [ISO7498-2] , un servicio de seguridad consiste en la prestación suministrada por uno o más niveles de un Sistema de Información que contribuye a la seguridad de este, así como a la transferencia de datos y los mecanismos de seguridad y su implementación física o lógica diseñada y construida para contrarrestar un ataque. Cada servicio de seguridad es instrumentado mediante uno o varios mecanismos de seguridad.

Los principales mecanismos de seguridad son el cifrado, la firma digital, el control de accesos, las funciones de autenticación, el control de flujo, el rellenado de tráfico y el notariado.

En el diagrama de la figura 4 se pueden ver todos los conceptos que se han explicado en este apartado y las relaciones que guardan entre ellos. El mapa conceptual que se ha elaborado para este TFM comienza en los mecanismos de seguridad más extendidos dentro de los sistemas de información como son: el cifrado, la firma digital, la autenticación o el control del acceso. Estos mecanismos implementan los servicios de seguridad destinados a mitigar las amenazas antes de que se conviertan en un posible ataque. Los ataques a su vez se aprovechan de las vulnerabilidades existentes para comprometer la integridad, disponibilidad confidencialidad de los activos que forman parte de los sistemas de información.

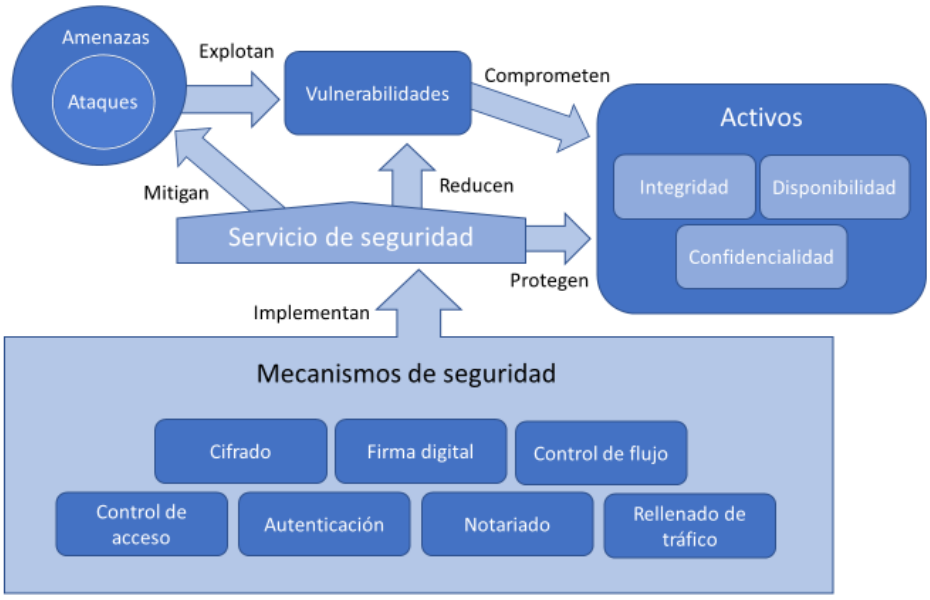

Figura 4. Mapa conceptual de la seguridad informática.

2.2. Ataques de ingeniería social

En la sección anterior se ha definido el concepto de ataque dentro del campo de la ciberseguridad como las acciones maliciosas encaminadas a explotar una o varias vulnerabilidades con el fin de materializar una amenaza de manera intencionada.

Los ataques de ingeniería social son aquellos tipos de ataques se basan en la obtención de información sensible de carácter confidencial a través de la manipulación intencionada de los usuarios legítimos de un sistema de información con el objetico de que el atacante obtenga información o accesos privilegiados a los sistemas de información para poder realizar acciones malintencionadas.

Los atacantes que realizan este tipo de ataques se denominan ingenieros sociales y para lograr sus objetivos utilizan el teléfono o internet para engañar a sus víctimas suplantando la identidad de otra persona, de una empresa o hasta de un compañero de trabajo de la víctima con el objetivo de obtener credenciales de acceso o información confidencial. Estos atacantes basan sus técnicas en principios psicológicos que permitan engañar al usuario en lugar de buscar como explotar vulnerabilidades tecnológicas dentro de los sistemas de información. La idea de un ataque de este tipo es conseguir recrear una situación irreal que pueda ser factible para la víctima y que la lleve a revelar información confidencial. Los atacantes utilizarán una serie de técnicas para lograr obtener la confianza de su víctima como pueden ser utilizar un determinado tono de voz que inspire autoridad, tener preparadas varias creíbles para las posibles preguntas que le surjan a la víctima o bien proporcionar información que consiga suplantar a otra persona o entidad.

El proceso de obtención de información para realizar un ataque de ingeniería social habitualmente comienza con una simple búsqueda en internet sobre la víctima. Dado el uso creciente de las redes sociales no es difícil para un atacante encontrar información personal y sensible para preparar las bases sobre las que realizar un ataque de ingeniería social. Por ejemplo, es posible encontrar imágenes de los lugares que frecuenta una persona, las relaciones que tiene, así como sus gustos personales de una manera bastante sencilla. Toda esa información que puede resultar inofensiva en un primer momento puede ser utilizada por un delincuente con el objetivo de establecer una base sólida para realizar un ataque de ingeniería social.

Como se ha mencionado anteriormente, este tipo de ataques este cimentado sobre aspectos psicológicos que se basan en explotar alguno de los siguientes principios básicos:

- **El respeto a la autoridad**. La mayoría de los ciudadanos y de los trabajadores tiene respeto por la autoridad de nuestros responsables o bien de las autoridades gubernamentales. Esta actitud puede ser explotada por un atacante que suplante la figura de autoridad de la víctima.
- **El temor a perder un servicio**. Este miedo se suele explotar en campañas de fraude con el pretexto de que si no se accede a un determinado sitio web y se introducen las credenciales de acceso se pierde algún tipo de servicio contratado por la victima legítimamente. Los atacantes se aprovechan de esa situación y crean portales web falsos para capturar las credenciales de acceso de los usuarios.
- **El ofrecimiento de servicios o bienes de manera gratuita**. A cambio de ofrecer algún producto o servicio de manera gratuita, los atacantes esperan recibir a cambio información privilegiada de la víctima.
- **La solidaridad entre los trabajadores.** En un entorno laboral es habitual que los trabajadores tengan entre sí la voluntad de ayudarse entre ellos lo que puede ser utilizado por un atacante que simule ser un compañero de trabajo o incluso pertenecer al servicio técnico de la empresa para poder acceder a los medios informáticos del trabajador y ganar una puerta de acceso a la compañía a través de la instalación de algún tipo de programa malicioso en el equipo de la víctima.

- **Necesidades sociales.** Los atacantes pueden explotar las necesidades de aceptación social de las personas o incluso el miedo de perder su reputación debido a haber realizado algún tipo de comportamiento socialmente no aceptado. Este tipo de necesidades se suele explotar en extorsiones con carácter sexual basados en la amenaza de difundir algún tipo de contenido sexual de la víctima que ni si quiera tiene por qué llegar a existir.

Este tipo de ataques son muy difíciles de prever y de proteger puesto que están sustentados sobre el principio de que los usuarios son el eslabón más débil dentro de un sistema de información. La mejor prevención en estos casos consiste en la formación y concienciación de los usuarios de los sistemas de información y de los empleados de una empresa. De nada sirve implementar las medidas de seguridad más eficaces si luego los propios empleados facilitan las credenciales de acceso por culpa de un ataque de este tipo.

A continuación, se describen cuáles son los tipos de ataques de Ingeniería social más comunes:

Phishing

Es el tipo de ataque de ingeniería social más extendido y exitoso. Consiste en utilizar una serie de técnicas para engañar a las personas y obtener información [OSI20_1],[OSI20_2]. Los atacantes desean hacerse pasar por figuras de autoridad como pueden ser los administradores de un sistema de información con el objetivo de persuadir a las víctimas para que divulguen información restringida o confidencial que tienen en su poder. Este tipo de ataques suelen realizarse mediante el envío de correos electrónicos y pueden contener código malicioso camuflado dentro de ficheros adjuntos que deben ser ejecutados por las víctimas del ataque, de ahí la necesidad del engaño para que los destinatarios del correo electrónico tomen decisiones basadas en el miedo y en la urgencia en lugar de actuar de manera racional.

Vishing

Se trata de un tipo de ataque de ingeniería social en el que el delincuente realiza una llamada a un empleado de una empresa haciéndose pasar por un colaborador, por un compañero o por un representante de otra empresa y realiza una serie de preguntas para socavar información que podrá ser utilizada más adelante en algún otro ataque. Este tipo de ataque es más complejo puesto que requiere habilidades sociales y de una mayor interacción humana.

Baiting

Consiste en utilizar un dispositivo de almacenamiento extraíble como puede ser una unidad USB o un CD con software maliciosos y abandonarlo en un lugar público como una cafetería o un ascensor y que sea frecuentado habitualmente por la posible víctima del ataque con la esperanza de que lo recoja y lo introduzca en su ordenador infectándose y poniendo a disposición del atacante la información que contenga.

Quid pro quo

Consiste en ofrecer algo a cambio de algo. Un atacante puede por ejemplo ofrecer su ayuda para solucionar un problema técnico legítimo de la víctima si esta le facilita información de acceso con el objetivo de solucionar el problema. Otro tipo de ataque de este tipo consiste en ofrecer grandes descuentos en productos generalmente de precios elevados siempre que se rellene algún tipo de formulario con información personal valiosa para un atacante.

Farming

Este es un tipo de ataque de larga duración en el que los atacantes establecen una relación con un objetivo basándose en la información que han obtenido de una fase de investigación previa. Durante el tiempo que dure el engaño, los atacantes trataran de obtener la mayor cantidad de información que puedan a través de la víctima.

Lo normal en los ataques de ingeniería social es que los delincuentes combinen algunos de los ataques definidos con anterioridad para lograr sus objetivos.

2.3. Phishing

En este apartado se detalla uno de los tipos de ataque de ingeniería social más conocidos y efectivos es el denominado Phishing. Este término hace referencia todo un conjunto de técnicas destinadas a que un atacante engañe a una víctima haciéndose pasar por una persona, organización o un servicio de confianza con el objetivo de manipular a la víctima y persuadirle para lograr que lleve a cabo alguna acción que no debiera realizar como puede ser revelar información importante o privilegiada o la sustracción de credenciales de acceso de carácter confidencial. Normalmente, el objetivo es el robo de información, pero en otras ocasiones también puede ser llevar a cabo la instalación de malware, robar dinero a través de fraudes o el sabotaje de sistemas de información.

La forma más habitual de phishing consiste en el envío de mensajes electrónicos, generalmente por correo electrónico, aunque también pueden ser mensajes de texto a un teléfono o de redes sociales, donde el atacante suplanta a una entidad legitima o una persona que cuente con la confianza de la víctima para de esta manera poder manipularla que ponga en peligro sus datos.

En la literatura consultada [VayanskiSathish2018], [NORTON20] se analizan los distintos tipos de phishing más extendidos y en la lista siguiente se ha tratado de recopilar y de definir los principales tipos de phishing más frecuentes:

Phishing general

Consiste en el envío de correos electrónicos de forma masiva suplantando a entidades de confianza con el objetivo de engañar a los usuarios y obtener su información privada.

Vhising

Es un tipo de ataque muy parecido al phishing general solo que en esta ocasión el medio utilizado es una llamada telefónica.

Smishing

El atacante manda un mensaje de texto ya sea un SMS o por algún servicio de mensajería instantánea como WhatsApp o Telegram haciéndose pasar por alguna entidad bancaria con alguna comunicación urgente y facilitando un número de teléfono falso en el que atenderán su caso personalmente.

URL Phishing

El engaño consiste en crear una dirección de internet parecida a una real y a la que pueda llegar el usuario cuando tecle mal la dirección de internet en el navegador web.

Whaling

El objetivo de este tipo de ataque son personas con un puesto relevante dentro de una organización haciendo que este tipo de ataque sea más personalizado que los anteriores.

Spear Phishing

Al igual que en el caso anterior, este tipo de ataque vs dirigido contra una persona especifica dentro de una organización o empresa por lo que es necesario que se realice un trabajo de investigación por parte de los delincuentes antes de llevar a cabo el ataque.

Search Engine Phishing

Los atacantes desarrollan un sitio web de apariencia legitima con contenido atractivo y lo posicionan mediante técnicas SEO para que aparezca entre los resultados legítimos de los principales buscadores de internet.

Phishing con evasión de filtros

Sustituyen textos que suelen ser detectados por los sistemas de detección de phishing de los servidores de correo electrónico por imágenes o protegen con contraseña los adjuntos con malware con el fin de burlar este tipo de mecanismos de protección automáticos.

Pharming

Para este ataque es necesario que previamente se vulnere la seguridad de un servidor de internet de traducción de dirección IP a nombre de dominio de tal manera que cuando el usuario introduzca una dirección de internet en su navegador el servidor DNS le redireccione a un sitio fraudulento creado por los ciberdelincuentes.

Watering Hole Phishing

Es un tipo de ataque en el que se infecta con algún tipo de malware algún sitio web de un tercero que sea de uso común para los miembros de una determinada organización o empresa con el objetivo final de que los empleados de esa empresa se infecten cuando acceda a la página del tercero.

Evil Twin

Consiste en crear un punto de acceso malicioso que proporcione conectividad a través de una red Wireless en apariencia legitima pero que en realidad está siendo contralada por los delincuentes.

Social Network Phishing

Esta categoría incluye todos los tipos de phishing relacionados de alguna manera con las redes sociales como puede ser la publicación de entradas falsas en cuentas que previamente se han visto comprometidas en un ataque anterior.

Man in the Browser

A través de la instalación de programas maliciosos en el navegador web de una víctima se puede llegar a insertar información adicional en páginas legítimas que visita la victima para cometer el fraude y facilitar el engaño ya que la información fraudulenta se añade a paginas legítimas dificultando de esta manera su detección.

Phishing móvil

En este tipo de ataques se suelen utilizar aplicaciones maliciosas para dispositivos móviles que recopilan información personal de las víctimas para perfeccionar algún ataque posterior.

Hardware Phishing

Ataques en los que los ciberdelincuentes introduzcan código malicioso dentro de equipos electrónicos que posteriormente serán vendidos par que se activen cuando sean comprados por una víctima.

Los mensajes fraudulentos utilizados para este tipo de fraude suelen hacer uso de todo tipo de estratagemas para engañar a sus víctimas pretendiendo forzarlas a tomar una decisión rápida que escape a un razonamiento lógico que de no ser obedecido pueda tener graves consecuencias para el receptor del mensaje como puede ser la perdida de algún servicio contratado o el acceso a una determinada plataforma o sitio web. Es muy habitual que dentro de los mensajes enviados por los ciberdelincuentes se incluyan enlaces manipulados a paginas fraudulentas que fingen ser legítimas o bien incluir datos adjuntos con código malicioso para que sea ejecutado por la víctima y comprometa la seguridad de su equipo informático.

Los principales servicios online que pretenden suplantar los delincuentes en los ataques de phishing suelen estar relacionados con las siguientes actividades: Bancos y cajas online, pasarelas electrónicas de pago como PayPal o Visa, redes sociales (Twitter, Facebook, LinkedIn, etc.), páginas web de compra/venta como Amazon o eBay, juegos online, el soporte técnico de grandes compañías con muchos usuarios como Apple, Gmail, etc., servicios públicos como la Agencia Tributaria o Correos, servicios de almacenamiento basados en la nube como Dropbox, Google Drive o OneDrive, servicios de paquetería como DHL o UPS, vales descuento para utilizar en el tiendas de comercio electrónico, supermercados o restaurantes o incluso falsas ofertas de empleo.

El Instituto Nacional de Ciberseguridad [INCIBE20] a través de la Oficina de Seguridad del Internauta [OSI20-3] se dedica a recopilar ejemplos reales recientes de ataques de phishing con el objetivo de difundirlos y contribuir a la educación digital de los usuarios de internet.

A continuación, se muestran algunos de los ejemplos difundidos por el OSI:

SMS fraudulentos con enlace a una web para solicitar una supuesta ayuda económica con motivo de la pandemia originada por el COVID19

El día 12 de mayo de 2020, se ha publicado una noticia sobre un nuevo caso de phishing en el que los atacantes envían un SMS relacionado con la pandemia COVID10 con el siguiente texto:

"*Hola, en relación con la pandemia COVID19 el Estado destina 350-700 euros a sus ciudadanos https://xxxxxxx.es*"

Si se accede a la URL que aparece en el SMS, redirige al usuario a una supuesta página que simula pertenecer a un organismo oficial del estado. La figura 5 contiene una captura de pantalla de la página web fraudulenta donde podemos obtener la supuesta ayuda.

Figura 5. Ejemplo de un caso de phishing reportado por el INCIBE que utilizó como reclamo la pandemia sanitaria COVID19 [OSI20-4].

Para obtener la ayuda económica, nos solicitan información personal, así como los datos de nuestra tarjeta de crédito supuestamente para poder cobrar la ayuda.

En este ejemplo de ataque, se han aprovechado de la situación excepcional ocasionada por la pandemia sanitaria del COVID19 y por la necesidad económica de una gran parte de la población para recibir una supuesta ayuda económica.

Campaña fraudulenta de correos relativos al pago de multas de tráfico

El pasado 18 de marzo de 2020 se detectó una campaña de phishing en el que se enviaba un correo electrónico suplantando la identidad de la Dirección General de Tráfico en el que se apremiaba al receptor del correo a pagar una multa de tráfico de manera urgente. El asunto del correo era “PAGA TU MULTA”. El correo incluía los logotipos e l ministerio del interior y de la DGT, así como un enlace malicioso que descarga un fichero de malware que infectara el equipo del destinatario en caso de ser descargado.

Después de haber revisado algunos de los últimos casos reportados de phishing se puede constatar el peligro de este tipo de ataques y lo elaborado de los mismos. Para evitar ser víctima de este tipo de fraudes, el propio INCIBE detalla las siguientes recomendaciones para que los usuarios puedan detectar este tipo de correos:

- Cualquier mensaje de entidades bancarias o de servicios online reconocidos que pueda resultar alarmista e inesperado debe hacernos sospechar que estemos ante una suplantación de identidad o un intento de phishing.
- Es muy poco habitual que cualquiera de las empresas conocidas de internet nos solicite nuestros datos de acceso a través de un mensaje personal.
- Además, debemos sospechar si encontramos errores gramaticales en el texto del mensaje. Lo más seguro es que hayan utilizado herramientas automáticas de traducción

de texto con el objetivo de difundir el mismo ataque al mayor número de personas posible. Ningún servicio con cierta reputación enviará mensajes mal redactados.

- La mayoría de los ataques de phishing cuentan con el factor de urgencia para obligarte a tomar una decisión precipitada. Si el correo recibido implica la necesidad de actuar inmediatamente lo mejor es contrastarlo con otra fuente de información alternativa de la fuente como puede ser realizando una llamada telefónica para contrastar el comunicado recibido.
- Las comunicaciones de las empresas sobre las que tienes una cuenta de usuario registrada ya tienen información tuya como tu nombre por lo que, si el mensaje recibido se dirige a ti de manera anónima como "Estimado cliente, o "Querido usuario de" deberías sospechar.
- Revisa siempre los remitentes de correo del mensaje recibido, cualquier empresa sería contará con un servicio de dominio propio para enviar sus comunicaciones corporativas y no mandará correos electrónicos desde direcciones de correo electrónico gratuitas como puede ser Hotmail, Gmail, Yahoo, etc.
- Revisa siempre las direcciones de internet que aparecen en el mensaje en forma de enlace. Los ciberdelincuentes suelen utilizar direcciones de internet parecidas, pero nunca iguales que las legítimas como por ejemplo sustituyendo letras por números que se puedan confundir en una lectura rápida como por ejemplo cambiar la letra i por el numero 1 o incluir guiones o caracteres adicionales al nombre de una empresa.
- Extrema las precauciones cuando se trate de correos relacionados con entidades bancarias o con cualquier plataforma digital que maneje dinero como comercios electrónicos y te soliciten las credenciales de acceso a sus servicios. No se trata de una práctica habitual y es prácticamente seguro que se trata de algún tipo de fraude.

Los ataques de ingeniería social relacionados con el phishing son las más extendidos y los más difíciles de defender debido a que, según su naturaleza, no dependen tanto de fallos tecnológicos sino más bien de la ingenuidad o el mal uso que pueden llegar a hacer los usuarios de los sistemas de información y además, por regla general suelen ser bastante sencillos de llevar a cabo siendo en algunos casos solo necesario el envío de un correo electrónico para robar la información de acceso de un usuario. Mediante la recolección de información sobre una víctima, los atacantes pueden crear correos electrónicos personalizados y que resulten creíbles a los ojos de la víctima y sean difíciles de detectar para los sistemas automatizados de detección de correos malintencionados incluso a los usuarios de sistemas que cuenten con la protección de los mejores programas ciberseguridad en sus equipos ya que al final lo que hay que evitar es introducir datos en un formulario y no la infección del equipo a través de la instalación de algún malware.

Con el objetivo de proteger a los usuarios de esta amenaza, podemos dividir el problema de la defensa de estos ataques en dos principales aproximaciones basándonos en cada caso en cada una de las etapas de un ataque de phishing: antes de que el ataque llegue al usuario y una vez que el usuario haya llegado al sitio malicioso. Además, se debe trabajar de manera paralela en realizar una formación continua que permita el entrenamiento de los usuarios de los sistemas de información para que sean capaces de detectar un ataque antes de que este se materialice.

En la actualidad, la gran mayoría de artículos de investigación acerca del phishing hacen referencia a las principales amenazas que puede suponer para la actual sociedad de la información y se centran en utilizar dos enfoques diferentes para la detección del phishing

[QabajehThabtahChiclana2018]. Por un lado, el enfoque más tradicional orientado a la detección previa a su visualización de los correos electrónicos basándose en el análisis individualizado de los enlaces que se incluyen en los correos o en los ficheros adjuntos que llevan y, por otro lado, se centra en un enfoque más moderno que utilice técnicas de *Machine Learning* que permitan identificar estos correos como correos de phishing considerando que se trata de un problema de clasificación principalmente.

Las habilidades que demuestran los delincuentes que se dedican al phishing están a la orden del día y son capaces de generar sitios web prácticamente idénticos a los que pretenden simular incluyendo la misma interfaz gráfica, logos, etc. La parte de los ataques dedicada a la simulación de sitios web se ha perfeccionado mucho en los últimos años e incluso se ha llegado a industrializar de tal manera que los atacantes pueden obtener recreaciones de sitios legítimos a demandan en internet sin la necesidad de tener los conocimientos necesarios para realizar replicas que resulten creíbles. Esto permite a los delincuentes centrarse en idear ingeniosas maneras de engañar a los usuarios y a recopilar la cada vez más frecuente información personal de usuario que podemos encontrar en la red gracias al uso intensivo que buena parte de la población realizad de las redes sociales lo que aumenta las probabilidades de éxito de un ataque ya que los estudios han demostrado que estos ataques tienen hasta 4.5 veces más éxito si provienen de un contacto personal o incluyen información personal con la que la víctima se pueda sentir identificada.

El phishing afecta a todas las personas de manera global y muchas veces está dirigido por organizaciones criminales internacionales que dificultan mucho su persecución por parte de las autoridades gubernamentales.

Los delincuentes que utilizan este tipo de ataques, también conocidos como *phishers*, utilizan técnicas para dificultar en gran medida la localización de la ubicación real del servidor que contienen el sitio aloja el phishing. Alguna de estas técnicas como la que se conoce como "*fast flux*" se centra intercalar numerosos servidores que funcionan a modo de proxy y se utilizan URLs intermedias e incluso redes intermedias.

Si se analizan las principales características de los principales ataques de phishing podemos diferenciar al menos tres tipos principales de acuerdo con su arquitectura:

- Ataques de tipo *Man in the Middle* en tiempo real: En este escenario, los atacantes se sitúan entre el cliente y el servidor mediante el uso de un sitio web intermedio diseñado para parecerse a un sitio web legítimo. El atacante captura la información de autenticación que el usuario introduce en la página web y la transfiere al sitio web legítimo que está suplantando en tiempo real a través de una intervención manual del atacante o bien mediante algún automatismo de tal manera que obtiene el acceso a la cuenta de usuario de la víctima. Este tipo de ataque es más complejo que el típico correo de phishing tradicionales debido a que se suelen utilizar extensiones maliciosas en el navegador de la víctima o programas de captura de la imagen que se muestra por pantalla conocidos como *screen loggers* y que pueden aportar más información en tiempo real para la ejecución del ataque.
- Ataques de tipo *Man in the Middle* con transferencia de control: Este es un ataque todavía más invasivo que el anterior donde el atacante transfiere su escritorio a través de la terminal del usuario engañándole para que introduzca ahí sus credenciales de acceso. En esta tipología de ataques, los sistemas de autenticación en uno o dos factores son poco útiles puesto que el atacante es capaz de interceptar el token e introducirlo en

el sitio web real para realizar una autenticación con éxito. Como medida de prevención, solo son eficaces los sistemas de autenticación basados en algún tipo de token almacenados en hardware.

- Ataques basados en extensiones de navegador maliciosas. Este tipo de ataques se aprovechan de obtener permisos de usuario para realizar alguna tarea legitima como puede ser analizar el texto introducido en los formularios web para corregir posibles faltas de ortografía y de manera oculta utilizar dicho permiso para monitorizar de manera sigilosa la actividad que la víctima realizar en todas las páginas que visita. Los mecanismos de defensa típicos como los captchas basados en texto o en imágenes son vulnerables a este tipo de ataques si el atacante los monitoriza en tiempo real.

La clasificación anterior es necesaria para comprender el alcance de algunas de las soluciones que se describen en este trabajo y conocer mejor cual es el enfoque principal de cada una de las contramedidas de seguridad que existen en la actualidad para tratar de frenar este tipo de ataques digitales.

Podemos clasificar las principales soluciones existentes para mitigar los ataques de phishing en tres grandes tipologías distintas: detección de los ataques de phishing antes de que lleguen al usuario, detección de los ataques de phishing antes de que el usuario alcance el sitio que contiene el phishing y el entrenamiento de los usuarios para lograr prevenirlos que sean ellos mismos los que sean capaces de detectar los sitios fraudulentos. Cada una de estas aproximaciones tienes sus ventajas e inconvenientes y la mejor solución sería utilizar un enfoque que mezcle las tres para reducir al máximo las probabilidades de éxito de un ataque de estas características.

2.4. Sistemas de prevención de phishing

Loa sistemas de prevención de phishing se enfocan en tratar de evitar que las credenciales de acceso de un usuario a una determinada web caigan en manos de los atacantes aun cuando la víctima del ataque haya accedido a un sitio malicioso que suplante a uno genuino para robarle sus datos de acceso. Este tipo de sistemas de prevención se basan en mejorar la seguridad en el proceso de acceso a determinados servicios digitales a través de implementar medidas de seguridad adicionales.

Actualmente podemos dividir este tipo de mecanismos de seguridad en dos grandes grupos principales: los sistemas de autenticación de doble factor y los gestores de contraseñas.

Sistemas de autenticación de doble factor

La autenticación de doble factor es una herramienta que ofrecen varios proveedores de servicios a través de internet y cuyo objetivo es agregar una capa de seguridad adicional al proceso de inicio de sesión. Para lograrlo, una vez que el usuario introduce sus credenciales de acceso para acceder a su cuenta, si tiene activado el sistema de autenticación en dos pasos, será necesario que Introduzca un segundo elemento de identificación (generalmente un Código aleatorio con un periodo de validez de unos segundos). Por norma general, este código se le envía al usuario a un teléfono móvil que tenga previamente registrado o si no a un correo electrónico o a través de algún tipo de aplicación para smartphone como la que utiliza Google para ofrecer este servicio en sus servicios (*Google Authenticator*).

En esencia este mecanismo de seguridad obliga a que el usuario de un determinado servicio de internet deba recordar una contraseña y estar en posesión de un código de acceso o de un dispositivo hardware como una llave USB para poder acceder a su cuenta de usuario. Se consigue de esta manera aumentar la seguridad en los accesos a los servicios web que lo implemente ya que el mecanismo tradicional de seguridad de acceso, las contraseñas, suele perder su efectividad debido a que por su naturaleza cuanto más segura es una contraseña más difícil es de recordar por lo que al final los usuarios acaban utilizando contraseñas sencillas o incluso las repiten para muchos sitios web con pierde mucha eficacia como métodos de seguridad. Un atacante que desee robar las credenciales de acceso a un servicio que implemente este tipo de medidas no tendrá solo que conseguir la contraseña y el identificador del usuario si no que deberá hacerse también con el control del segundo factor de autenticación. En el caso de que sea un dispositivo hardware tendrá que hacerlo personalmente y si lo que se utiliza es un código estos tienen un periodo de validez temporal que los inhabilita.

Este tipo de sistemas de autenticación, sin embargo, siguen siendo vulnerables a los ataques de tipo *Man In the Middle* como se ha comentado con anterioridad ya que los atacantes pueden hacerse con el token de autenticación a través de por ejemplo una extensión maliciosa del navegador de la víctima. Algunos de las medidas propuestas por distintos autores son el uso de códigos QR que se intercambian entre el sitio web y el usuario para validar el token de autenticación y además validar que el navegador desde el que se accede y el smartphone del usuario se encuentran en la misma ubicación a partir de la información asociada a la dirección IP de ambos. Sin embargo, ninguno de estos sistemas presenta una solución completa puesto que como se sabe, las direcciones IP son susceptibles de ser modificadas.

Una forma más segura de este tipo de sistemas de autenticación de doble factor son los que en vez de obligar al usuario a introducir un código extra además de la contraseña utilizan un dispositivo hardware que generalmente consiste en una unidad USB. Este tipo de dispositivos funcionan de la siguiente manera: El usuario introduce sus credenciales de acceso (usuario y contraseña) y el dispositivo los cifra utilizando una clave secreta que se almacena en el propio dispositivo y en el servidor. La información cifrada del usuario se envía al servidor y este valida el proceso de autenticación y termina el proceso generando un código de autenticación valido por un periodo de tiempo limitado utilizando una semilla y una referencia de un instante de tiempo. Cuando después el servidor al que se quiere acceder recibe las credenciales de acceso, recibe también ese código y verifica que sea válido a partir de la semilla asociada al usuario y con el valor del reloj.

Algunos de los principales sistemas comerciales que existen en la actualidad son *Yubikey, RSA SecurID o DUO*.

A pesar de todas las ventajas que ofrecen, estos sistemas presentan algunos inconvenientes:

- En primer lugar, el usuario tiene que comprar y llevar con él un nuevo dispositivo para autenticarse en los sitios web. Esto supone una barrera para muchos usuarios al suponer un coste extra y que al tratares de un dispositivo físico debe estar en todo momento en posesión del usuario si quiere poder acceder al servicio que protege.
- Algunos de estos tokens utilizan sistemas de claves que pueden espiados a través de extensiones maliciosas que se instalan dentro de los navegadores de los usuarios. En el caso de que un atacante consiga que la víctima instale una extensión maliciosa en su navegador, podrá interceptar las claves que se intercambian entre el cliente y el servidor

donde se gestiona la legitimidad de los accesos de usuario a un determinado servicio digital.

- Alguno de estas llaves puede verse comprometidos mediante técnicas de Ingeniería inversa. Al tratares de unos pocos sistemas comerciales disponibles en el mercado todos basados en protocolos de seguridad privados, los atacantes pueden realizar técnicas de ingeniería inversa para averiguar el funcionamiento de los mecanismos de seguridad implementados y explotarlos en su beneficio sin que a la empresa que vende estos dispositivos le llegue a interesar nunca que se conozca que sus dispositivos se han visto comprometidos.
- Solo funcionan con los servicios de internet que lo tengan configurado por lo que para poder ser utilizados para hacer más seguro un servicio web en particular, este debe de haber sido preconfigurado primero por el suministrador del propio dispositivo.

Gestores de contraseñas

Por último, existe otro mecanismo de seguridad que se centra en mejorar la seguridad del primer factor de autenticación de acceso a un portal web, la contraseña de usuario. Estos sistemas son conocidos como gestores de contraseñas y se encargan de generar y almacenar contraseñas robustas y fiables para los sitios web que visite un usuario. La mayoría de estos sistemas almacenan las contraseñas encriptadas en el navegador del usuario y rellenan los formularios de manera automática cuando un usuario accede a alguno de los sitios con las credenciales de acceso almacenadas. La principal ventaja de estos programas es la de que son capaces de generar contraseñas únicas, largas, complejas y muy difíciles de averiguar mediante técnicas de fuerza bruta. Para acceder a las contraseñas que gestiona el programa, el usuario debe recordar una única contraseña maestra que le permite acceder al sistema de gestión de contraseñas. La mayoría de estos programas incluyen sistemas de relleno automático de formularios lo que permite que sea el propio programa el que reconozca la dirección URL de un servicio web frecuentado por el usuario y rellene automáticamente los datos de acceso, evitando así que se acceda a una página false durante un ataque de phishing. El hecho de que estos sistemas almacenen una gran cantidad de contraseñas de usuario los convierte en un objetivo muy codiciado por los atacantes por lo que deben implementar fuertes medidas de seguridad para evitar que la información que manejan caiga en las manos equivocadas.

Estos sistemas de gestión de contraseña se pueden dividir en

- Aplicaciones instaladas de manera local en el equipo de un usuario. La información protegida se encuentra en posesión del propio usuario por lo que la responsabilidad de su cuidad recae sobre el propio usuario,
- Servicios disponibles a través de internet. El usuario de este tipo de sistemas no tiene preocuparse de mantener a salvo la base de datos, pero tampoco tiene el control total sobre ella puesto que se almacena en el ordenador del proveedor del servicio de gestión de contraseñas.
- Dispositivos hardware accesibles de manera local y física. Son muy parecidos a las aplicaciones instaladas en el entorno local del usuario, pero en este caso son productos físicos como algún tipo de dispositivo USB y tienen un coste extra asociado a la compra del propio hardware.

Todos estos diferentes tipos de gestores de contraseñas almacenan las credenciales de usuario en una base de datos encriptada utilizando para ello una contraseña maestra que es la que el

usuario debe recordar si quiere acceder a la información de sus cuentas. Por este motivo, el usuario es responsable de la seguridad de dicha clave maestra lo que sin duda supone un punto vulnerable de estos sistemas ya que si se obtiene acceso a la máquina del usuario se puede obtener dicha contraseña mediante la instalación de algún programa que guarde las teclas pulsadas por el usuario (conocidos como *Key Loggers*). Además, tenemos que confiar que la generación aleatoria de contraseñas que realizan estos programas sea lo suficientemente robusta para que estas no se puedan averiguar mediante procesos automáticos de fuerza bruta.

Entre los principales gestores de contraseña que podemos encontrar en la actualidad tenemos *LastPass, 1Password, Dashlane, KeePass*, etc.

Dentro de la literatura académica relacionada con los sistemas de prevención del phishing podemos encontrar algunas soluciones muy interesantes para implementar medidas de prevención de estos ataques como el que se define en [VashneyMisraPradeep2018].

La solución propuesta en este artículo consiste en la creación de un sistema de seguridad basado en el uso de un smartphone, una aplicación Android y un ordenador con conectividad Bluetooth en el que se ha instalado una aplicación para el navegador Chrome. Con el objetivo de dificultar los posibles ataques el sistema propuesto rellena automáticamente la información de nombre de usuario necesaria para rellenar un formulario de registro/acceso a una web con la dirección Bluetooth del teléfono haciendo que sea necesario para el atacante estar en el radio de alcance del área de la red personal (PAN) de la víctima. De esta manera se enlaza el dispositivo Bluetooth con el ordenador del usuario evitando así los ataques de tipo *Man in the Middle* en tiempo real y los de transferencia de control. En el proceso de acceso a un sitio web protegido con este sistema, el usuario deberá introducir únicamente la contraseña y la propia aplicación será la encargada de suministrar el token del segundo factor de autenticación. Para garantizar la movilidad del sistema, lo que plantea el artículo es que el usuario lleve consigo el teléfono móvil y que acceda a los sitios web a través de la aplicación desarrollada para el navegador Chrome por lo que tendrá que conectarse con su cuenta de usuario de Google para que se sincronicen los datos almacenados en un nuevo ordenador que se quiera utilizar para autenticarse en un sitio web.

Si analizamos en detalle la solución que proponen los autores, podemos distinguir dos fases bien distintas, la fase de registro y la fase de acceso:

Fase de registro

En esta fase, un usuario registra su cuenta en el sitio web desarrollado para la prueba de concepto introduciendo su nombre de usuario, su contraseña, su correo electrónico y su número de teléfono móvil en un formulario web seguro accedido desde la aplicación Android desarrollada.

En ese momento, la dirección Bluetooth del dispositivo Android utilizado para el registro se envía al servidor web junto con el resto de información del usuario a través de una conexión realizada mediante el protocolo cifrado de comunicación web HTTPS como se muestra en la figura 6.

Figura 6. Registro de usuario em Android publicado en [VashneyMisraPradeep2018].

Cuando el servidor recibe toda la información envía al móvil del usuario una contraseña de un solo uso al teléfono móvil que acaba de registrar para este nuevo usuario.

A continuación, el usuario introduce la contraseña de un solo uso que ha recibido en el formulario de registro y el servidor la valida creando una cuenta en el sistema. En este paso, el servidor genera un identificador único y secreto asociado a la aplicación Android desde la que se ha registrado el usuario, así como un código aleatorio conocido como Salt con el que realiza el cifrado del identificador de la aplicación Android y se lo envía devuelta al dispositivo móvil del usuario a través de HTTPS.

La aplicación Android cifra el valor del identificador de aplicación recibido del servidor utilizando como clave la concatenación de la contraseña de usuario que hubiera introducido el usuario y el valor del Salt recibido del servidor y el resultado lo almacena en la memoria local del smartphone junto con el valor del identificador de aplicación Android cifrado con la clave Salt recibida del servidor. Por su parte, el servidor almacena el nombre de usuario, su contraseña, su correo electrónico, su número de teléfono móvil, su dirección de bluetooth, el identificador de su aplicación Android y el valor del Salt.

Para asegurar que el proceso de registro del usuario ha funcionado correctamente, el dispositivo Android del usuario envía un mensaje de OK cifrado con el identificador de la propia aplicación Android y si el servidor recibe un mensaje de OK cifrado correctamente marca el proceso de registro como un éxito y avisa a la aplicación Android.

Durante este proceso de registro, la contraseña y el código Salt utilizado para el registro no han sido almacenados en ningún momento en la aplicación Android para aumentar la seguridad del sistema.

Podemos resumir el proceso de registro planteado en el artículo de [VashneyMisraPradeep2018] de la siguiente manera:

1. El usuario introduce sus datos personales en un formulario.
2. La información del usuario junto con la dirección bluetooth del dispositivo Android se envían al servidor a través de HTTPS.

3. El servidor contesta con una contraseña de un solo uso que se envía al número de teléfono del usuario.
4. El usuario introduce la contraseña de un solo uso en el formulario de registro y el servidor la valida.
5. Una vez validada la contraseña temporal, el servidor crea un identificador único y secreto para la aplicación Android asociada al usuario, así como un código Salt con el que cifra el identificador anterior enviándoselo todo al dispositivo del usuario a través de HTTPS.
6. La aplicación Android cifra el identificador recibido utilizando la concatenación de la contraseña de usuario y el valor del Salt recibido y almacena el resultado de manera local. También se almacena el identificador de la aplicación cifrado solo con el valor del Salt.
7. El servidor almacena el nombre, contraseña, correo electrónico, número de teléfono móvil, dirección de bluetooth, identificador de la aplicación Android y el valor del Salt.
8. La aplicación cifra un mensaje de OK utilizando el identificador único de aplicación y si es correcto el servidor contesta con un mensaje de éxito que termina el proceso.

La figura 7 muestra un resumen de los pasos necesarios descritos durante la fase de registro en el sistema:

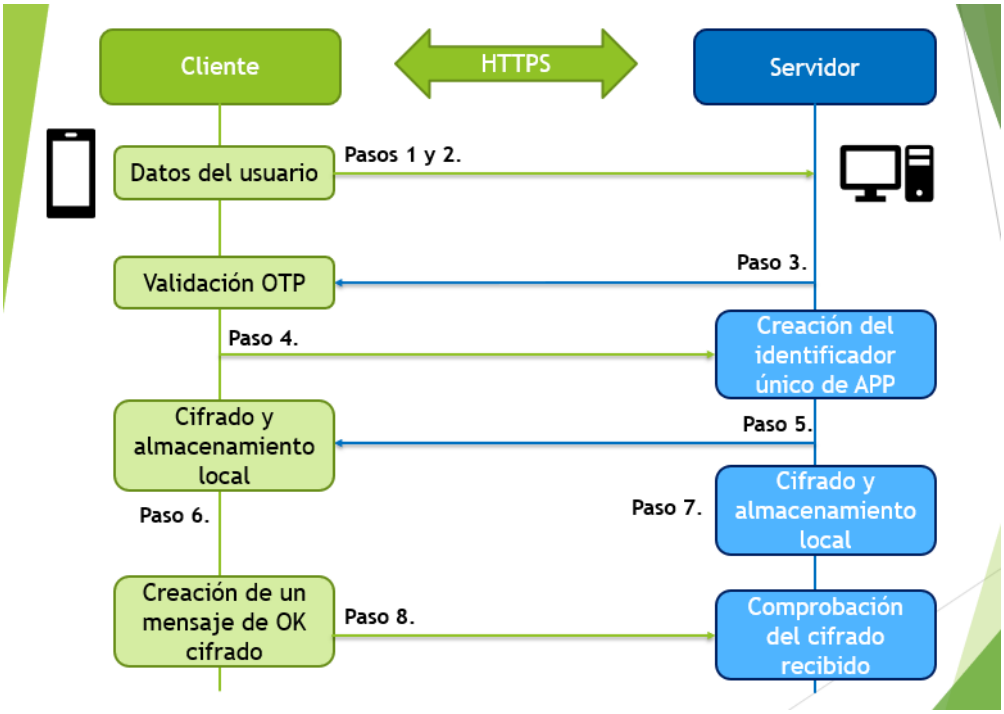

Figura 7. Flujo resumen de la fase de registro de la aplicación Android de [VashneyMisraPradeep2018].

Fase de acceso

En este punto el sistema planteado por los autores diferencia dos posibles escenarios: que el acceso del usuario se realice a través del propio dispositivo Android donde se encuentra instalada la aplicación del mecanismo presentado o bien, que el acceso se realice desde un ordenador a través del uso de un navegador web de escritorio, más en concreto desde el navegador Chrome.

Primero vemos como se ha diseñado la fase de acceso desde smartphone. En esta etapa el usuario solo tiene que introducir su contraseña de acceso en la aplicación Android. La aplicación móvil recupera de su almacenamiento local el valor del identificador de la aplicación Android único que se cifró durante el proceso de registro utilizando la contraseña de usuario y el valor Salt del servidor. Esta información se descifra usando la contraseña de usuario y se envía junto con la dirección bluetooth del dispositivo al servidor a través de HTTPS.

Ahora el servidor se encarga de identificar al usuario y de validar si la petición de acceso se ha realizado desde la aplicación Android que tiene registrada el usuario. Para esto, el servidor cifra el identificador de la aplicación Android que tiene almacenado para este usuario (identificado por un id de Bluetooth) y utiliza como clave el valor que ha recibido recientemente junto con el identificador bluetooth para luego descifrarlo mediante la contraseña de usuario almacenada en su base de datos con el objetivo de genera una copia local del valor que le acaba de enviar el usuario. De esta manera puede confirmar su identidad si el valor calculado en el servidor es igual al valor que le envía el cliente. En caso afirmativo, el servidor genera un nuevo valor del código identificador de la aplicación Android utilizada por el cliente y se lo envía de vuelta siendo cifrado con el valor antiguo del identificador de aplicación. Además, el servidor envía el valor de Salt utilizado y el valor del nuevo identificador de aplicación cifrado con ese valor de Salt.

En este momento, la aplicación Android que ejerce el rol de cliente utiliza el valor Salt que acaba de recibir junto con la contraseña de usuario para descifrar el valor del identificador de aplicación que tenía almacenado en su memoria local. Una vez obtenido este valor, lo utiliza para descifrar el valor del nuevo identificador de aplicación que le acaba de enviar el servidor. Además, el valor del cifrado por la concatenación de contraseña y valor Salt del identificador de aplicación que había sido almacenado en memoria local se sustituye por el nuevo identificador de aplicación cifrado de la misma manera y que será utilizado en el siguiente inicio de sesión. También se remplaza en memoria el valor del identificador de aplicación cifrado únicamente con el Salt recibido que será utilizado en el proceso de acceso a través del ordenador.

A continuación, la aplicación Android cifra la contraseña de usuario utilizando el identificador de aplicación de la sesión actual (el valor original) y se lo envía al servidor a través de HTTPS. El servidor descifra esta contraseña con el identificador de aplicación antiguo que tenía asociado al usuario y si la contraseña descifrada coincide con la almacenada el intento de acceso resultará satisfactorio.

Por último, el servidor remplaza el valor del identificador de aplicación asociada al usuario con el nuevo valor generado para que sea utilizado en la siguiente sesión.

Podemos resumir todo el proceso en los siguientes pasos:

1. El usuario introduce su contraseña de acceso en la aplicación.
2. La aplicación envía al servidor la dirección bluetooth del dispositivo junto con el valor almacenado y cifrado del identificador de aplicación que previamente se ha descifrado utilizando la contraseña del usuario.
3. El servidor realiza el mismo proceso de cifrado y descifrado del identificador de aplicación asociado a la dirección bluetooth del usuario y compara el valor calculado con el que acaba de recibir.
4. Si los valores coinciden, el servidor genera un nuevo identificador de aplicación lo cifra con el valor anterior de identificador de usuario y con el valor de Salt. Estos dos valores junto con el valor de Salt de envían de vuelta al cliente.
5. La aplicación descifra el identificador de aplicación que tenía almacenado localmente utilizando la contraseña de usuario y el valor Salt. Con este valor descifrado, puede también descifrar el nuevo identificador de aplicación recibido. En esta etapa, la aplicación sustituye en memoria la información del identificado de aplicación por el nuevo valor.
6. La aplicación cifra la contraseña de usuario con el valor del identificador de aplicación original y se lo envía al servidor.
7. El servidor comprueba el valor recibido comparándolo con la información que tiene almacenada y si coincide autoriza el acceso del usuario.
8. Por último, el servidor remplaza el valor del identificador de aplicación asociado al usuario para el siguiente intento de inicio de sesión.

La figura 8 muestra el protocolo planteado para el proceso de acceso desde un smartphone:

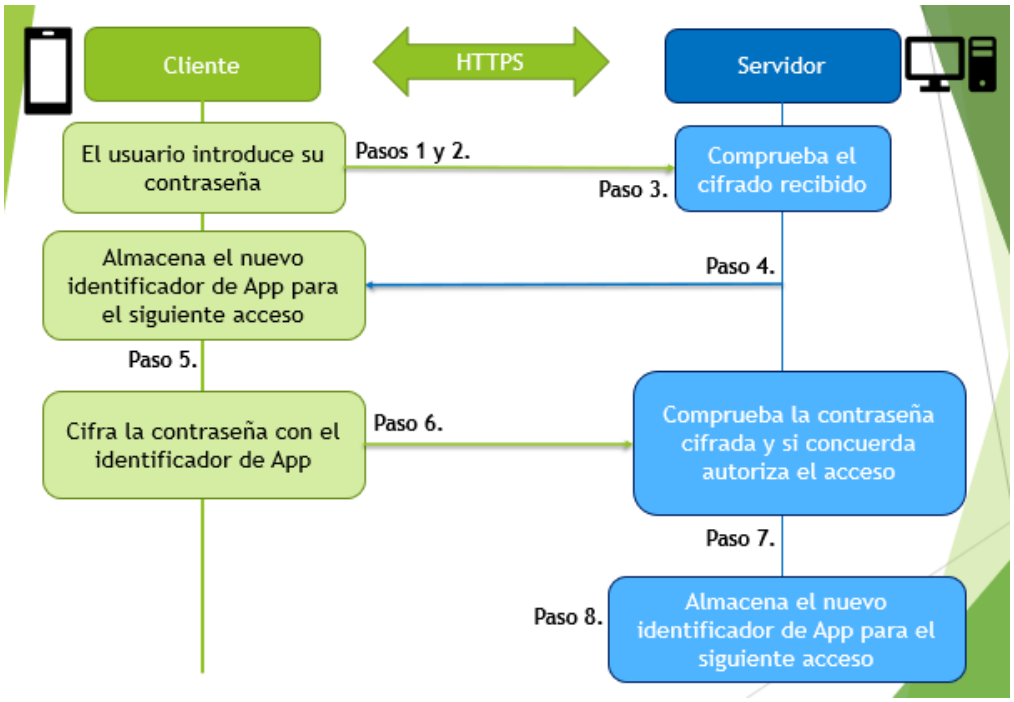

Figura 8. Proceso de acceso seguro del artículo de [VashneyMisraPradeep2018].

Después de haber visto el proceso de registro y acceso desde un smartphone vamos a ver cómo han planteado los autores del artículo el acceso desde un ordenador personal. Esta parte tiene

como requisito el que el usuario se descargue e instale en su ordenador una aplicación desarrollada para el navegador Google Chrome y que el proceso de registro a través del Smartphone de usuario se haya realizado con anterioridad.

Cuando el usuario ejecuta la aplicación en Chrome le aparecerá una pantalla de registro. En ese momento, deberá conectar su dispositivo móvil con su ordenador a través de la conectividad bluetooth y la aplicación de Chrome enviará al servidor el identificador bluetooth del dispositivo junto con la contraseña de acceso que el usuario ha introducido en la pantalla de registro de su navegador de escritorio.

El servidor valida los datos recibidos y envía una contraseña de un único uso al número de teléfono del usuario para que este la introduzca en la aplicación Chrome y la envíe devuelta al servidor donde se validará.

En este punto el servidor generará un identificador único para la aplicación Chrome del usuario y se lo envía al cliente junto con el valor Salt utilizado. La aplicación Chrome cifra el identificador recibido utilizando como clave la concatenación de la contraseña de usuario y el valor Salt recibido y almacena el resultado en el almacenamiento seguro de la propia aplicación Chrome.

La aplicación Chrome envía un mensaje predefinido de OK utilizando el identificador de aplicación Chrome de usuario como clave y el servidor lo verifica y contesta con el mismo mensaje si todo es correcto o con un mensaje de error en caso contrario. Además, el servidor almacena localmente el identificador único de aplicación Chrome asociado al usuario.

En este punto el proceso de registro de la aplicación Chrome ha terminado para este usuario. Todo este proceso de registro solo se realiza la primera vez que se accede con un navegador Chrome y si el usuario quisiera acceder desde otro ordenador tendría que iniciar sesión primero en el navegador con su cuenta de Google y se sincronizarían sus datos de la aplicación Chrome de sistema de prevención de phishing planteado.

Podemos resumir el flujo de registro de la aplicación para el navegador Chrome en los siguientes pasos:

1. El navegador Chrome del cliente envía la contraseña de acceso del usuario junto con el identificador bluetooth del dispositivo Android registrado.
2. El servidor valida los datos y envía una OTP al teléfono móvil del cliente.
3. El cliente introduce la OTP en su navegador y la envía devuelta al servidor.
4. El servidor genera un identificador único para la aplicación Chrome del usuario y se lo envía al cliente.
5. El cliente cifra el identificador recibido utilizando la concatenación de la clave de usuario y el valor Salt que le ha enviado el servidor y lo almacena en la memoria privada de la aplicación Chrome.
6. El cliente envía un mensaje de OK cifrado con el identificador recibido-
7. El servidor valida que el mensaje este bien cifrado y almacena el identificador de aplicación Chrome junto con los datos del usuario registrado.

La figura 9 muestra el flujo del proceso de registro inicial de la aplicación Chrome según el algoritmo presentado:

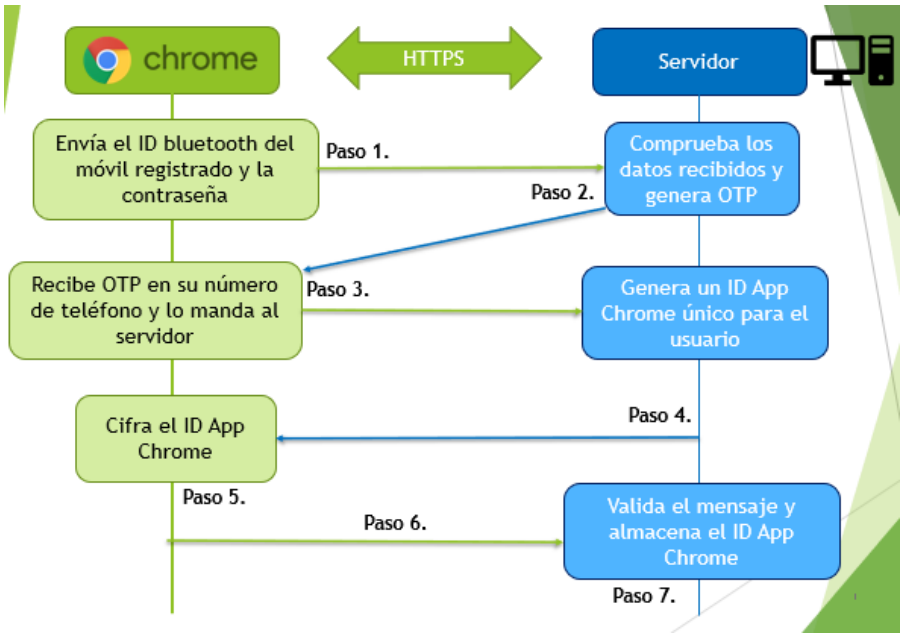

Figura 9. Proceso de registro inicial de usuario en la aplicación Chrome de [VashneyMisraPradeep2018].

Una vez registrada la aplicación Chrome vamos a ver como se ha planteado el acceso desde la misma. Para empezar, el usuario conecta mediante bluetooth su teléfono Android con su ordenador. Una vez emparejados, abre la aplicación Android y selecciona la opción de "Acceso desde Ordenador". La aplicación Android recupera de la memoria el valor del identificador de aplicación del usuario cifrado con el valor Salt como clave y lo escribe en la propiedad nombre de la red bluetooth del dispositivo y establece un tiempo de 60 segundos para reiniciar al valor original del nombre del adaptador bluetooth del dispositivo.

A continuación, el usuario abre la aplicación Chrome e introduce su contraseña de usuario. La aplicación Chrome obtiene los valores nombre y dirección del adaptador bluetooth del dispositivo móvil emparejado. La dirección del adaptador se envía al servidor junto con el valor almacenado del identificador de aplicación Chrome que se había encriptado utilizando la contraseña de usuario y el valor Salt. Este último valor se descifra antes de ser enviado utilizando como clave la propiedad nombre del dispositivo bluetooth vinculado que este momento está formado por el valor del identificador de aplicación Android cifrado con el valor Salt.

Con los dos valores que acaba de recibir el servidor por parte del navegador Chrome del usuario es posible identificarlo asegurándose que la petición del valor Salt proviene del usuario autentico. Después de la validación del usuario, el servidor envía un nuevo valor de identificador de aplicación Chrome que será utilizado en el siguiente intento de inicio de sesión del usuario. Este nuevo identificador se envía cifrado con el valor original del identificador de aplicación Chrome y el valor Salt correspondiente al usuario. Además, se envía un token de sesión valido para una única vez y con un periodo de validez temporal corto que también está cifrado con el identificador de la aplicación Chrome de la sesión actual.

En este momento, la aplicación Chrome utiliza la contraseña de usuario y el valor Salt recibido para descifrar el valor del identificador de la aplicación Chrome que tenía almacenado en la

memoria local. Con este valor, puede descifrar el nuevo identificador de aplicación Chrome recibido que se utilizará en el siguiente intento de acceso por lo que procederá a almacenarlo en memoria de manera encriptada sustituyendo al valor anterior.

A continuación, la aplicación descifra la clave temporal recibida utilizando como clave el valor de identificador de aplicación Chrome y crea una petición GET a través de HTTPS a la página web final pasando como parámetros por un la clave temporal y por otro la contraseña de usuario junto con la dirección bluetooth del dispositivo del usuario cifrados con el valor del identificador de la aplicación Chrome.

La aplicación crea una nueva pestaña en el navegador del usuario utilizando la URL creada en el paso anterior y la comunicación se realiza a partir de ahora entre el navegador y el servidor. Por último, el servidor analiza la clave temporal recibida a través la petición HTTPS y si la clave es válida y se corresponde con el usuario que ha realizado la petición utiliza el identificado de aplicación Chrome para descifrar el otro parámetro recibido vía HTTPS obteniendo como resultado la dirección bluetooth y la contraseña del usuario. En este punto si se validan los datos recibidos en el servidor, se concede el acceso y los datos de la página web ser devuelven en la respuesta al navegador del usuario que inicio el proceso de acceso.

Podemos resumir el proceso de acceso desde un navegador Chrome de la siguiente manera:

1. El usuario debe instalar una aplicación para el navegador Chrome y emparejar su dispositivo Android registrado con el ordenador desde el que quiere acceder.
2. El proceso de registro de la aplicación Chrome solo se realiza una única vez y se deberá realizar después del registro del dispositivo Android.
3. Durante el proceso de registro de la aplicación Chrome se intercambian varios mensajes entre la aplicación y el servidor y se genera una clave única de identificación de la aplicación Chrome de usuario.
4. Cuando el usuario quiere realizar el acceso desde su navegador deberá haber iniciado sesión en el navegador Chrome utilizando su cuenta de Google e introduciendo su contraseña de acceso a través de la aplicación Chrome instalada.
5. La aplicación intercambia varios mensajes cifrados con el servidor donde se incluye la generación y el almacenamiento de un nuevo identificador de aplicación Chrome que se usara en el siguiente intento de inicio de sesión.
6. Para la comunicación entre el servidor y el navegador web del usuario se utiliza la dirección y el nombre del adaptador bluetooth del dispositivo Android registrado del usuario.
7. El servidor genera una clave temporal y de un solo uso una vez que se han validado los datos del usuario.
8. Una vez validados los datos de acceso, la comunicación se realiza entre el navegador de usuario y el servidor a través del protocolo HTTPS.

El sistema de autenticación presentado es eficaz contra los últimos tipos de ataques de phishing conocidos de tipo *Man in the Middle* de tiempo real, con transferencia de control o través de extensiones malignas instaladas en el navegador del usuario. Las principales ventajas que tiene este método de autenticación son:

1. Reduce el número de credenciales que el usuario tiene que recordar y que introducir durante el proceso de acceso a un servicio online evitando de esta manera que un ataque de tipo *Man in the Middle* se pueda llevar a cabo con toda la información necesaria para suplantar a un usuario.
2. Al utilizar una aplicación Android y una aplicación Chrome para que el usuario introduzca la contraseña, las extensiones de navegador maliciosas n tienen acceso a la misma.
3. El uso de un identificador único asociado al dispositivo Android del usuario y la necesidad de contar con acceso físico al dispositivo hace prácticamente imposibles los ataques de tipo *Man in the Middle* con transferencia de control.

2.5. Sistemas de detección de phishing

Los sistemas de detección de phishing tienen la misión de identificar cuando un recurso digital es legítimo o no. Por lo general, los recursos sobre los que deben decidir son principalmente de dos tipos, correos electrónicos o páginas webs accesibles a través de internet.

El enfoque tradicional de este tipo de sistemas de detección está basado en listas blancas o negras:

- Las listas blancas son listas en las que se almacenan las direcciones de correo electrónico de los remitentes conocidos de un usuario o por ejemplo las direcciones URL de los sitios web que son frecuentadas por dicho usuario. De esta manera cuando se recibe un correo electrónico de alguno de los contactos que forman parte de la lista blanca, el software encargado de la gestión del correo electrónico lo identificará como un correo legítimo. De la misma manera, el navegador web consultará con la lista blanca la dirección URL que el usuario desea visitar.
- Las listas negras tienen un funcionamiento contrario al de las listas blancas. Estas almacenan las direcciones web o las cuentas de correo electrónico que hayan sido identificados como phishing o maliciosos para alertar al usuario cuanto vaya a interactuar con alguna de ellas.

Por lo general las listas negras suelen ser de ámbito más general con el objetivo de que una comunidad mayor esté alertada de los posibles sitios maliciosos que existen en internet mientras que las listas blancas son privadas y se almacenan asociadas a una única cuenta de usuario.

Dado el volumen de tráfico que se ha alcanzado en las redes de datos actuales como Internet, resulta prácticamente imposible mantener actualizadas este tipo de listas por lo que de por sí mismas no suponen un mecanismo de seguridad eficaz para frenar los ataques de phishing. Además, las técnicas que utilizan los ciberdelincuentes han evolucionado hasta tal punto que hoy en día para el proceso de creación de un sitio web malicioso existen conjuntos de herramientas disponibles que automatizan el proceso de suplantación de sitios web (como pueden ser *Super Phisher* o *Rock Phish*)

Por estos motivos ha sido necesario crear otro tipo de mecanismo de seguridad complementario que permita detectar en tiempo real cuando una página web o un correo electrónico es potencialmente peligroso para un usuario. Podemos entender este problema como un problema de clasificación, por lo que parece lógico pensar que lo que se necesita es algún tipo de algoritmo de clasificación que permita identificar sitios web y correos electrónicos como potencialmente maliciosos o legítimos. Para este tipo de problemas, los algoritmos de *Machine Learning* suelen ser muy efectivos y podemos encontrar numerosos ejemplos en recientes publicaciones académicas donde se obtienen resultados cercanos al 99,8% de efectividad en lo que a la detección de páginas web maliciosas se refiere en [GowthamKrishnamurthib2014].

Arquitectura del sistema planteado

En este artículo, los autores describen un sistema de detección de phishing basado en técnicas de *Machine Learning*. En primer lugar, se crea una lista blanca de páginas controlada y mantenida por el usuario donde se almacenan las direcciones web que suele frecuentar y que forman parte de una navegación segura entre sitios de confianza. Si el usuario desea visitar un sitio web cuya dirección URL no se encuentra dentro de dicha lista blanca el sitio web al que pertenece la URL se analiza mediante un algoritmo siempre que este tenga algún tipo de formulario de acceso susceptible de filtrar información sensible. En caso afirmativo, el sistema descrito procede a obtener 15 características principales del sitio web a partir de las heurísticas que han definido previamente los autores. Para agilizar el proceso de extracción de características, el sistema que han desarrollado trabaja con distintas agrupaciones de heurísticas de forma paralela.

Una vez obtenido el vector de características se envía a un clasificador que previamente ha sido entrenado con los vectores de características que han sido generados para todos los sitios del conjunto de datos de entrenamiento.

La figura 10 contiene un diagrama donde se detalla el funcionamiento planteado por los autores en [GowthamKrishnamurthib2014].

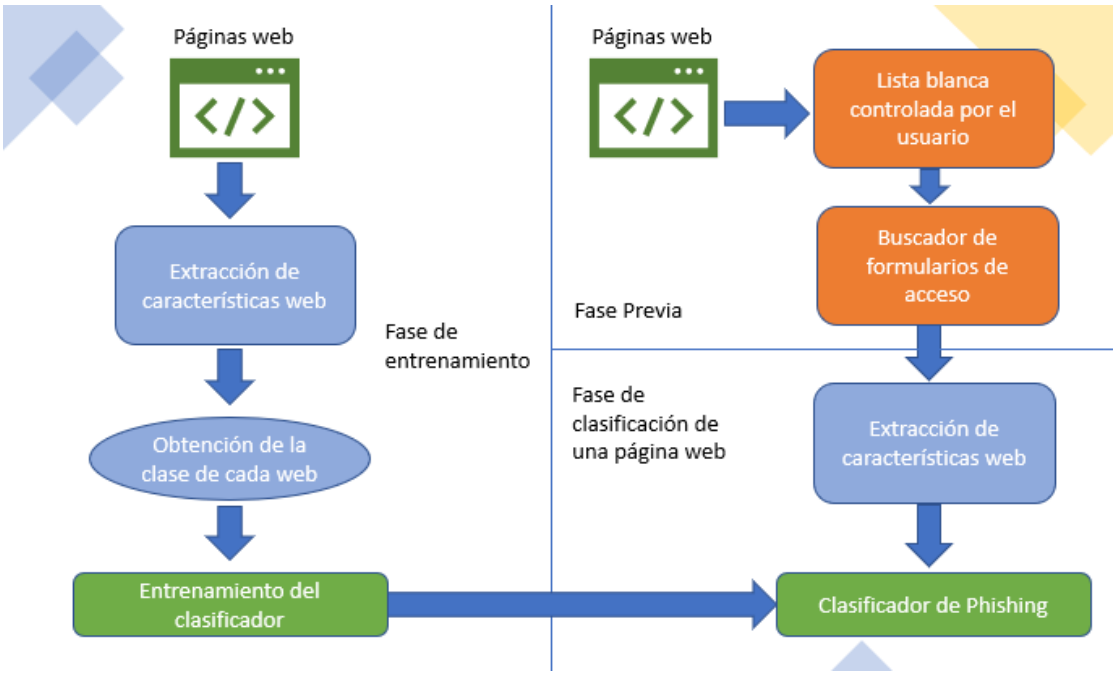

Figura 10. Arquitectura del sistema de identificación de phishing propuesto en [GowthamKrishnamurthib2014].

Una vez visto a grandes rasgos la arquitectura del sistema de detección planteado por los autores del artículo vamos a entrar en detalle a ver cómo se puede implementar un algoritmo de *Machine Learning* que sea eficaz para resolver este problema.

Lista de sitios web controlada por el usuario

Se trata de una lista privada que contiene la dirección UL de los sitios webs que gozan de la confianza del usuario, así como las direcciones IP de estos. En el momento de añadir una nueva dirección IP a la lista obtenidos de un servidor DNS local, esta se comprueba con al menos dos servidores DNS públicos gestionados por empresas de confianza como Google o Norton para asegurar que la dirección IP que se añade a la lista es legítima. De esta manera se protege al usuario de un ataque de tipo *pharming* conocidos en castellano como ataques de "*farmeo"* donde se intercepta el servidor DNS local para que redirija las peticiones de URL del usuario hacia servidores comprometidos.

Cuando un usuario accede a una página web, lo primero que hace el sistema es comprobar si el par URL - dirección IP del sitio se encuentran dentro de la lista blanca. De ser así se considera una navegación legítima y termina el mecanismo de identificación de phishing. En caso contrario se pasa al siguiente modulo del sistema, el buscador de formularios de inicio de sesión.

Buscador de formularios de inicio de sesión

Esta etapa del procedimiento se encarga de buscar formularios de inicio de sesión dentro de un sitio web al cual el usuario quiere acceder y que no se encuentra registrado en su lista blanca. En caso de que el sitio web no tuviera ningún formulario, el proceso de detección de phishing se detendría puesto que no hay posibilidad de que las credenciales de acceso fueran comprometidas por un atacante. El algoritmo de detección de formularios funciona de la siguiente manera:

1. Recorre el árbol DOM de la página HTML como entrada del algoritmo buscando etiquetas de tipo input de entrada de datos y de tipo formulario y devuelve un 1 si considera que la pagina contiene algún formulario de acceso y un 0 si no lo tiene.

   Para cada uno de los formularios que existan dentro del HTML analizado realiza las siguientes acciones:

   a. Si el formulario analizado contiene alguna palabra reservada destinada a la búsqueda se interpreta como un formulario de búsqueda y se pasa a analizar el siguiente formulario. Si no lo encuentra continua con el paso siguiente.
   b. Busca alguna de las palabras reservadas destinadas a los formularios de acceso. Si encuentra alguna, considera que se trata de un formulario de acceso y termina el proceso devolviendo el valor 1. En caso contrario, continua con el paso siguiente.
   c. Realiza la misma búsqueda, pero en los dos niveles superiores del árbol DOM desde donde se encuentra el formulario actual. Si encuentra en este proceso alguna de las palabras reservadas asociadas al proceso de acceso, devuelve 1 sino continua con el paso siguiente.
   d. Si en el formulario analizado o en su alcance solo encuentra imágenes entonces devuelve un 1 ya que es una técnica habitual de los cibercriminales utilizar imágenes en lugar de texto para evitar ser detectados por las herramientas

automáticas de detección de phishing. Si no es así continua con el paso a para analizar el siguiente formulario y en caso de que no hubiera más formularios devolvería un 0.

2. Si el DOM del HTML analizado solo contiene etiquetas de tipo input de entrada y no formularios:
    a. Busca en la lista de palabras reservadas destinadas al proceso de acceso en el total del árbol DOM. Si encuentra alguna devuelve un 1 y si no continua con el paso siguiente.
    b. Si solo encuentra imágenes en el árbol DOM devuelve un 1 en caso contrario devuelve un 0.
3. Si el DOM del HTML analizado no contiene etiquetas de formulario ni de entrada de datos el algoritmo devuelve un 0.

Generador de características de un sitio web

Este es uno de los procesos claves del sistema que han planteado los autores del artículo ya que es donde se va a generar el vector de características que posteriormente se analizara en el clasificador para identificar un sitio como legitimo o sospechoso de ser phishing.

El proceso de extracción de características se divide en grupos independientes que funcionan de manera concurrente a través de diferentes hilos de ejecución paralelos y que en función de las heurísticas de cada hilo generarán vectores de características locales que serán procesados y unificados en un módulo destinado a sincronizar los vectores locales en un único vector de características.

A continuación, vamos a ver en que consiste cada uno de los módulos encargado de obtener las características fundamentales de un sitio web:

1. **Extracción de identidad.**
   Este paso obtiene dos tipos de identidad de una web, la identidad de URL y la de palabras clave.
   La identidad de URL de una página web se determina analizando su estructura de hipervínculos. En las páginas legítimas, la mayoría de los enlaces apuntan al propio dominio de la web analizada o hacia algún dominio asociado. Solamente se consideran los valores contenidos dentro de los valores *<href>* y *<src>* de cada etiqueta HTML de tipo *<a>, <area>, <link>, <img>* y *<script>.* Para cada una de las direcciones encontradas, se extrae el dominio base de la web analizada y se calcula el número de veces que el dominio base aparece en la página siendo el que mayor frecuencia tenga el que se considerara como la identidad de la URL de la página web.
   La identidad de palabras consiste en el conjunto de palabras que tienen relevancia dentro de la propia página y que pueden ser descriptivas de la identidad del sitio web. Las palabras clave se obtienen de inspeccionar las siguientes etiquetas del árbol DOM: *title, meta, meta description* y el valor de las propiedades *alt, title* y *body* de todas las etiquetas del árbol DOM. Para extraer el conjunto de las palabras clave se utiliza el método TF-IDF.

2. **Evaluaciones basadas en los elementos de tipo formulario.**
   En este paso existen dos heurísticas diferentes: la identidad de los formularios de acceso y la edad de los formularios. La primera hace referencia al algoritmo que trata de verificar la legitimidad de un formulario de acceso basándose en que lo habitual es que exista relación entre la página que se encuentra dentro de la propiedad *action* de un formulario HTML y el propio dominio del sitio web actual. La segunda heurística consiste en calcular la edad de cada formulario en función de la dirección URL a la que apunte su propiedad *action*. Si esta dirección es similar al dominio base del sitio web se considera que la edad del formulario es la misma que la del dominio principal. En caso contrario se analiza mediante una búsqueda WHOIS y si la edad de alguno de las direcciones de cada formulario es menor que un mes el algoritmo devuelve un 1 lo que implica que el formulario puede ser sospechoso de phishing.

3. **Estructura de dominios y evaluación de los números de puertos.**
   En este paso existen otras tres heurísticas: direcciones IP sin nombre de dominio, direcciones URL que contienen solo parte del dominio y números de puerto inválidos. Es bastante habitual en los ataques de phishing que se utilicen direcciones IP en lugar de su traducción en nombre de dominio debido a que muchos de los servidores que alojan el servicio de phishing suelen ser maquinas comprometidas que no están registradas en ningún servidor DNS. Por otro lado, lo normal es que los accesos legítimos a un servidor se realicen a través de su nombre de dominio.
   En lo que respecta a la segunda heurística, los ataques de phishing suelen contener parte del dominio real de la web para que las direcciones URL engañen a los usuarios haciéndose pasar por direcciones seguras.
   Por último, la tercera heurística del sistema planteado compara los números de puerto que pueda contener el sitio web con el protocolo que forma parte de cada URL para ver si tienen concuerda. Por ejemplo, una petición a una página web lo normal es que utilice el puerto 80 que es el puerto por defecto para este tipo de servicio.

4. **Análisis de los nombres de dominio.**
   En esta etapa se utilizan tres heurísticas diferentes: la credibilidad del nombre de dominio, la edad del nombre de dominio y la identidad del nombre del dominio.
   La credibilidad del nombre del dominio se basas en el ranking que el sistema *PageRank* le otorga a un dominio en particular denotando su importancia. Para el sistema propuesto por los autores se ha utilizado el sistema *PageRank* de Google que otorga una puntuación de 10 a las páginas más populares de internet 0 a las paginas desconocidas. El valor por defecto establecido por los autores para determinar la credibilidad del nombre de dominio es un 5 dentro de la escala del Google *PageRank*.
   La edad del nombre de dominio se obtiene a través de una implementación de una petición a un servicio WHOIS determinando que los sitios que tengan una fecha de creación inferior a un mes son sospechosos de ser fraudulentos ya que durante un ataque phishing lo normal es que los dominios solo tengan unas horas o días de vida.
   La identidad del nombre de dominio relación el contenido de un sitio web con el dominio empleado. Solo si la identidad de la URL es similar al contenido del sitio web este se considera legítimo.

5. **Evaluaciones de los dominios TLD.**
   Todas las URL contienen un TLD en su nombre de dominio que sirve para identificar a información asociada al sitio web con el que se relación el domino. Para esta evaluación se utilizan dos heurísticas diferentes: la presencia de múltiples dominios TLD y las validaciones del código de país. La primera heurística comprueba que una URL solo exista un único dominio TLD. La segunda heurística verifica que el código del país especificado en el dominio concuerde con la información que facilita el hosting públicamente a partir de la geolocalización de la dirección IP donde se encuentra alojado el dominio.

6. **Evaluaciones adicionales de URL.**
   Esta etapa cuenta con dos heurísticas diferentes: la existencia de palabras clave relacionadas con el phishing dentro de la URL y el certificado SSL/TLS. Es normal que en los sitios destinados al phishing se puedan encontrar determinadas palabras clave que se repitan en varios ataques por ese motivo se analizan todas las palabras que forman las direcciones URL para ver si alguna forma parte de las listas públicas de varios estudios relacionados con los ataques de phishing. La segunda heurística comprueba la legitimidad de los certificados empleados por los sitios web que contengan formularios de acceso o registro.

7. **Evaluaciones de dominio basadas en caracteres especiales.**
   Esta etapa comprueba la ocurrencia de caracteres especiales dentro de los nombres de dominio de un sitio web mediante el uso de tres heurísticas distintas: la ocurrencia del carácter @, la ocurrencia de más de 5 caracteres punto (.) y la ocurrencia de las dobles barras (//). Si aparece alguno de estos tres caracteres dentro de los dominios de un sitio web se consideran sospechosos puesto que suelen estar asociados a ataques de phishing.

Como resumen, el generador de características de un sitio web del sistema descrito por los autores del artículo plantea el uso de las siguientes heurísticas:

1. Identidad de la URL.
2. Identidad del conjunto de palabras clave.
3. Identidad de los formularios de acceso.
4. Edad de los formularios de acceso.
5. Direcciones IP sin nombre de dominio asociado.
6. Direcciones URL que contienen solo parte del dominio.
7. Números de puerto inválidos.
8. Credibilidad del nombre de dominio.
9. Edad del nombre de dominio.
10. Identidad del nombre del dominio.
11. La presencia de múltiples dominios TLD.
12. Validaciones del código de país.
13. Existencia de palabras clave relacionadas con el phishing dentro de la URL.
14. Certificados SSL/TLS.
15. Carácter @ dentro del nombre de dominio.
16. Aparición de más de 5 caracteres punto (.) dentro del nombre de dominio.
17. Aparición de dobles barras (//) dentro del nombre de dominio.

Clasificador de phishing

El clasificador elegido para el algoritmo de *Machine Learning* empleado por los autores consiste en un método de análisis de datos y reconocimiento de patrones conocido como máquinas de vectores de soporte (SVM).

Este clasificador toma un conjunto de datos de entrada y es capaz de predecir una de dos posibles clases a partir del entrenamiento de un modelo generado para un conjunto de datos de muestras. Se trata de un clasificador de aprendizaje supervisado. Este tipo de aprendizaje consiste en un conjunto de técnicas cuya finalidad es deducir una función a partir de un conjunto de datos de entrenamiento. Por este motivo, es necesario dividir los conjuntos de datos en datos de entrenamiento del clasificador y datos de prueba. Los datos de entrenamiento contienen las etiquetas de las clases (phishing o legítimo) junto con el valor conocido de cada una de las características analizadas de cada página web. En función de esos datos de entrenamiento se genera un modelo de clasificación junto con una serie de reglas que permitirá clasificar una página web durante la fase de prueba o test.

Para los clasificadores de tipo SVM es necesario que la lista de las características definidas por el descriptor del dominio sea un vector de números reales, por lo que si alguno de los valores se corresponde con alguna categoría discreta será necesario convertirlo a un equivalente numérico. Además, será necesario escalar los valores para que todos estén normalizados y ninguno domine la clasificación.

En la etapa de clasificación los autores han optado por el tipo de modelo SVC y por el núcleo RBF.

Evaluación de resultados obtenidos

Una vez definidos las principales características o módulos del sistema de detección de phishing planteado por los autores, el artículo expone los resultados experimentales de los mismos:

1. Evaluación del sistema buscador de formularios de acceso: Este módulo detectó de forma correcta el 98.05% de las páginas analizadas que contenían algún tipo de formulario de acceso donde los usuarios pudieran introducir sus credenciales de entrada. En cuanto a las paginas analizadas que contenían phishing, el algoritmo empleado fue capaz de detectar el 97.27% de los formularios que utilizaban dichas páginas web.

2. Evaluación del sistema sin utilizar la lista blanca de sitios y el buscador de formularios de acceso: En un primer experimento se comprobó la eficacia del sistema detector de phishing basado solamente en el módulo de extracción de características de una página web. En la figura 11 se muestra la información de la matriz de confusión con los resultados obtenidos en la publicación sin utilizar formularios ni lista blanca.

Matriz de Confusión		
	Páginas con Phishing	Páginas Legítimas
Clasificadas como Phishing	Verdaderos Positivos = 1733	Falsos Positivos = 12
Clasificadas como Legítimas	Falsos Negativos = 31	Verdaderos Negativos = 688

Figura 11. Resultados obtenidos en el escenario sin detección de formularios ni lista blanca.

3. Evaluación de sistema utilizando la lista blanca de sitios y el buscador de formularios de acceso: En este caso los resultados obtenidos son del sistema en su totalidad. En primer lugar, las páginas web han pasado por el filtro de la lista blanca de sitios preaprobados por el usuario y se ha ejecutado el algoritmo de búsqueda de formularios de acceso. La figura 12 muestra la matriz de confusión en donde se pueden apreciar cómo se han mejorado los resultados de las pruebas del escenario anterior.

Matriz de Confusión		
	Páginas con Phishing	Páginas Legítimas
Clasificadas como Phishing	Verdaderos Positivos = 1758	Falsos Positivos = 3
Clasificadas como Legítimas	Falsos Negativos = 6	Verdaderos Negativos = 697

Figura 12. Resultados obtenidos del sistema planteado en su totalidad.

A continuación, en la figura 13, se muestra el tanto por cierto que aporta cada una de las heurísticas más importantes del sistema de obtención de características de un sitio web que han sido empleadas en el algoritmo de clasificación SVM empleado por los autores:

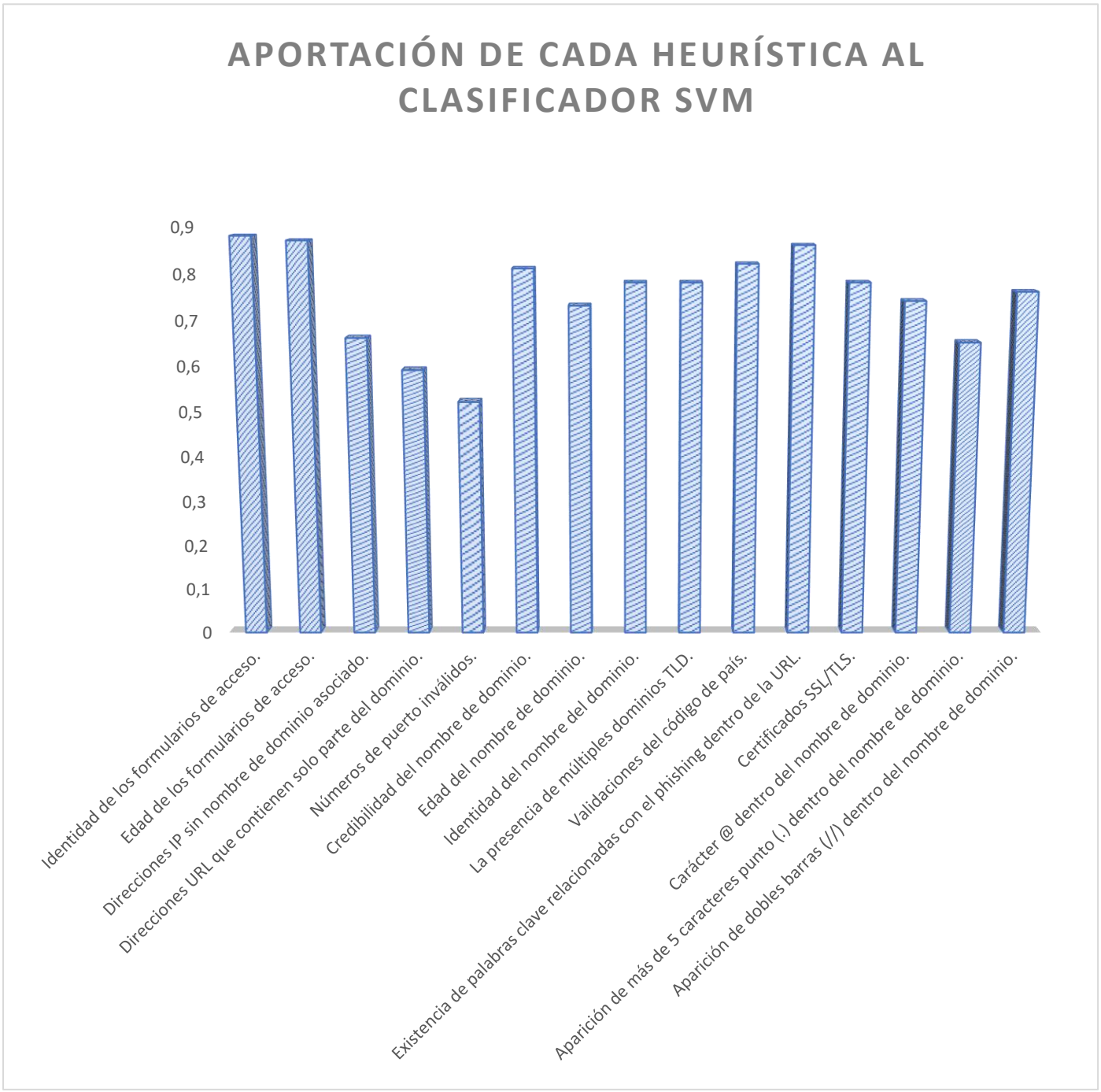

Figura 13. Aporte de las heurísticas a los resultados del clasificador SVM descrito por [GowthamKrishnamurthib2014].

2.6. Entrenamiento de usuarios para la auto detección de phishing

En ocasiones, las medidas más sencillas pueden resultar las más eficaces para contrarrestar los intentos de Phishing por parte de los cibercriminales. En la actualidad se destinan grandes cantidades de dinero y de recursos al desarrollo de complejos sistemas de detección e identificación de sitios maliciosos o de campañas de phishing, pero estos poco pueden hacer si luego los propios usuarios de los sistemas de información no aplican el sentido común durante la navegación a través de la red. Es por este motivo por el que cobra vital importancia realizar campañas de formación y de entrenamiento para que los usuarios de los sistemas de información tengan siempre una actitud preventiva y estén realmente concienciados sobre la importancia de proteger sus credenciales de acceso y lo que su pérdida puede suponer tanto para ellos mismos como para las organizaciones a las que pertenecen.

Se han desarrollado numerosas aproximaciones para lograr la concienciación de los usuarios de una organización. Algunos de estas aproximaciones consisten en el desarrollo de videojuegos [ArachchilageLove2013]. Los autores de este articulo investigaron si era posible concienciar a los usuarios del problema que supone el phishing a través del uso de un videojuego para móvil. Sus conclusiones fuero claras, y demostraron que los usuarios que formaron parte del entrenamiento eran capaces de detectar páginas falsas con una mayor tasa de aciertos que los usuarios que no formaron parte del estudio.

Otra de las soluciones habituales para el entrenamiento de los usuarios en la detección de ataques de phishing dentro de las organizaciones consiste en que los departamentos de seguridad envíen correos electrónicos maliciosos desde una dirección de correo legítima con el objetivo de simular un ataque de phishing dentro de la propia organización. De esta manera se mantiene en alerta a los usuarios de los sistemas de información y se puede evaluar el grado o nivel de exposición que tiene la propia organización ante un posible ataque futuro de phishing.

Este tipo de entrenamientos se discute en algunas publicaciones científicas como [ArachchilageRheeShengHasan2007] o en [RolandCurtisAaron2007]. La técnica sobre la que se basan estos programas de entrenamiento de usuarios consiste en lo que se denomina entrenamiento embebido y consiste en que para lograr que los ataques simulados sean lo más creíble posible, estos deben estar integrados dentro de la actividad normal del usuario dentro de la organización.

3. Metodología

Durante la realización de este TFM se ha estudiado la problemática de los ataques de ingeniería social en los entornos de red. Se ha puesto el foco del trabajo en el caso de los ataques más conocidos, los ataques de phishing. Una vez visto el impacto que tienen estos ataques y como se suelen comportar se ha analizado el estado del arte actual para ver cuáles son las principales medidas de seguridad que hoy en día tenemos para defendernos. Sin duda, cuando hablamos de medidas de seguridad hay una máxima que casi siempre se cumple: "es mejor prevenir que curar". Por ese motivo, la parte práctica de este TFM se ha centrado en los sistemas de identificación de phishing, más en concreto, en la detección de sitios web maliciosos.

Como se ha detallado en el capítulo de Planificación de este documento, la parte práctica del TFM se ha enfocado a la realización de un proyecto de ingeniería con el objetivo de desarrollar un algoritmo de *Machine Learning* capaz de identificar y clasificar direcciones URL como legítimas o maliciosas para elaborar una herramienta de defensa frente al tipo de ataque de ingeniería de red más extendido, el phishing.

Siguiendo las ideas expuestas en el artículo de [GowthamKrishnamurthib2014] y partiendo de las bases técnicas que se describen el libro de [MalwareDataScience2018], se ha diseñado un algoritmo de *Machine Learning* que una vez entrenado con un conjunto de direcciones URL de sitios web maliciosos y sitios web legítimos es capaz de clasificar una dirección URL recibida como parámetro identificándola como segura o como posible sitio malicioso. La ventaja que tiene esta implementación es que el algoritmo requiere durante la fase de entrenamiento de una cantidad de tiempo considerable para ejecutarse pero luego se genera un modelo matemático que puede identificar sitios web no conocidos en muy poco tiempo por lo que en futuros trabajos se podría integrar este algoritmo en otro tipo de software como pueda ser un navegador web o un programa de gestión de correo electrónico que avisaría al usuario del análisis realizado a una URL antes de permitirle el acceso.

En este punto es conveniente recalcar que cuando se trabaja con programas maliciosos es necesario extremar la precauciones a la hora de manipular recursos o activos que puedan estar comprometidos minimizando el riesgo de exponer nuestra propia infraestructura ante dichas amenazas. Cuando se trabaja con binarios con código malicioso se suelen utilizar máquinas virtuales o contenedores que aíslen el objeto sospechoso del resto del equipo utilizado para evitar su posible infección. En el caso práctico desarrollado para este TFM no ha sido necesario puesto que el malware sobre el que se ha trabajado en el caso práctico está alojado en sitios web ajenos y en la gran mayoría de enlaces analizados los sitios fraudulentos han sido creados con el objetivo final de robar información de usuario. Las visitas que se han realizado a las páginas web sospechosas de ser sitios fraudulentos se han realizado desde el navegador Firefox instalado en una máquina virtual con las siguientes características:

Sistema operativo Ubuntu 64 bits 17.10 con 2048 MB de memoria RAM.

El resto del tiempo dedicado al desarrollo de los algoritmos necesarios el riesgo se ha atenuado accediendo exclusivamente al contenido de páginas analizadas a través de comandos específicos del programa desarrollado evitando así poner en riesgo el equipo informático utilizado para la realización del presente TFM. No se ha optado por realizar los experimentos en una máquina virtual debido a que durante la fase de extracción de características ha sido necesario contar con el 100 % de los recursos hardware disponibles y con la virtualización el rendimiento empeora

al introducir elementos intermedios como el hipervisor (el monitor de la máquina virtual) que a implica un consumo de recursos adicional no deseado.

Una vez establecido el principal objetivo de caso práctico a desarrollar es conveniente especificar que herramientas tecnológicas se van a utilizar para llevarlo a cabo. Para este TFM se ha elegido Python en su versión 3.7 como lenguaje de scripting debido a su sencillez y a la gran cantidad de librerías y recursos disponibles de manera libre y gratuita en la red y que facilitarán la resolución de la tarea definida en esta sección.

Para el diseño del algoritmo de clasificación se ha utilizado la biblioteca de aprendizaje automático [SCIKITLEARN20]. Se trata de una de las librerías se software libre dedicadas a la realización de algoritmos de *Machine Learning* más importantes que existen actualmente dentro del ecosistema del lenguaje de programación Python. Esta librería incluye varios algoritmos de clasificación, regresión y análisis de grupos que encajan en el propósito del actual trabajo practico de este TFM puesto que nos van a permitir generar un modelo matemático de predicción de clase [URL legitima – URL phishing] para los conjuntos de datos analizados.

En este punto es importante destacar la necesidad de contar con grandes volúmenes de datos tanto para la fase de entrenamiento como para la fase de clasificación y evaluación de los algoritmos probados. Para la obtención de URLs de sitios web donde se haya identificado actividad fraudulenta se ha recurrido a sitio web [PHISTANK20]. *PhishTank* es una comunidad colaborativa donde se almacenan datos e información sobre phishing en Internet. Además, *PhishTank* proporciona acceso gratuito a miles de sitios webs de phishing que han sido validados por su comunidad. Detrás de esta página web se encuentra la compañía *OpenDNS* que destina parte de sus recursos en compartir la información de todo el phishing que se reporta en internet con el objetivo de hacer más segura la red. Desde la sección "*Developers*" de su web es posible descargar un fichero CSV con todos los sitios clasificados como phishing y que permanece activos. El que los sitios web sigan activos es fundamental para el caso práctico puesto que para poder desarrollar algoritmos de aprendizaje automático y obtener los vectores de características es necesario poder extraer información de todas las características que los identifican. Una vez descargado el fichero CSV lo hemos reducido dejando únicamente la dirección URL del sitio web por cada línea del archivo.

Por otro lado, la recopilación de sitios web legítimos, aunque pueda parecer más trivial también es muy importante y de nuevo el problema se encuentra en el volumen necesario. En esta ocasión, las fuentes utilizadas han sido [MOZ2020] y [GTMETRIX20]. En la primera página se puede descargar un listado con las 500 direcciones URL más visitadas de internet mientras que en la segunda hay disponible para su descarga de manera gratuita de otro ranking con los 1000 sitios web más visitados de acuerdo con las estadísticas de Google. Al igual que en el caso del fichero CSV con los sitios malignos, se ha filtrado el fichero de partida dejando únicamente la dirección URL de cada sitio web legítimo por cada línea del CSV.

4. Caso de estudio

4.1. Detección de Phishing mediante un algoritmo de *Machine Learning*.

Machine Learning o como se dice en castellano, el aprendizaje automático, es un subcampo de la inteligencia artificial que se encargar del desarrollo de algoritmos que mejoren sus resultados a partir de la experiencia y del resultado de inferir conocimiento a partir de amplios conjuntos de datos. Dentro de este campo de estudio se suele trabajar con dos metodologías diferentes para lograr que un ordenador extraiga o "aprenda" información relevante sobre un conjunto de datos dado. Las dos formas de aprendizaje son el supervisado y el no supervisado.

El primero de ellos es el más extendido y el objetivo es que el algoritmo aprenda las conclusiones necesarias para interpretar la salida del programa que se conoce de antemano. Para estos algoritmos necesitamos de conjuntos de datos de entrenamiento clasificados de manera correcta y una vez realizado el proceso de aprendizaje el algoritmo debería de ser capaz de inferir la clasificación de una nueva entrada no vista anteriormente. Los algoritmos más utilizados de este tipo son los clasificadores, las máquinas de vector soporte, los métodos de regresión, etc.

Por otro lado, el aprendizaje no supervisado se centra en el desarrollo de algoritmos que sean capaces de identificar patrones dentro de conjuntos de datos sin conocer ningún tipo de clasificación previa de los datos. Entre los algoritmos más utilizados de esta tipología están los algoritmos de *clustering* y *k-means* entre otros.

En este caso de estudio se quiere diseñar un programa capaz de identificar si un sitio web es un sitio legítimo o si se trata de algún tipo de phishing, es decir, queremos diseñar un programa capaz de clasificar un sitio web entre dos clases (legítimo o phishing) a partir de la información que forma parte de lo que se conoce como el "vector de características". Estos vectores contienen una representación característica cada uno de los elementos que forman parte de los datos de entrenamiento.

En el caso de estudio que vamos a desarrollar vamos a utilizar un algoritmo de aprendizaje supervisado, en el que tengamos una función que clasifique una serie de vectores característicos en función de la experiencia que el algoritmo obtiene de un entrenamiento previo a partir de un conjunto de datos sobre los que conocemos su clasificación a priori. Uno de los modelos predictivos más utilizados en los campos de minería de datos, la estadística y el aprendizaje automático que existe en la actualidad es el basado en lo que se conoce como arboles de decisión. En este caso, estamos hablando de un árbol de clasificación en el que la clase final de destino de cada entrada de la función puede tomar un conjunto finito de valores. En los árboles de decisión, cada uno de los nodos internos se etiquetan a partir del resultado de una función de entrada y el resultado de la función genera una nueva rama hasta llegar a los nodos finales, o nodos hoja. Cada una de estas hojas se marca con la prioridad de pertenecer a una de las clases finales buscadas.

En el algoritmo diseñado se ha optado por utilizar un clasificador *Random Forest*. Este tipo de métodos de aprendizaje automático está basado en la construcción de múltiples arboles de decisión durante el tiempo de entrenamiento y generar la clasificación final como la moda de las clases o la media de los árboles individuales. Con este mecanismo se consigue corregir en

parte el problema de sobre entrenamiento de los árboles de decisión durante la fase de entrenamiento.

El código completo del algoritmo que se ha desarrollado para la realización del caso de estudio de este TFM se puede encontrar en un repositorio público de *Bitbucket* accesible desde este enlace [REPOSITORIO20].

4.1.1. Proceso de instalación

Para la instalación del programa es necesario hacer uso del fichero *requirements.txt* que se encuentra en el [REPOSITORIO20] de código. Una vez descargado, basta con lanzar el siguiente comando desde una consola de terminal con Python 3.7 instalado desde la carpeta donde se encuentre el repositorio descargado:

pip install -r requirements.txt

Ahora que témenos listo el entorno de Python con las dependencias de ejecución del programa, tenemos que preparar los ficheros *.csv* con las URL de los sitios que vamos a utilizar para el entrenamiento y evaluación del algoritmo. Estas direcciones URL están separadas en dos ficheros. En primer lugar, está el fichero con las direcciones URL legítimas denominado *"full_legit.csv"*. El segundo fichero es el que contiene la lista de URL con sitios web que han sido identificados como phishing por la comunidad online *PhishTank*. Ambos ficheros son CSV con formato *ISO 8859-1*.

4.1.2. Estructura del código

El programa desarrollado es capaz de entrenar un algoritmo de aprendizaje automático, de analizar la dirección URL de un sitio web para clasificarla como legitima o phishing y también de evaluar el rendimiento del propio algoritmo.

Lectura de argumentos

En primer lugar, se ha generado un analizador de argumentos recibidos por parámetro cuando se invoca el programa. Actualmente, el script acepta los siguientes argumentos gracias al uso de la librería *argparse* como se puede ver en la figura 14:

```
parser.add_argument("--generate_model",default=None,help="Generate a model")
parser.add_argument("--phishing_sites",default=None,help="CSV phishing training URLs")
parser.add_argument("--legit_sites",default=None,help="CSV legitimate training URLs")
parser.add_argument("--analyze_site",default=None,help="URL to analyze")
parser.add_argument("--features",default=None,help="Features array filename. If exists opens. If not, creates it")
parser.add_argument("--evaluate",  choices=['normal', 'cross'], help="Perform normal evaluation or crossvalidation")
parser.add_argument("--max_urls",default=10000,help="Max number of URL to analyze from each CSV files")
```

Figura 14. Argumentos de entrada del clasificador de phishing desarrollado.

El programa tiene varios modos de uso:

- Análisis de una URL. Este modo permite analizar una URL recibida por parámetro a través de *analyze_site*. Es necesario incluir a través del parámetro *features* el nombre de un modelo de datos previamente generado para que el análisis se pueda realizar.
- Generación de un modelo. Permite generar un modelo con el vector de características de todos los sitios URL analizados. Para acceder a este modo es necesario invocar el programa principal con el parámetro *generate_model* y además tenemos que incluir los parámetros *phishing_sites* (con el fichero CSV de sitios con phishing), *legit_sites* (con el

fichero CSV con sitios legítimos), *features* (con el nombre del modelo que se va a generar y almacenar en el disco). La generación del modelo puede durar bastantes horas.

- Evaluación de un modelo. Este modo genera la evaluación del modelo recibido a través del parámetro *features*. Es necesario indicar que tipo de evaluación se va a realizar a través del parámetro *evaluate* que puede ser *normal* o *cross* para realizar la validación cruzada. Se muestra un ejemplo de ejecución en las instrucciones del repositorio del trabajo.

Entrenamiento del algoritmo

Esta parte de algoritmo contiene el código que más tiempo de ejecución necesita. Esto es debido a que, durante el procesamiento de los datos de entrenamiento, el programa debe de obtener un vector de características específico para cada una de las direcciones URLs tratadas.

El éxito o el fracaso del algoritmo que se ha desarrollado depende en gran medida del proceso de extracción de características llevado a cabo, así como del volumen de datos previamente clasificados con el que se parta de inicio. Cuanto más representativas sean las características a la hora de diferenciar a los sitios web legítimos de los sitios web que contienen phishing más preciso será el programa de clasificación desarrollado. De ahí la importancia de elegir correctamente las características que debemos seleccionar ya que, por otro lado, al tratarse de muchos datos de entrenamiento, no tenemos recursos infinitos para entrenar nuestro algoritmo. El criterio seguido para elegir las características ha sido el análisis del artículo publicado por [GowthamKrishnamurthib2014] y tras estudiar las características como se vio en el apartado *Generador de características de un sitio web* de este mismo TFM seleccionar solo las más representativas o al menos las que pudieran ser implementadas a través de algún tipo de script en Python.

Las características elegidas para el algoritmo son las siguientes:

- Aparición del carácter arroba dentro del texto que forma la URL. La aparición del carácter *"@"* no se suele dar en URL de sitios legítimos por lo que su aparición se considerar como sospechosa.
- Aparición de más de 5 caracteres "." en el texto que forma la URL. Las direcciones URL suelen contener el carácter punto para separar dominios de subdominios, pero las estadísticas indican que si aparecen más de 5 caracteres punto el sitio es sospechoso de contener algún tipo de malware.
- Aparición de dobles barras "//" en el texto que forma la URL. Este tipo de carácter se suele emplear para redirecciones a otros sitios web y no es habitual encontrarlo dentro de una dirección URL legitima salvo en la parte de la dirección URL que indica el protocolo necesario.
- Que en la dirección URL aparezca la dirección IP en lugar de nombre de dominio. Los sitios web que contienen algún tipo de phishing suelen ser alojados en servidores comprometidos y cuya dirección IP no se registra dentro de los servidores públicos de dominio para evitar su rastreo y por agilizar los ataques de este tipo. Cualquier página web legitima sí que tendrá un nombre de dominio registrado por lo que si aparece en la dirección URL una dirección las probabilidades de que se trate de un sitio web malicioso son muy elevadas. Esto es algo que los ciberdelincuentes tienen muy en cuenta por lo que no suele ser habitual encontrarlo en casos reales.

- Que en la dirección URL se incluya el número de puerto. Al igual que en el punto anterior, no es habitual que una dirección URL contenga ni la dirección IP ni el puerto al que se debe dirigir el navegador del cliente para recuperar un recurso en línea.
- Que dentro de la URL exista más de un único TLD. La aparición de múltiples dominios de alto nivel suele significar que los atacantes intentan suplantar un sitio web legitimo añadiendo otro domino al que es conocido para engañar a una posible víctima.
- Que la fecha de creación del dominio registrado sea inferior o igual a 20 días. Los sitios web utilizados durante los ataques de phishing no suelen tener una vida útil muy larga por que suelen ser detectados y reportados por lo que la edad de los dominios suele ser de pocas horas o días por lo que esto nos puede ser bastante útil a la hora de clasificar un sitio web.
- Que el código del país que viene especificado en el dominio no coincida con la información de geolocalización de la dirección IP. Si el código de país que forma parte de la dirección URL analizada no coincide con el análisis de geolocalización de la IP asociada con dicho domino es una señal bastante clara de que el sitio web es fraudulento ya que esto es una práctica prohibida por los servidores de dominios públicos de internet. Para la obtención de esta información es necesario registrarse de manera gratuita en [IPINFO20] y obtener una clave de API que nos permita hacer peticiones hacia una API REST para obtener la información de geolocalización de una dirección IP. Este servicio gratuito tiene una limitación de 50000 peticiones al mes.
- Que dentro de la URL aparezca repetida la palabra http. Es una táctica habitual dentro de los ataques de phishing el intentar engañar al usuario mediante una redirección dentro de la propia dirección URL por lo que si aparece es sospechosos de ser un sitio fraudulento.
- Si dentro del contenido HTML del sitio web analizado existen formularios, que las direcciones URL a las que redirigen los mismos apunten a un dominio diferente del propio de la URL principal. Lo normal es que la dirección que aparece dentro del atributo *action* apunte a una página web alojada dentro del propio dominio del sitio web, ya sea a través de un enlace relativo o a través de un enlace absoluto. Si se trata de un enlace absoluto y este apunta a un dominio distinto es algo bastante sospechoso. En el caso de que el valor del atributo *action* tenga el valor en blanco o bien contenga el texto *"about:blank"* también se va a considerar como sospechoso de phishing.
- Que dentro del texto de la dirección URL aparece alguna palabra repetida. La repetición de algunas palabras dentro de una URL puede ser utilizada para engañar a una víctima cuando se pretende generar la ilusión de que la URL pertenece a un domino parecido a uno real y conocido por el usuario.
- Que dentro del texto de la dirección URL aparezca alguna de las palabras sospechosas reportadas por los autores de [GaretaProvosChevRubin2007]. Los autores de ese artículo realizaron un estudio estadístico de las palabras que aparecían con más frecuencia en un conjunto de sitios maliciosos analizados y llegaron a la conclusión de que las palabras más repetidas dentro las URLs de sitios web maliciosos eran: *webscr, secure, banking, ebayisapi, account, confirm, login, signin.*

El objetivo final del entrenamiento es obtener dos vectores **X** e **y** para entrenar el clasificador.

El array denominado X contiene para todas las URLs analizadas, una lista con diccionarios con todas las características vistas anteriormente y su correspondiente valor (1 si se han encontrado

durante la fase de análisis y 0 en el caso contrario). En la figura 15 se puede apreciar una captura de pantalla con el contenido de uno de los diccionarios almacenados dentro del vector **X**:

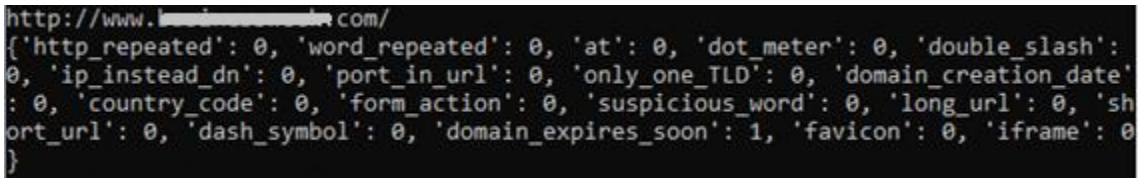

Figura 15. Contenido del diccionario con las características extraídas para una URL

El segundo vector, el vector **y**, representa la etiqueta o la clase de cada una de las URL analizadas (0 si el sitio es legítimo y 1 si el sitio contiene algún tipo de phishing).

Por último, una vez se han obtenido los dos vectores **X** e **y** se genera una nueva instancia del clasificador *RandomForest* que nos facilita la librería *scikit-learn* y se le entrena pasándole los dos vectores como argumento. Como todo este proceso puede ser bastante lento, al finalizar serializamos el objeto clasificador generado almacenándolo en el disco duro gracias al uso de la librería *pickle* de Python. De esta maneara podremos utilizarlo en la siguiente fase de análisis de una nueva URL sin necesidad de tenerlo que generar de nuevo.

Análisis de una nueva URL

Esta parte del programa analiza una URL recibida por parámetro siempre que previamente se haya entrenado y almacenado un clasificador con el paso anterior. Básicamente lo que hace es extraer las mismas características del apartado de entrenamiento e invocar a la función *predict_proba* del clasificador entrenado previamente. Si el resultado es una probabilidad mayor del 0.5 el sitio se clasifica como que probablemente contenga phishing y si las probabilidades que devuelve la función de *scikit-learn* son inferiores a 0,5 se considera que es un sitio web legítimo. Las siguientes figuras muestran el resultado de analizar dos URLs distintas, una se corresponde con un sitio web legítimo (figura 16) y la segunda a un sitio web clasificado como phishing obtenido del sitio web [PHISTANK20] (figura 17).

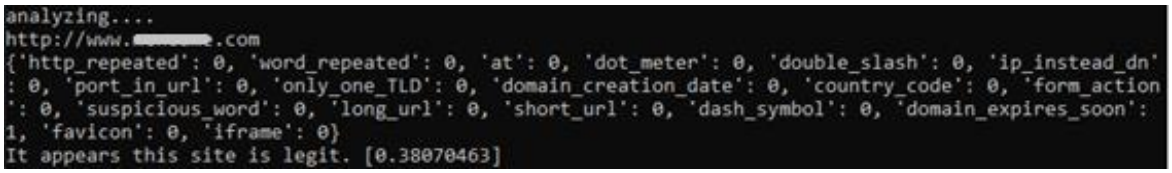

Figura 16. Captura de pantalla del resultado de analizar un sitio web seguro.

Figura 17. Captura de pantalla de analizar un sitio web malicioso.

En ambas capturas de pantalla se puede apreciar el resultado del vector de características que el algoritmo ha extraído del análisis del sitio web cuya URL se recibe por parámetro al ser invocado.

Evaluación del algoritmo

Para la evaluación del algoritmo se han diseñado dos funciones o métodos independientes.

El primero de ellos recibe las matrices de las características y las clases y las separa de manera aleatoria entre datos de entrenamiento y datos de prueba. Luego genera una nueva versión del clasificador *RandomForest* y lo entrena con los conjuntos de datos destinados al entrenamiento.

Cuando el entrenamiento termina, y gracias a la ayuda de las funciones específicas de evaluación de algoritmos que se incluyen en el paquete *scikit-learn* de Python como *predict_proba*, se obtiene una nueva matriz con las probabilidades de que el conjunto de datos de prueba que vamos a utilizar pertenezca a sitios legítimos o a sitios maliciosos.

Esta información se utiliza dentro de la función de evaluación para generar un gráfico con la curva ROC mediante el uso de la librería *pyplot*. Las curvas ROC muestran la evolución de los falsos positivos por el contrario posición con los falsos negativos que se detectan con el clasificador a partir de la modificación del valor de umbral de discriminación. Este valor de discriminación lo genera automáticamente la función *roc_curve* y tiene la siguiente peculiaridad: cuanto mayor es la sensibilidad de ese parámetro mayor es el número de falsos positivos que se obtienen, pero mejor es la ratio de detección del clasificador. Por el contrario, cuanto menor sea la sensibilidad del parámetro, menos falsos positivos se obtienen, pero también disminuyen el número de detecciones correctas del clasificador.

En este primer método de evaluación, se han dividen los datos de entrenamiento para obtener dos matrices, la matriz con los datos que van a ser utilizados para el entrenamiento contendrá un 80 % de los datos y la matriz con los datos que se van a destinar a probar el modelo contendrá el restante 20 % de los datos. De esta manera podemos comprobar la eficacia del detector de phishing cuando lo probamos con direcciones URL nuevas que no forman parte del conjunto de datos de entrenamiento.

El segundo método de evaluación es una extensión del método anterior solo qué, en esta ocasión se hace uso de una técnica conocida como validación cruzada. Esta técnica consiste en realizar varios experimentos en nuestros datos de entrenamiento en lugar de uno solo. Lo que se hace es dividir los datos de entrenamiento en un número de subconjuntos (en el caso práctico se han utilizado cinco subconjuntos de datos) y luego realizar cinco fases de prueba con esos datos. El proceso consiste en seleccionar probar con cada uno de los subconjuntos frente a un clasificador entrenado con los otros cuatro subconjuntos de datos repitiendo el proceso hasta haber probado con todos los subconjuntos de manera independiente. Una vez realizado el proceso de entrenamiento y prueba de manera completa obtenemos cinco curvas ROC en lugar de una única curva. Esta forma de evaluación ofrece unos resultados más completos ya que realizamos cinco experimentos distintos con los datos en lugar de uno, pero tiene un problema importante y es que, al dividir el volumen de datos de entrenamiento la precisión del modelo puede decrecer.

5. Resultados

En este apartado se analizan los resultados obtenidos por el clasificador como resultado de aplicar los dos métodos de evaluación desarrollados y que se han comentado en la sección anterior. Para obtener los resultados se han utilizado las siguientes funciones del paquete *metrics* de la librería *scikit-learn*:

- ***mean_absolute_error***, o como se diría en castellano, el error absoluto medio, representa la diferencia que existe entre dos variables continuas. Si se tienen un par de conjuntos de datos, unos observados y otros calculados sobre el mismo fenómeno se puede utilizar el error absoluto medio como un elemento que cuantifique la precisión de la predicción realizada por el sistema.
- ***mean_squeared_error***, o error cuadrático medio, es un estimador de un procedimiento para la estimación de cantidades no observadas que mide el valor promedio de los cuadrados de los errores. De esta manera ofrece información entre la diferencia de los cuadrados promedio de los valores estimados contra el valor real obtenido por el sistema. A este valor también se le ha calculado la raíz cuadrada que representa el error de la raíz cuadrada de la media o la desviación de la raíz cuadrada media. Cuando el estimador no tiene sesgo, este último valor, el de la raíz cuadrada, se conoce como la desviación estándar y nos permite cuantificar la dispersión del conjunto de datos medidos.
- ***confusion_matrix***, o matriz de confusión es una herramienta que se utiliza en el campo de la estadística y de la inteligencia artificial para visualizar como ha funcionado un algoritmo de aprendizaje supervisado. En estas matrices, las valores que aparecen en las columnas representan las predicciones mientras que los valores de las filas son los valores reales asociados a cada clase real.
- ***classification_report***, esta función nos muestra un informe con varios indicadores que pueden resultar bastante útiles a la hora de analizar el rendimiento de nuestro clasificador. Los indicadores que muestra son los siguientes:
    - ***precision:*** Representa el número de miembros de una clase correctamente identificados por el clasificador divido por todas las veces que el modelo clasificó correcta e incorrectamente la clase en cuestión. Por ejemplo, en el caso de la clase “phishing [1]” el valor de este indicador representa el número de veces que se identificó correctamente una dirección URL como contendora de phishing dividido por el número total (correctas e incorrectas) de predicciones que el modelo realizo sobre sitios con phishing. El valor de *precision (P)* se puede definir con siguiente ecuación:

$$\mathrm{P} = \frac{\mathrm{Tp}}{\mathrm{Tp} + \mathrm{Fp}}$$

      Por ejemplo, si queremos calcular el valor de P para un clasificador que ha identificado correctamente (Tp) como phishing 100 URLs y que ha identificado incorrectamente (Fp) 30 URLs como phishing cuando no lo eran, podríamos calcular el valor de *precision de la siguiente manera*:

$$\mathrm{P} = \frac{100}{100 + 30} = 0{,}77$$

- ***recall:*** Este valor se calcula como el número de elementos correctamente identificados de una clase (TP) dividido por el número total de miembros de dicha clase que se obtiene de sumar los elementos que son de la clase y se han identificado como tal (TP) y los que son de la clase pero no han sido identificados (FN). El valor de *recall (R)* se puede definir con siguiente ecuación:

$$\mathrm{R} = \frac{\mathrm{Tp}}{\mathrm{Tp} + \mathrm{Fn}}$$

Por ejemplo, si queremos calcular el valor de R para un clasificador que ha identificado correctamente (Tp) como phishing 100 URLs pero que ha identificado como legítimos, aunque no lo fueran (Fn) 25 URLs, podríamos calcular el valor de *recall de la siguiente manera:*

$$\mathrm{R} = \frac{80}{80 + 20} = 0,8$$

- ***f1 score:*** Combina los dos valores anteriores (*P* y *R*) en una única métrica definida como la media armónica de los valores *precision* y *recall* y se calcula con la siguiente formula:

$$\mathrm{F1} = 2\frac{\mathrm{P} \times \mathrm{R}}{\mathrm{P} + \mathrm{R}}$$

Si seguimos con los datos de los dos ejemplos anteriores, podemos calcular el valor de F1 como de la siguiente manera:

$$\mathrm{F1} = 2\frac{\mathrm{P}\times\mathrm{R}}{\mathrm{P}+\mathrm{R}} = 2\frac{0,77\times0,8}{0,77+0,8} = 0,78$$

El reporte que nos facilita esta función pude aportar información muy útil que complementada con la matriz de confusión puede aportarnos una idea de cómo de bien está funcionando nuestro modelo. Por ejemplo, un valor alto de *precisión* de la clase "phishing [1]" significa que el modelo empleado está siendo muy cuidadoso a la hora de etiquetar cosas como phishing cuando no lo son. Por otro lado, si el valor del indicador *recall* es bajo, indica que el clasificador ha fallado en identificar sitios con phishing por ser demasiado cuidadoso y por consiguiente tendrá un valor de *f1 score* bajo. Este caso se pude ver en los resultados de la primera evaluación realizada.

- ***accuracy_score***, o puntuación de precisión del algoritmo, es un indicador que nos permite ver de manera directa la calidad de las predicciones del algoritmo midiendo cuantas etiquetas calculo correctamente el clasificador del número total de predicciones, o lo que es lo mismo, el número de predicciones correctas que realizó nuestro algoritmo. Esto se calcula comparando las clases verdades de cada conjunto de datos analizado contra las clases predichas del clasificador.

5.1. Resultados iniciales

Después de explicar las métricas de evaluación empleadas, vamos a analizar los resultados de ambas validaciones para el primer modelo generado.

En esta primera fase de evaluación se han utilizado 600 URLs de phishing y 600 URLs de sitios legítimos haciendo un total de 1200 URLs. Este conjunto de datos se ha divido en dos bloques de igual tamaño dejando 600 URLs (entre phishing y sitios legítimos) para el entrenamiento el clasificador y otras 600 URLs (también entre phishing y sitios legítimos) para probar el modelo. El contenido de los dos bloques (entrenamiento y prueba) se ha obtenido de una selección aleatoria de las URLs de partida. Hay que tener en cuenta que, si durante el proceso de extracción de características se produce alguna excepción durante la obtención de la información del dominio asociado debido a problemas de conectividad o directamente a problemas reportados por el servicio *whois* empleado, esta URL se descarta para el entrenamiento del modelo. De ahí que en las métricas resultantes siempre tengamos un conjunto de valores un poco más reducido, 276 sitios legítimos y 261 sitios web con phishing para un total de 537 URLs de pruebas en lugar de las 600 pensadas inicialmente.

Después del entrenamiento del algoritmo se ha generado la siguiente curva ROC que podemos ver en la figura 18:

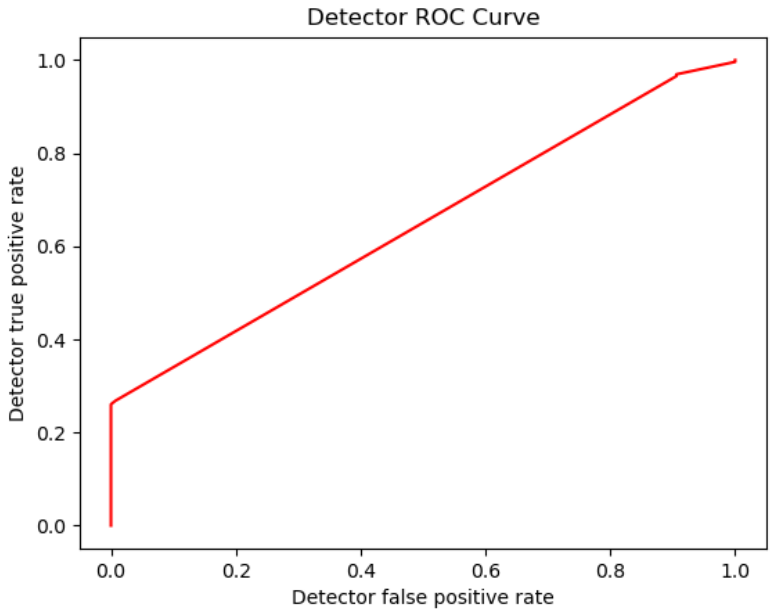

Figura 18. Curva ROC con el resultado del primer mecanismo de evaluación empleado.

Sin necesidad de analizar más indicadores ya se puede intuir que el resultado del entrenamiento del clasificador no ha sido muy bueno ya que la curva generada discurre muy cerca de la diagonal que cruza la gráfica desde la coordenada [0,0] hasta la coordenada [1,1] y que representa la tasa de acierto de un clasificador con el 50 % de éxito.

En cualquier caso si seguimos analizando el resto de métricas realizadas podemos confirmar los malos resultados obtenidos como se aprecia en la figura 19:

```
Mean Absolute Error:0.4124681432046263
Mean Squared Error:0.21472085157798634
Root Mean Squared Error:0.4633798135201687
Confusion Matrix:

[[276   0]
 [194  67]]
Classification Report:

              precision    recall  f1-score   support

           0       0.59      1.00      0.74       276
           1       1.00      0.26      0.41       261

   micro avg       0.64      0.64      0.64       537
   macro avg       0.79      0.63      0.57       537
weighted avg       0.79      0.64      0.58       537

Accuracy_score: 0.638733705772812
```

Figura 19. Métricas calculadas para el primero de los métodos de evaluación empleados.

La figura 20 muestra con un poco más de detalle el contenido de la matriz de confusión resultante de esta primera evaluación:

Matriz de Confusión

	Clasificadas como Legítimas	Clasificadas como Phishing
Páginas Legítimas [0]	Verdaderos Negativos = 276	Falsos Positivos = 0
Páginas con Phishing [1]	Falsos Negativos = 194	Verdaderos Positivos = 67

Figura 20. Matriz de confusión del primero de los métodos de evaluación empleados.

Con los datos de los indicadores calculados podemos concluir que el clasificador generado funciona bien identificando sitios web legítimos, pero sin embargo falla bastante a la hora de detectar páginas web con phishing. El clasificador identifico erróneamente 194 URLs como sitios legítimos cuando no lo eran y solamente acertó a identificar 67 de las 261 páginas con phishing. Como se ha mencionado anteriormente, si nos fijamos en el reporte de clasificación, el valor *f1-score* para la clase phishing [1] es bastante bajo corroborando que nuestro clasificador no es demasiado bueno identificando sitios web maliciosos. De hecho, el valor de precisión calculado para la clase phishing [1] es 1.0 lo que indica que el clasificador no ha sido capaz de identificar como phishing ninguna de las páginas legítimas y al tener un valor bajo de *recall* (0.26) se entiende que los resultados de la identificación de sitios phishing es poco relevante.

Las siguientes ecuaciones explican en detalle el proceso por el que *scikit-learn* obtiene los valores más representativos de la evaluación del clasificador desarrollado. En primer lugar, para la clase phishing [1]:

$$\mathrm{P}[1] = \frac{\mathrm{Tp}\ [1]}{\mathrm{Tp}\ [1] + \mathrm{Fp}[1]} = \frac{67}{67 + 0} = 1$$

$$\mathrm{R}[1] = \frac{\mathrm{Tp}\ [1]}{\mathrm{Tp}[1] + \mathrm{Fn}[1]} = \frac{67}{67+194} = 0,26$$

$$F1[1] = 2\frac{P[1]\times R[1]}{P[1]+R[1]} = 2\frac{1\times 0{,}26}{1+0{,}26} = 0{,}41$$

Si repetimos los mismos cálculos para la clase legitima [0] obtenemos los siguientes valores:

$$P[0] = \frac{Tp\ [0]}{Tp\ [0] + Fp[0]} = \frac{267}{267 + 194} = 0,59$$

$$R[0] = \frac{Tp\ [0]}{Tp[0] + Fn[0]} = \frac{267}{267+0} = 1$$

$$F1[0] = 2\frac{P[0]\times R[0]}{P[0]+R[0]} = 2\frac{0{,}59\times 1}{0{,}59+1} = 0{,}74$$

Por último, hemos obtenido el valor de puntuación de la precisión del clasificador invocando a la función *accuray_score*. Esta función devuelve el tanto por cierto de elementos de phishing identificados del total de URLs de phishing. En esta primera versión del clasificador, obtenemos un valor de precisión (S) del **64 %.** Este valor se ha calculado sumando el número de URLs etiquetadas como legítimas cuando lo eran (Tn) más la suma del número de URLs etiquetadas como phishing cuando realmente lo eran (Tp) entre el número total de URLs que se han etiquetado. La siguiente formula lo expresa con los datos de esta primera evaluación:

$$S = \frac{Tn + Tp}{Tn + Fn + Tp + Fp} = \frac{276 + 67}{276 + 194 + 67 + 0} = 0,64$$

A continuación, vamos a analizar los resultados obtenidos durante la evaluación cruzada para la que se han empleado tres subconjuntos de datos. Al igual que en el apartado anterior, hemos utilizado 600 URLs con sitios clasificados como phishing y 600 URLs legítimas . Lo primero que hacemos es generar la curva ROC de cada uno de los tres subconjuntos sobre los que se han dividido los datos de partida como se puede apreciar en la figura 21:

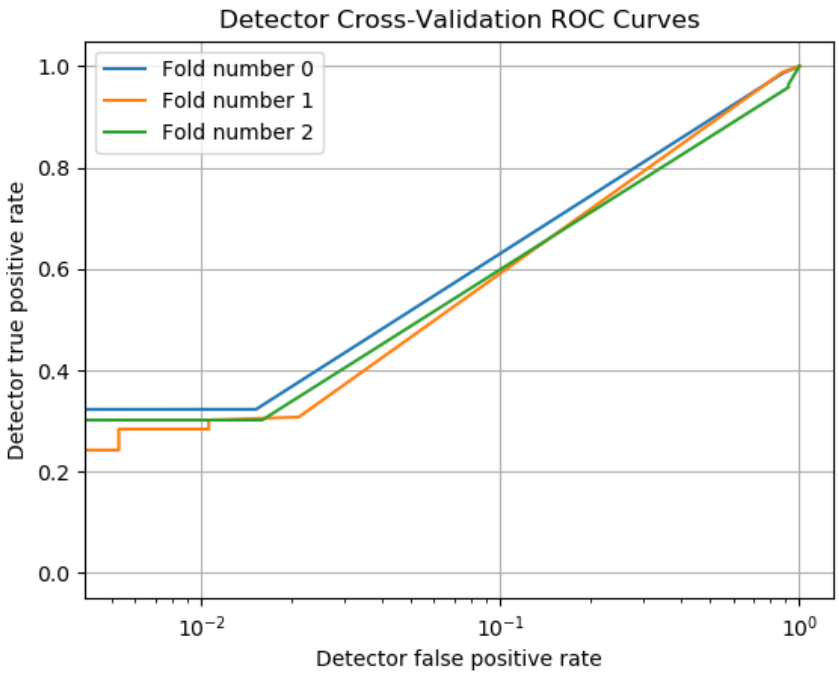

Figura 21. Diagrama con las tres curvas ROC calculadas durante el proceso de evaluación cruzada.

En este punto es conveniente recordar que para la realización de este método de evaluación los datos de entrenamiento y prueba utilizados se han divido en tres subconjuntos aleatorios de igual tamaño con aproximadamente 200 sitios cada uno teniendo en cuanta que algunas de las direcciones URL pueden ser excluidas del experimento debido a las posibles excepciones que tengan lugar durante el proceso de extracción de características al igual que en el caso de evaluación anterior. Las tres curvas ROC que obtenemos son bastante parecidas entre sí y como ocurría anteriormente vemos que las 3 curvas transcurren cerca de la diagonal asociada con el 50 % de acierto siendo las variaciones entre las tres curvas fruto del resultado de entrenar y probar el algoritmo con tres subconjuntos de datos aleatorios.

Durante este tipo de validación cruzada, el programa desarrollado muestra los valores de cada uno de los indicadores de evaluacion empleados para cada uno de los tres experimentos de evaluación que se han realizado. La figura 22 muestra los resultado obtenidos para el primero de los subconjuntos aleatorios generados para el que se han empleado 197 URLs legítimas y 161 URLs con contenido malicioso.

```
Mean Absolute Error:0.4280460591623799
Mean Squared Error:0.20560543923910563
Root Mean Squared Error:0.4534373597743195
[[197   0]
 [115  46]]
              precision    recall  f1-score   support

           0       0.63      1.00      0.77       197
           1       1.00      0.29      0.44       161

   micro avg       0.68      0.68      0.68       358
   macro avg       0.82      0.64      0.61       358
weighted avg       0.80      0.68      0.63       358

0.6787709497206704
```

Figura 22. Métricas calculadas durante la evaluación del primer subconjuntos de datos.

La figura 23 muestra los resultados obtenidos para el segundo de los subconjuntos aleatorios generados para la que se ha utilizado 189 URLs legítimas y 169 URLs maliciosas:

```
Mean Absolute Error:0.4236248521516733
Mean Squared Error:0.20392188329313882
Root Mean Squared Error:0.4515771066973378
[[185   4]
 [117  52]]
              precision    recall  f1-score   support

           0       0.61      0.98      0.75       189
           1       0.93      0.31      0.46       169

   micro avg       0.66      0.66      0.66       358
   macro avg       0.77      0.64      0.61       358
weighted avg       0.76      0.66      0.62       358

0.6620111731843575
```

Figura 23. Métricas calculadas durante la evaluación del segundo subconjunto de datos.

Por último, la figura 24 muestra los resultados obtenidos para el tercero de los subconjuntos aleatorios para la que se ha empleado 188 URLs legítimas y 169 URLs maliciosas:

```
Mean Absolute Error:0.4248652245833601
Mean Squared Error:0.20824318066009087
Root Mean Squared Error:0.45633669659593545
[[185   3]
 [118  51]]
              precision    recall  f1-score   support

           0       0.61      0.98      0.75       188
           1       0.94      0.30      0.46       169

   micro avg       0.66      0.66      0.66       357
   macro avg       0.78      0.64      0.61       357
weighted avg       0.77      0.66      0.61       357

0.6610644257703081
```

Figura 24. Métricas calculadas durante la evaluación del tercer subconjunto de datos.

Como se puede observar, los resultados de la evaluación cruzada vienen a confirmar los resultados obtenidos durante la primera evaluación. Esto tiene sentido ya que esta técnica de evaluación cruzada emplea los mismos métodos de evaluación solo que sobre conjuntos diferentes de datos. El hecho de que el experimento repetido sobre los tres subconjuntos de datos implica que en el total de URLs empleadas es bastante homogéneo.

Con toda esta información de las dos evaluaciones podemos confirmar que el algoritmo desarrollado hasta el momento no es demasiado preciso a la hora de identificar sitios web maliciosos llegando únicamente a unos valores de precisión de entorno al 66 % de acierto algo que dentro de los algoritmos de aprendizaje automático no se puede considerar como muy aceptable.

5.2. Mejorando el rendimiento

Llegados a este punto y con el objetivo de mejorar los resultados, tenemos dos vías de trabajo que vamos a explorar.

En primer lugar, es necesario ampliar el volumen de datos sobre los que se realiza el entrenamiento del algoritmo. Para ello ha sido necesario ampliar el contenido del fichero CSV de sitios legítimos incluyendo muchas más direcciones URL de páginas web legítimas a partir del ranking de los 50 sitios más populares por país que se publica en la página web [ALEXATOPSITES20]. En cuanto a los sitios maliciosos, el fichero que descargamos de *PhishTank* contiene datos de sobra por lo que no es un problema. Para evitar que el proceso de entrenamiento sea demasiado largo vamos a limitar el número de URLs de cada uno de los ficheros CSV que se procesen hasta un máximo de 10.000 líneas.

La segunda forma de mejorar nuestro algoritmo es mejorando la selección de características que empleamos para el entrenamiento del clasificador. Para este segundo experimento, además de las características originales vamos a incluir las siguientes características nuevas:

- Longitud de la dirección URL. Si analizamos la longitud de las direcciones URL y la longitud de las direcciones de los sitios maliciosos vemos que por regla general las direcciones legítimas no son demasiado largas ya que los dominios que las empresas utilizan suelen ser cortos para que sean más fáciles de recordar y sin embargo dentro de las direcciones maliciosas es bastante habitual que se utilice un nombre de dominio real al que se le han añadido más palabras para tratar de engañar a los usuarios. Analizando el tamaño medio de los sitios web analizados hemos delimitado que si la dirección URL tiene más de 54 caracteres es sospechosa de tratarse de un sitio web malicioso.
- El uso de servicios de recorte de URL. Otra de las técnicas que se pueden utilizar para engañar a una posible víctima es la de utilizar algún servicio para recortar una dirección disponible en internet como pueda ser tinyurl.com. Por este motivo si la dirección esa recortada la clasificaremos como sospechosa de ser phishing.
- Aparición del carácter "-" dentro de la URL. Es bastante habitual que los atacantes utilicen el carácter "-" para engañar a los usuarios de un sitio web a la hora de construir una dirección URL maliciosa. Por ese motivo, si dentro de una dirección URL aparece un "-" la clasificaremos como sospechosa.
- Tiempo de expiración del dominio. La mayoría de los dominios de internet de las empresas se suelen renovar con bastante antelación para evitar que estos expiren y puedan ser adquiridos por algún otro competidor del mercado. Por ese motivo, si dentro de la información del dominio vemos que este expira dentro de menos de 180 días lo consideraremos como un sitio sospechoso.
- Icono del sitio en un dominio distinto. En ocasiones las páginas web fraudulentas se generan copiando el contenido de páginas legítimas y modificando solo ciertas partes del código HTML como los formularios para que la información de usuario se envíe a algún servidor controlado por los atacantes. Es por ese motive que la dirección donde se encuentra el icono que aparece en la pestaña del navegador se encuentra en un dominio diferente al de la web visitada. Esto es bastante extraño que ocurra en las páginas web legítimas por lo que lo vamos a considerar sospechoso de phishing.
- Uso de elementos *iframe*. El uso de contendores de tipo *iframe* fue durante mucho tiempo utilizado por los ciberdelincuentes para mostrar el contenido de una página web malicioso dentro de otra web que aparentemente es legítima. Por ese motivo y sabiendo que su uso está descontinuado en sitios web actuales y legítimos, vamos a considerarlo como un indicador sospechoso de phishing.

Una vez hemos codificado la nueva versión de la función de extracción de características dentro de nuestro código, es necesario volver a generar el nuevo vector de características para todos los sitios web que vamos a utilizar. En esta ocasión el proceso es mucho más lento debido a que el número de características ha pasado de 12 a 18 y además el número total de sitios web maliciosos y legítimos ha pasado de 1200 a casi 19000. Por estos motivos, han sido necesarias casi 14 horas el obtener el vector **X** con todas las características. Finalizado el proceso de extracción de características se ha obtenido un vector **X** con un total de **18387** direcciones URL de las cuales **9415** pertenecían a sitios maliciosos y **8972** a sitios legítimos.

Repetimos el proceso de evaluación, primero con el método inicial que divide los datos en dos conjuntos. En esta ocasión vamos a destinar el 80 % de los datos en datos de entrenamiento y 20 % restante en datos de prueba. La figura 25 muestra la nueva curva ROC que se ha obtenido.

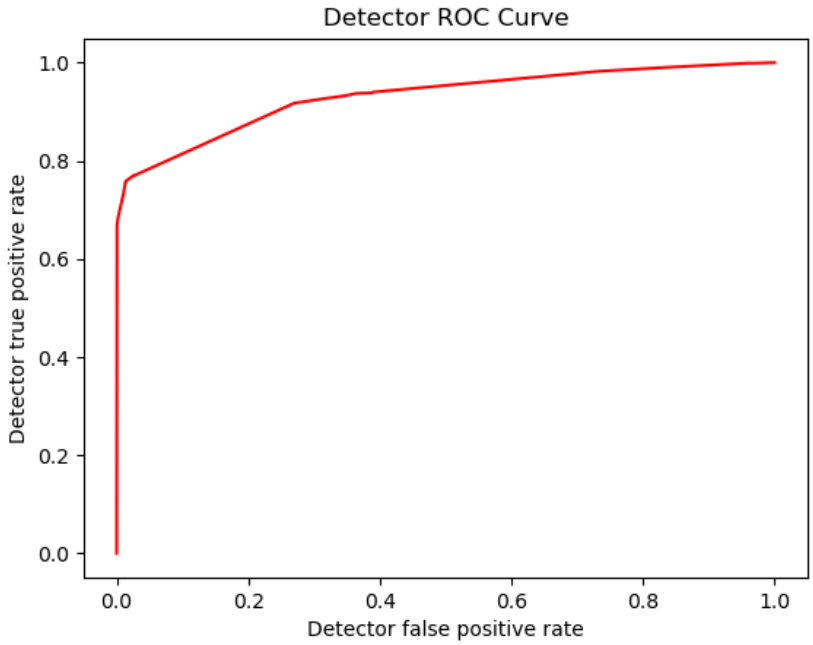

Figura 25. Curva ROC con el resultado del nuevo modelo.

En esta ocasión parece que la curva es mucho mejor, se aleja de la diagonal del 50 % de aciertos y está mucho más cerca de la curva ROC ideal, la que parte del punto [0,0] va hasta el punto [0,1] y termina en el punto [1,1]. Vamos a analizar los valores de los indicadores de evaluación que se han calculado para comprobar si realmente hemos mejorado el rendimiento tal y como parece indicar la curva ROC generada.

En la figura 26 se muestran los indicadores calculados durante la evaluación donde podemos confirmar la mejora experimentada. En esta ocasión hemos creado un clasificador que tiene una precisión con una puntuación del **87 %**:

```
Mean Absolute Error:0.18917304045882896
Mean Squared Error:0.09193799377407649
Root Mean Squared Error:0.3032127862971423
Confusion Matrix:

[[1819   27]
 [ 442 1390]]
Classification Report:

              precision    recall  f1-score   support

           0       0.80      0.99      0.89      1846
           1       0.98      0.76      0.86      1832

   micro avg       0.87      0.87      0.87      3678
   macro avg       0.89      0.87      0.87      3678
weighted avg       0.89      0.87      0.87      3678

Accuracy_score: 0.8724850462207722
```

Figura 26. Métricas del clasificador final donde se aprecia la mejora de los resultados obtenidos.

Si analizamos la matriz de confusión resultante en mayor detalle como vemos en la figura 27:

Matriz de Confusión

	Clasificadas como Legítimas	Clasificadas como Phishing
Páginas Legítimas [0]	Verdaderos Negativos = 1819	Falsos Positivos = 27
Páginas con Phishing [1]	Falsos Negativos = 442	Verdaderos Positivos = 1390

Figura 27. Matriz de confusión del nuevo clasificador generado.

Las siguientes ecuaciones explican en detalle el proceso por el que *scikit-learn* obtiene los valores más representativos de la evaluación del clasificador desarrollado. En primer lugar, para la clase phishing [1]:

$$\mathrm{P}[1] = \frac{\mathrm{Tp}\ [1]}{\mathrm{Tp}\ [1] + \mathrm{Fp}[1]} = \frac{1390}{1390 + 27} = 0{,}98$$

$$\mathrm{R}[1] = \frac{\mathrm{Tp}\ [1]}{\mathrm{Tp}[1] + \mathrm{Fn}[1]} = \frac{1390}{1390+442} = 0{,}76$$

$$\mathrm{F1}[1] = 2\frac{\mathrm{P}[1]\times\mathrm{R}[1]}{\mathrm{P}[1]+\mathrm{R}[1]} = 2\frac{0{,}98\times0{,}76}{0{,}98+0{,}76} = 0{,}86$$

Si repetimos los mismos cálculos para la clase legitima [0] obtenemos los siguientes valores:

$$\mathrm{P}[0] = \frac{\mathrm{Tp}\ [0]}{\mathrm{Tp}\ [0] + \mathrm{Fp}[0]} = \frac{1819}{1819 + 442} = 0{,}80$$

$$\mathrm{R}[0] = \frac{\mathrm{Tp}\ [0]}{\mathrm{Tp}[0] + \mathrm{Fn}[0]} = \frac{1819}{1819+27} = 0{,}99$$

$$\mathrm{F1}[0] = 2\frac{\mathrm{P}[0]\times\mathrm{R}[0]}{\mathrm{P}[0]+\mathrm{R}[0]} = 2\frac{0{,}80\times0{,}99}{0{,}80+0{,}99} = 0{,}89$$

Ahora vamos a repetir el proceso de validación pero utilizando esta vez la validación cruzada. La figura 28 muestra las cinco curvas ROC de los subconjuntos de datos aleatorios empleados durante esta evaluación. Se puede apreciar como en el caso anterior que las curvas tienen todas bastante similitud por lo que los datos empleados son bastante homogéneos tanto para el entrenamiento como para las pruebas.

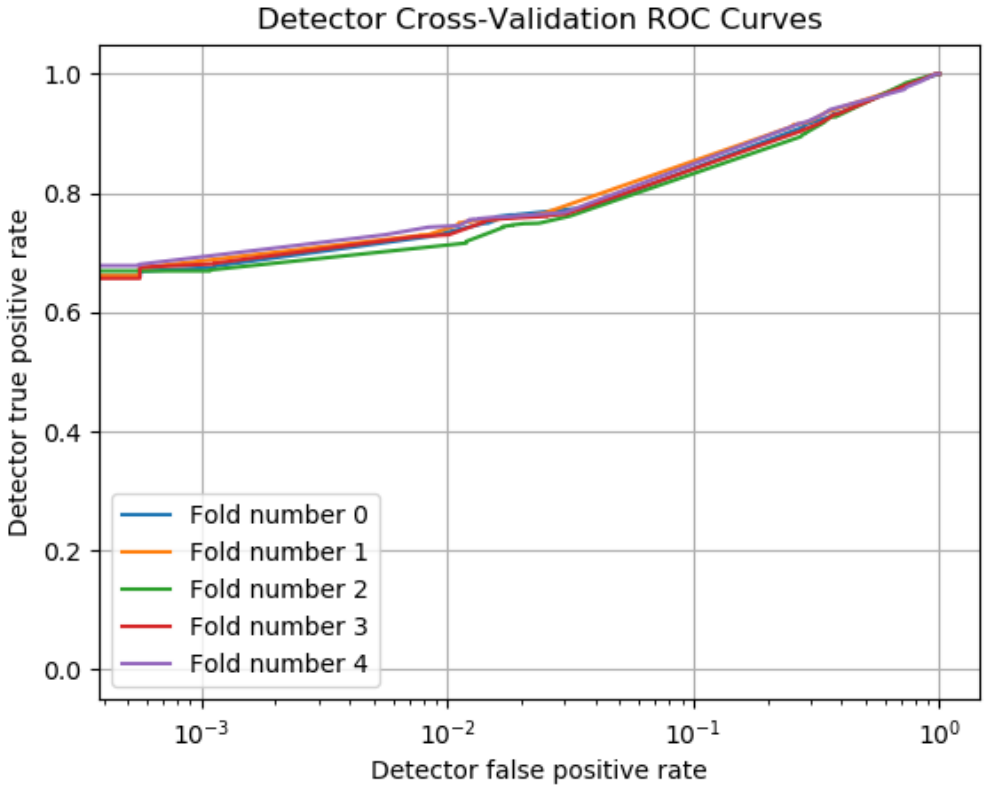

Figura 28. Gráfico con las cinco curvas ROC calculadas durante el proceso de validación cruzada final.

Por último, vamos a estudiar mediante un mapa de calor la correlación existente entre cada una de las características empleadas durante la fase de entrenamiento del clasificador haciendo uso de las librerías *pandas* y *seaborn* de Python. Para ello, lo que hacemos es crear una estructura de tipo *DataFrame* con el contenido de todos los vectores de características junto con la clase que tienen para cada una de las URLs utilizadas. La figura 29 muestra el mapa de calor generado para todas las características:

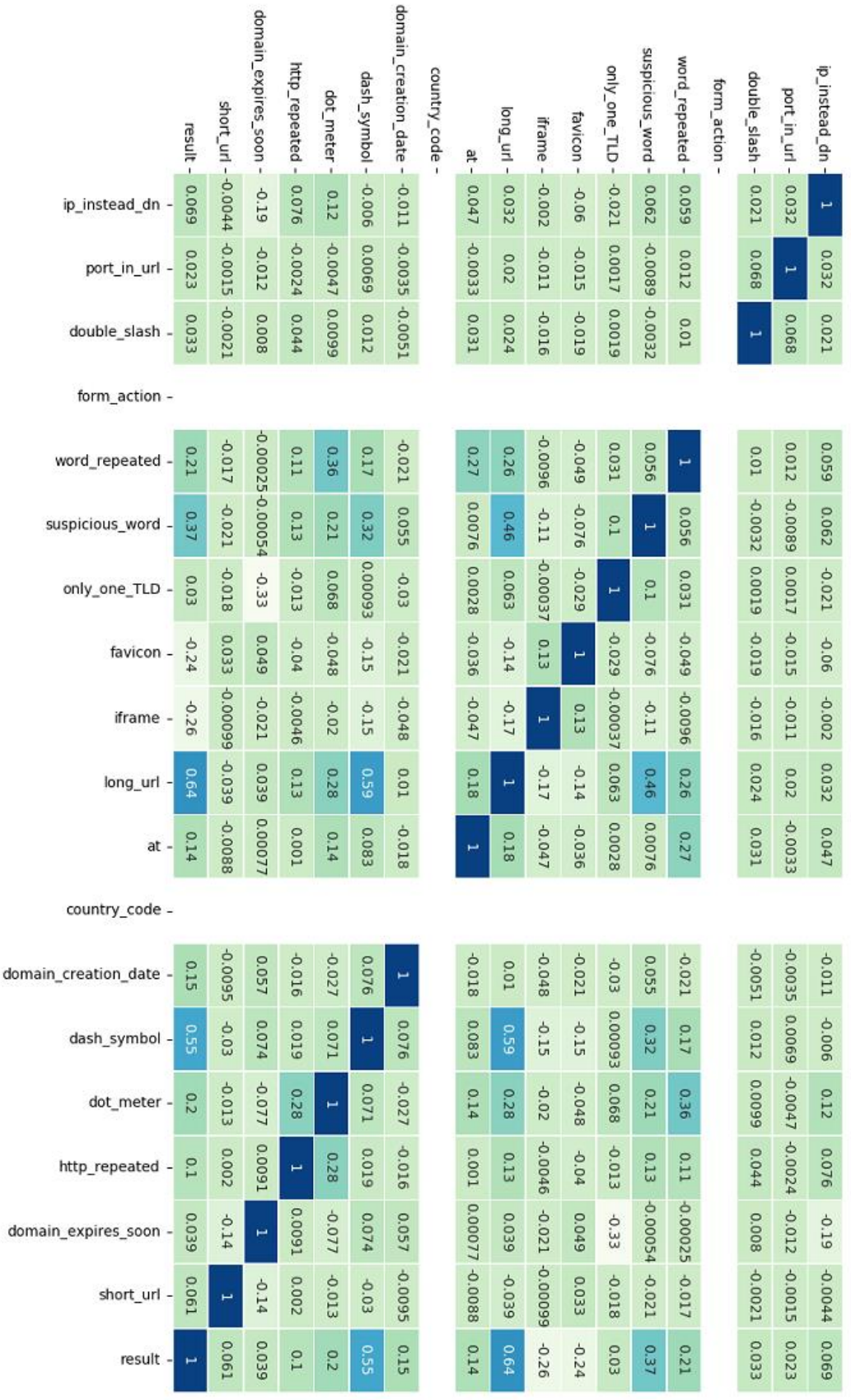

Figura 29. Mapa de calor de la correlación de las características empleadas.

Llama la atención que hay dos características cuya correlación no se ha podido calcular. Si analizamos en detalle el contenido del vector de características como vemos en la figura 30, todas las características tienen valor:

```
Data columns (total 19 columns):
 #   Column               Non-Null Count  Dtype
---  ------               --------------  -----
 0   ip_instead_dn        18387 non-null  int64
 1   port_in_url          18387 non-null  int64
 2   double_slash         18387 non-null  int64
 3   form_action          18387 non-null  int64
 4   word_repeated        18387 non-null  int64
 5   suspicious_word      18387 non-null  int64
 6   only_one_TLD         18387 non-null  int64
 7   favicon              18387 non-null  int64
 8   iframe               18387 non-null  int64
 9   long_url             18387 non-null  int64
 10  at                   18387 non-null  int64
 11  country_code         18387 non-null  int64
 12  domain_creation_date 18387 non-null  int64
 13  dash_symbol          18387 non-null  int64
 14  dot_meter            18387 non-null  int64
 15  http_repeated        18387 non-null  int64
 16  domain_expires_soon  18387 non-null  int64
 17  short_url            18387 non-null  int64
 18  result               18387 non-null  int64
dtypes: int64(19)
memory usage: 2.7 MB
```

Figura 30. Información del vector de características empleado para la generación del modelo.

Como podemos ver en la figura 31, el problema consiste en que estas dos características (*country_code* y *form_action*) siempre tienen valor 0 luego durante la fase de extracción de características del algoritmo no se ha encontrado ninguna página web del conjunto de datos de entrenamiento empleado que cumpliera los requisitos de análisis de estas dos características, es decir, no se han encontrado sitios web donde el código del país que viene especificado en el dominio no coincida con la información de geolocalización de la dirección IP ni sitios web que incluyan formularios que apunte a dominios diferentes del dominio donde se encuentran alojadas los sitios.

```
>>> df['form_action'].value_counts()
0    18387
Name: form_action, dtype: int64
>>> df['country_code'].value_counts()
0    18387
Name: country_code, dtype: int64
```

Figura 31. Características no encontradas durante la fase de entrenamiento.

Aparte de esto último, el mapa de calor nos muestra que la mínima correlación llega hasta -0,33. La correlación negativa entre dos características como son la de un único domino TLD y los dominios que expiran en menos de 180 días significa que una de las características está muy asociada con sitios legítimos y la otra con sitios etiquetados como phishing y que si un sitio web tiene más de un TLD registrado, lo normal es que su dominio no expire pronto.

La figura 32 nos muestra que los sitios legítimos tienen en su gran mayoría un único TLD dentro de su URL y que los sitios con phishing tienen registrados dominios que caducan en menos de 180 días:

```
>>> df['only_one_TLD'].value_counts()
0    16921
1     1466
Name: only_one_TLD, dtype: int64
>>> df['domain_expires_soon'].value_counts()
1    16055
0     2332
Name: domain_expires_soon, dtype: int64
```

Figura 32. Detalle de alguna de las características con menor correlación entre ellas.

Por último, cabe destacar que se ha probado a implementar el algoritmo con otros de los clasificadores que ofrece *scikit-learn* con el mismo modelo de datos para comprobar si la precisión de estos mejoraban o empeoraban y tal y como se puede observar en la figura 33 la tasa de precisión es igual:

```
Random Forest Accuracy_score: 0.8675910821098423
SVC Accuracy_score: 0.8675910821098423
KNeighborsClassifier Accuracy_score: 0.8675910821098423
```

Figura 33. Precisión obtenida con distintos clasificadores disponibles en scikit-learn

Llegados a este punto hemos desarrollado un algoritmo de detección de phishing con una tasa de acierto de 87 %. Tras repasar el modelo publicado en [GowthamKrishnamurthib2014], considero que es posible mejorar la tasa de acierto de nuestro algoritmo si se utiliza algún tipo de medida basada en información de un tercero con una gran reputación como Google. Los autores del estudio utilizaron *PageRank,* pero desde hace años esta herramienta no está disponible. Sin embargo, es posible hacer uso del servicio *Google Safe Browsing* [GOOGLESAFEBROWSING20] que nos permite hacer llamadas a una API publica gratuita con un límite de 10.000 peticiones cada 24 horas para saber si una URL está clasificada como phishing por la propia Google. Para poder hacer uso de esta nueva característica, ha sido necesario generar un nuevo modelo e invocar para cada una de las URLs analizadas un nuevo valor, 1 en caso de que Google considere la URL como maliciosa y un 0 en caso contrario.

Vamos a evaluar de nuevo el clasificador y a analizar los resultados obtenidos. En esta ocasión hemos generado un modelo con 4290 URLs, con un total de 2024 direcciones URL con phishing y 2266 direcciones legítimas. Se ha utilizado un modelo más pequeño reducir el tiempo de extracción de características hasta las 5 horas.

Primero, como se puede ver en la matriz de correlación de la figura 34, se puede observar que la nueva característica (*googlesafe)* está fuertemente relacionada con la etiqueta phishing:

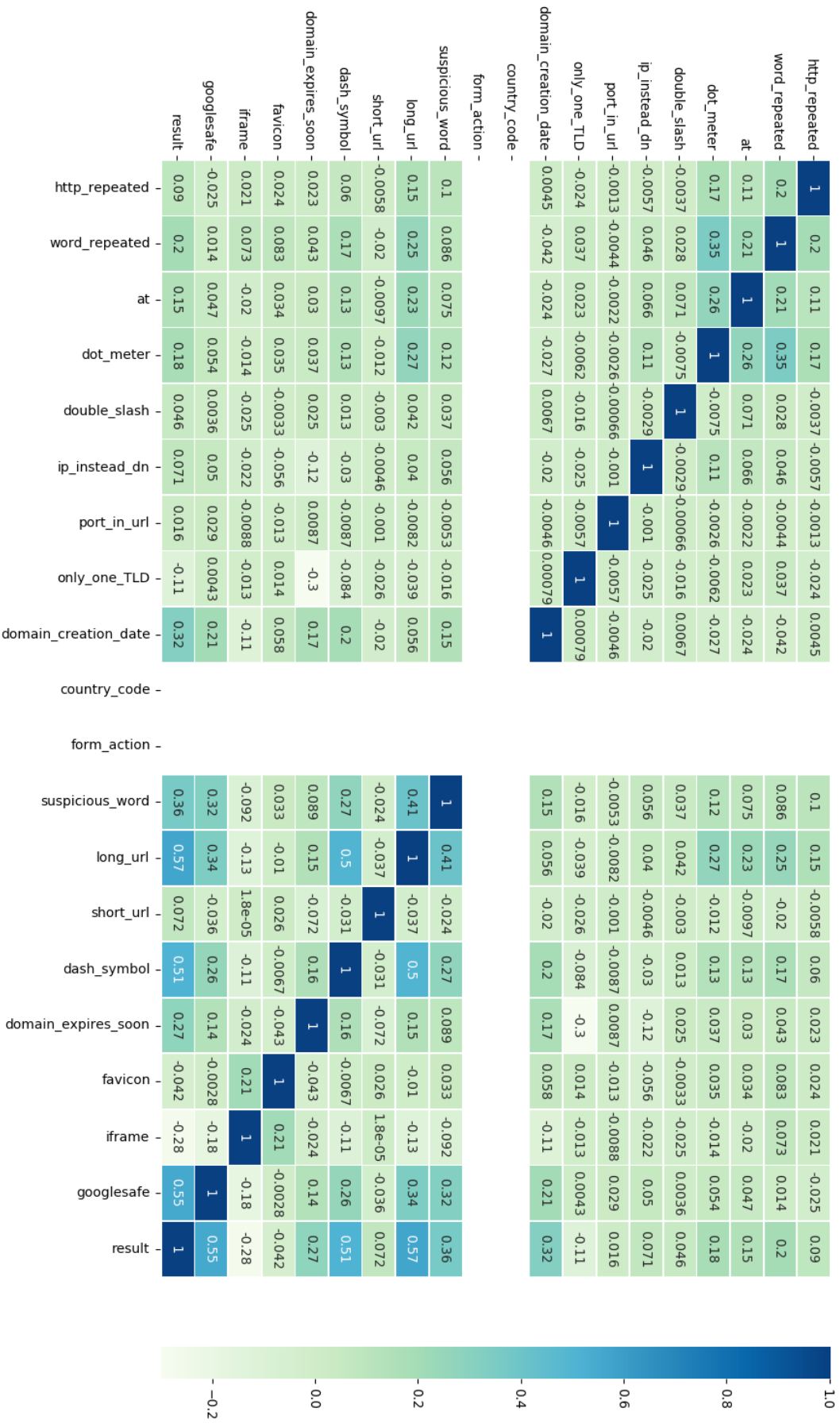

Figura 34 Matriz de correlación que incluye la nueva característica googlesafe.

En la figura 35 se puede apreciar la mejora obtenida al incluir esta nueva característica a las que ya formaban el vector anterior:

```
[[429  11]
 [ 62 356]]
Classification Report:

              precision    recall  f1-score   support

           0       0.87      0.97      0.92       440
           1       0.97      0.85      0.91       418

   micro avg       0.91      0.91      0.91       858
   macro avg       0.92      0.91      0.91       858
weighted avg       0.92      0.91      0.91       858

Accuracy_score: 0.914918414918415
```

Figura 35. Indicadores de la evaluación realizada para el modelo con la característica googlesafe.

Por último, vamos a analizar la matriz de confusión resultante en mayor detalle como vemos en la figura 36:

Matriz de Confusión

	Clasificadas como Legítimas	Clasificadas como Phishing
Páginas Legítimas [0]	Verdaderos Negativos = 429	Falsos Positivos = 11
Páginas con Phishing [1]	Falsos Negativos = 62	Verdaderos Positivos = 356

Figura 36 Matriz de confusión generada para el modelo con la característica googlesafe.

Las siguientes ecuaciones explican en detalle el proceso por el que *scikit-learn* obtiene los valores más representativos de la evaluación del clasificador desarrollado. En primer lugar, para la clase phishing [1]:

$$\mathrm{P}[1] = \frac{\mathrm{Tp}\,[1]}{\mathrm{Tp}\,[1] + \mathrm{Fp}[1]} = \frac{356}{356 + 11} = 0{,}97$$

$$\mathrm{R}[1] = \frac{\mathrm{Tp}\,[1]}{\mathrm{Tp}[1] + \mathrm{Fn}[1]} = \frac{356}{356+62} = 0{,}85$$

$$\mathrm{F1}[1] = 2\frac{\mathrm{P}[1]\times\mathrm{R}[1]}{\mathrm{P}[1]+\mathrm{R}[1]} = 2\frac{0{,}97\times0{,}85}{0{,}97+0{,}85} = 0{,}91$$

Si repetimos los mismos cálculos para la clase legitima [0] obtenemos los siguientes valores:

$$\mathrm{P}[0] = \frac{\mathrm{Tp}\,[0]}{\mathrm{Tp}\,[0] + \mathrm{Fp}[0]} = \frac{429}{429 + 62} = 0,87$$

$$\mathrm{R}[0] = \frac{\mathrm{Tp}\,[0]}{\mathrm{Tp}[0] + \mathrm{Fn}[0]} = \frac{429}{429+11} = 0{,}97$$

$$\mathrm{F1}[0] = 2\frac{\mathrm{P}[0]\times\mathrm{R}[0]}{\mathrm{P}[0]+\mathrm{R}[0]} = 2\frac{0{,}87\times0{,}97}{0{,}87+0{,}97} = 0{,}92$$

En esta ocasión hemos obtenido una tasa de acierto en el clasificador del **91 %**.

6. Conclusiones y Trabajo futuro

Para la realización de este TFM se ha estudiado en profundidad la problemática que representan los ataques de ingeniería social poniendo el foco en el tipo de ataque más extendido entre los cibercriminales, el Phishing. Hemos analizado el impacto económico que tiene en la sociedad digital en la que vivimos y las principales medidas de seguridad existentes en la actualidad diseñadas para reducir el impacto negativo que tiene un ataque de estas características cuando se materializa. De entre todas las técnicas, la más prometedora es sin duda la que está basada en algoritmos de *Machine Learning* ya que permite diseñar programas muy eficaces a la hora de detectar estas amenazas como se ha podido comprobar durante el caso práctico desarrollado.

La existencia de un lenguaje de programación tan sencillo y a la vez tan potente como Python y sobre todo gracias a la gran cantidad de librerías gratuitas destinadas al campo de la inteligencia artificial y al mundo del aprendizaje automático hacen posible desarrollar algoritmos capaces de identificar y clasificar páginas web maliciosas con tasas de acierto de más del 90 %.

Como se ha podido ver durante los capítulos que describen el caso práctico, cuanto mejores sean las características empleadas para el entrenamiento del clasificador mayor será su tasa de acierto. Por otro lado, también es importante destacar la importancia que tiene dentro del mundo del aprendizaje automático el poder trabajar con volúmenes de datos de prueba y de entrenamiento suficientemente grandes. Debido a las limitaciones del ordenador donde se han realizado los experimentos se han limitado los experimentos a 20000 direcciones URL.

Una de las principales diferencias entre características del modelo desarrollado en este TFM y las reportados en el artículo de [GowthamKrishnamurthib2014] son las que se basan en indicadores de terceros. En concreto me estoy refiriendo a la heurística del indicador del algoritmo *PageRank* de *Google*. Se trata de un indicador que permite al clasificador valorar la importancia en internet de una página web a partir de la puntuación del 0 al 10 que le otorga el buscador *Google*. Partiendo de la base de que esa información proviene de un gigante de internet como Google, podemos asumir que la credibilidad que otorga dicha característica a un sitio web es un estimador bastante importante a la hora de clasificar una URL como legitimo o no. El problema que me he encontrado durante la realización de este TFM es que el servicio de *PageRank* fue deshabilitado hace ya varios años por la propia *Google* con la intención de que los creadores de contenido de internet no se centraran únicamente en diseñar sitios web con el objetivo de obtener puntuaciones altas de *PageRank* por lo que no ha sido posible incluirlo dentro del modelo de clasificación desarrollado. A pesar de eso, en un último paso del caso práctico llevado a cabo se han podido mejorar los resultados de precisión del algoritmo de clasificación utilizando una medida características similares que aporta información de un tercero con una gran reputación como *Google* a través de su servicio *Google Safe Browsing*.

Para el futuro sería posible mejorar los resultados obtenidos entrenando el algoritmo para que trabaje con un volumen mayor de datos y puede que incluyendo alguna característica nueva al vector de características. Teniendo en cuanta que el proceso de extracción de características con el que se ha trabajado durante la realización de este TFM ha necesitado más de 14 horas para terminar por lo que sería interesante poder repetir los experimentos en otra infraestructura más potente, o bien desarrollar algún tipo de concurrencia para analizar de forma paralela el conjunto de sitios web empleados para el entrenamiento del algoritmo.

La seguridad total de un sistema de detección de este tipo de amenazas nunca será completa ya que los delincuentes dedican todo sus esfuerzos a burlar las medidas de seguridad que impiden realizar este tipo de actividades fraudulentas. En cualquier caso, los porcentajes de existo de un sistema de detección y prevención de este tipo de ataques serán mayores si se combinan las tres ramas de defensa explicadas en este TFM: los sistemas tradicionales de detección y prevención, los sistemas automatizados de detección de phishing basados en algoritmos de *Machine Learning* y la formación continuada a los usuarios de los sistemas de información.

Bibliografía

[Bishop2003] BISHOP, M. (2003): What is computer security? IEEE Security & Privacy, vol. 1, no. 1, p67-69.

[QabajehThabtahChiclana2018] QABAJEH, I., THABTAH, F., CHICLANA, F. (2018): A recent review of conventional vs. automated cybersecurity anti-phishing techniques, Computer Science Review, vol. 29, p44-55.

[GowthamKrishnamurthib2014] GOWTHAM, R., KRISHNAMURTHIB, I. (2014): A comprehensive and efficacious architecture for detecting phishing webpages, Computers & Security, vol. 40, p23-37.

[VashneyMisraPradeep2018] VASHNEY, G., MISRA, M., PRADEEP, A. (2018): Secure authentication scheme to thwart RT MITM, CR MITM and malicious browser extension-based phishing attacks, Journal of Information Security and Applications, vol. 42, p1-17.

[VayanskiSathish2018] VAYANSKY, I., SATHISH, K. (2018). Phishing – challenges and solutions, Computer Fraud & Security, vol. 2018, issue 1, p15-20.

[ArachchilageLove2013] ARACHCHILAGE, NAG., LOVE, S. (2013): A game design framework for avoiding phishing attacks, Computers in human behaviour, vol. 29, issue 3, p706–714.

[ArachchilageRheeShengHasan2007] ARACHCHILAGE, NAG., RHEE, Y., SHENG, S., HASAN, SH., ACQUISTI, A., CRANOR, LF., HONG, J. (2007): Getting users to pay attention to anti-phishing education: evaluation of retention and transfer, ECrime, p70–81.

[RolandCurtisAaron2007] RONALD, DJC., CURTIS, C., AARON, FJ. (2007): Phishing for user security awareness, Computer Security, vol. 26, issue 1, p73-80.

[GaretaProvosChevRubin2007] GARETA, S., PROVOS, N., CHEV, M., RUBIN, A. (2007): A framework for detection and measurement of phishing attacks, Association for Computing Machinery, WORM 2007, p1-8.

[MalwareDataScience2018] SAXE, J., SANDERS, H. (2018): Malware Data Science. Attack Detection and Attribution, Septiembre 2018, 272 pp. ISBN-13: 978-1-59327-859-5

Enlaces

[NIST20] NIST. "National Vulnerability Database". { https://nvd.nist.gov/vuln/search/statistics?form_type=Basic&results_type=statistics&search_type=all } – 5 Junio 2020.

[INCIBE20] INCIBE. "Ingeniería social, técnicas usadas por los ciberdelincuentes y cómo protegerse". { https://www.incibe.es/protege-tu-empresa/blog/ingenieria-social-tecnicas-utilizadas-los-ciberdelincuentes-y-protegerse } - 10 Julio 2020.

[NORTON20] Norton. "What is social engineering". { https://mx.norton.com/internetsecurity-emerging-threats-what-is-social-engineering.html } - 10 Julio 2020.

[OSI20_1] Oficina de Seguridad del Internauta. "Conoce a fondo qué es el phishing". { https://www.osi.es/es/banca-electronica } - 14 Julio 2020.

[OSI20_2] Oficina de Seguridad del Internauta. "El phishing, la moda que nunca pasa". { https://www.osi.es/es/actualidad/blog/2016/03/15/el-phishing-la-moda-que-nunca-pasa } - 14 Julio 2020.

[OSI20-3] Oficina de Seguridad del Internauta. "Avisos publicados". { https://www.osi.es/es/actualidad/avisos } - 14 Julio 2020.

[OSI20-4] Oficina de seguridad del Internauta. "Campaña fraudulenta relacionada con la pandemia COVID19". { https://www.osi.es/es/actualidad/avisos/2020/05/identificados-sms-fraudulentos-con-enlace-una-web-para-solicitar-una } - 14 Julio 2020.

[FBI20] FBI. "Internet Crime Complaint Center". { https://www.ic3.gov/media/2018/180712.aspx } - 19 Julio 2020.

[CYBRIANT20] Cybriant. "Astonishing Cybercrime Statistics in 2019". { https://cybriant.com/2019-cybercrime-statistics/ } - 20 Julio 2020.

[FORBES20] Forbes. "Phishing Scams Cost American Businesses Half A Billion Dollars A Year". { https://www.forbes.com/sites/leemathews/2017/05/05/phishing-scams-cost-american-businesses-half-a-billion-dollars-a-year/ } - 11 Julio 2020.

[ACCENTURE20] Accenture. "Ninth Annual Cost of Cybercrime Study". - 11 Julio 2020.

[ISO7498-2] International Organization for Standardization "ISO 7498-2:1989" { https://www.iso.org/obp/ui/#iso:std:iso:7498:-2:ed-1:v1:en } - 11 Julio 2020.

[SCIKITLEARN20] Machine Learning en Python. scikit-learn. { https://scikit-learn.org/stable/index.html } 10 Julio 2020.

[PHISTANK20] Comunidad online para compartir recursos de sitios web clasificados como phishing. { http://phishtank.org/index.php } - 10 Julio 2020.

[MOZ20] Listado con los 500 sitios más populares de internet. { https://moz.com/top500 } - 14 Julio 2020.

[GTMETRIX20] Listado con los 1000 sitios más populares de internet de acuerdo con Google. { https://gtmetrix.com/top1000.html } - 14 Julio 2020.

[REPOSITORIO20] Repositorio donde se encuentra el código fuente desarrollado para el caso de estudio de este TFM. { https://bitbucket.org/DuB0k/tfm_phishing/src/master/ } – 5 Septiembre 2020.

[GOOGLESAFEBROWSING20] Api publica donde se puede consultar si un sitio web ha sido clasificado como Phishing por los algoritmos de Google. { https://developers.google.com/safe-browsing } - 7 Septiembre 2020.

[ALEXATOPSITES20] Contiene un listado con los 50 sitios web más populares de cada país. { https://www.alexa.com/topsites/countries } - 10 Septiembre 2020.

[GOOGLECLOUD20] Plataforma para desarrolladores de Google donde podemos obtener una clave para utilizar la API de Google Safe Browsing. { https://cloud.google.com/ } – 17 Septiembre 2020.

[IPINFO20] API publica que permite obtener información de una dirección IP con una limitación de 50000 peticiones al mes. { https://ipinfo.io/ } – 17 Septiembre 2020.