HERRAMIENTA PARA IDENTIFICAR VULNERABILIDADES EN APLICACIONES OFIMÁTICAS DE WINDOWS MEDIANTE TÉCNICAS DE PROCESAMIENTO DE LENGUAJE NATURAL

CRISITAN CAMILO TARAZONA BUSTAMANTE

UNIVERSIDAD AUTÓNOMA DE BUCARAMANGA
FACULTAD DE INGENIERÍAS
INGENIERÍA DE SISTEMAS
GRUPO DE INVESTIGACIÓN TECNOLOGÍAS DE INFORMACIÓN - GTI
BUCARAMANGA
2020

HERRAMIENTA PARA IDENTIFICAR VULNERABILIDADES EN APLICACIONES OFIMÁTICAS DE WINDOWS MEDIANTE TÉCNICAS DE PROCESAMIENTO DE LENGUAJE NATURAL

CRISITAN CAMILO TARAZONA BUSTAMANTE

DIRECTOR: YAMID GABRIEL GAMBA GONZALEZ

UNIVERSIDAD AUTÓNOMA DE BUCARAMANGA
FACULTAD DE INGENIERÍAS
INGENIERÍA DE SISTEMAS
BUCARAMANGA
2020

Nota de Aceptación

_____

_____

_____

_____

_____

Firma del jurado

_____

Firma del Jurado

TABLA DE CONTENIDO

LISTA DE TABLAS

RESUMEN EJECUTIVO

Las vulnerabilidades en las aplicaciones son actualmente encontradas por medio de un fallo que es explotado por ciberdelincuentes afectándolas y por ende los datos que estas contienen , con el fin de identificar cuáles son las vulnerabilidades que tiene una aplicación en este caso ofimáticas , se logra haciendo uso de técnicas de procesamiento de lenguaje natural que permiten obtener la frecuencia con que se repite una vulnerabilidad mediante el conteo de sus palabras que son grupadas por n-gramas y lo que permite visualizar la aparición e identificación de vulnerabilidades en las diferentes aplicaciones que son ejecutadas en Microsoft Office como Microsoft Word , Microsoft Excel , Power Point , Skype, Microsoft Teams y Outlook.

The vulnerabilities in the applications are currently found through a failure that is exploited by cybercriminals affecting them and therefore the data they contain, in order to identify themselves are the vulnerabilities that an application has in this case office automation, it is achieved by using natural language processing techniques that allow obtaining the frequency with which a vulnerability is repeated the content of its words by means of which they are grouped by n-grams and which allows visualizing the appearance and identification of vulnerabilities in the different applications that are executed in Microsoft Office such as Microsoft Word, Microsoft Excel, Power Point, Skype, Microsoft Teams and Outlook.

INTRODUCCIÓN

En la actualidad el uso masivo de los equipos informáticos ha hecho que las aplicaciones ofimáticas sean imprescindibles en las labores cotidianas de las personas, ello ha generado que los ciberdelincuentes usen este tipo de aplicaciones para la generación de ataques informáticos, es por esto que se deben generar herramientas que permitan conocer de primera mano las amenazas y vulnerabilidades a las cuales se está expuesto. En este sentido el análisis de la información oportuna y la generación de reportes o estadísticas mediante la aplicación de nuevas tecnologías es de vital importancia para que los administradores de sistemas ofimáticos y los usuarios se sensibilicen sobre las fallas de seguridad informática en sus aplicaciones, con el fin de identificar vulnerabilidades o fallas en el sistema para posibilitar la creación de un nivel de seguridad propio que permita la adopción de medidas de ciberseguridad.

1 PLANTEAMIENTO DEL PROBLEMA

De acuerdo con la RAE el malware es un programa introducido de manera oculta en la memoria de una computadora que, al activarse, afecta su funcionamiento destruyendo total o parcialmente la información almacenada o causando mal luso de los datos allí contenidos. Estos datos pueden ser cuentas bancarias, números de tarjetas de crédito, contraseñas y archivos personales o información confidencial de una organización.

El malware moderno es muy complejo y hay muchas variantes del mismo con diferentes habilidades que evolucionan cada día, una infección de malware puede afectar a cualquier tipo de ordenador ya sea de uso personal o empresarial que al aprovechar defectos de seguridad y vulnerabilidades del sistema como versiones desactualizadas del software operativo o de aplicaciones (navegadores, office, correo, antivirus), logran acceder al dispositivo de manera no autorizada y sin conocimiento alguno del propietario, se debe tener un control riguroso en la actualización periódica de estas aplicaciones y del software operativo para así mejorar el rendimiento y seguridad del equipo. El malware representa un gran problema para la tecnología porque la estrategia de infección varia para diferentes tipos de malware (puertas traseras, clave registradores, ladrones de contraseñas, entre otros) y gracias al avance de la tecnología en sistemas operativos , aplicaciones y entornos informáticos son aspectos que dificultan el control de este malware y aumentan las amenazas y vulnerabilidades a los sistemas informáticos, con el fin de poder realizar un esfuerzo para poder ejecutar técnicas que permitan la detección y prevención de nuevas amenazas informáticas, se vuelve un aspecto clave el poder analizar patrones , similitudes y vectores de ataque para lo cual es importante reconocer las familias existentes de malware y las vulnerabilidades que han explotado y esto se puede lograr analizando cuidadosamente el código de cada malware o el contenido de los documentos de texto no estructurados y así saber cuál vulnerabilidad se puede explotar o no y es en esta última manera de almacenar las vulnerabilidades y su malware donde nos centraremos y donde el concepto de minería de texto juega un papel importante.

La problemática anteriormente descrita motiva la pregunta de investigación que da origen a la presente propuesta: ¿De qué manera se puede utilizar técnicas de procesamiento de lenguaje natural para identificar las vulnerabilidades en herramientas ofimáticas de la suite office?

2 JUSTIFICACIÓN

La Transformación Digital supone un profundo cambio en la manera en la que trabajamos y nos aprovechamos de la tecnología. El avance tecnológico de la informática y la computación, incorporaron en las organizaciones un enfoque diferente al habitual para acceder al conocimiento, flexibilidad, interactividad, economía, rapidez, independencia, comunicación y desarrollo (Mujica, 2000).

Conforme al avance de la tecnología y para lograr ser competitivo dentro de este entorno es indispensable aprovechar nuevas alternativas, así mismo es de vital importancia innovar en una solución tecnológica propia utilizando tecnología a la vanguardia con el fin de automatizar controlar e integrar en línea aspectos de minería de datos en vulnerabilidades ofimáticas con el fin de obtener análisis mediante técnicas de machine learning.

Es imposible saber el alcance total de las vulnerabilidades de ciberseguridad en una aplicación o un sistema operativo ya que al avanzar el tiempo las características van evolucionando, haciendo procesos obsoletos, inseguros y fáciles de alterar con fines maliciosos, pero al examinar los datos disponibles de estas vulnerabilidades que se presentan, se determina si se pueden identificar tendencias o patrones útiles que pueden aparecer con el fin de mitigar impactos futuros, errores adversos y que en la mayoría de las defensas cibernéticas a menudo son el resultado de un error humano fácil de remediar. Los enfoques más vitales de la analítica de malware pueden ayudar en el campo de la seguridad a explorar más y a los ingenieros a elegir el análisis de malware como su campo de estudio aplicando el uso de machine learning y minería de datos para una mejor experiencia en cuanto al análisis exploratorio del malware.

3 OBJETIVOS

3.1 OBJETIVO GENERAL

Diseñar una herramienta para identificar vulnerabilidades en aplicaciones ofimáticas de la suite office de Windows implementando técnicas de procesamiento de lenguaje natural en Python.

3.2 OBJETIVOS ESPECÍFICOS

- Identificar vulnerabilidades en aplicaciones ofimáticas de la suite de office mediante búsqueda sistemática de información y revisión en bases de datos de vulnerabilidades.

- Construir el data set con las vulnerabilidades ofimáticas identificadas que soporten la implementación del procesamiento de lenguaje natural mediante técnicas de extracción, transformación y carga de los datos (ETL) usando web scraping.

- Implementar técnicas de procesamiento de lenguaje natural para el reconocimiento de tendencias útiles en el análisis de las vulnerabilidades de la data set generado.

- Desarrollar un módulo de interfaz gráfica web usando el entorno de desarrollo de Django de Python que soporte el procesamiento de lenguaje natural para identificar vulnerabilidades.

4 ESTADO DEL ARTE

En este capítulo se presenta una revisión de literatura relacionada con el proyecto, el cual se divide en dos partes, una con la revisión sistemática de literatura y otra, con los resultados de investigación más relevantes de terceros sobre sistemas de machine learning y minería de datos aplicado para análisis de malware en aplicaciones y que se han desarrollado tanto a nivel mundial como nacional.

Siendo un objetivo específico el estado del arte, se irá actualizando y mejorando en el transcurso del proyecto.

4.1 REVISIÓN SISTEMÁTICA DE LITERATURA

Se empezó buscando con palabras claves, intentado con “data mining to natural processing lenguaje ,malware and machine learning ”, con principios de inclusión entre los años 2015 - 2020 el cual no arrojó una búsqueda exacta exitosa de los temas pertinentes para trabajar en el proyecto pero si temáticas para poder destacar , después se intentó unificando palabras y realizar la búsqueda con la palabra clave: “data mining to malware , “text analysis and machine learning ” delimitando los principios de inclusión entre los años 2010 - 2020 en donde se escogió los más relevantes, se rechazaron algunos documentos por la fecha, documentos que no eran acordes, entre otras, se leyeron algunos por las condiciones relacionado al tema, la última búsqueda realizada fue el 28 de Abril del 2020.

4.2 RESULTADOS DE INVESTIGACIÓN

Automated microsoft office macro malware detection using machine learning

Este articulo intensifica la necesidad de automatizar la detección de macro malware ya que os atacantes implementando técnicas de extracción de datos para hacer parecer el malware de office como legítimo. Se propone una clasificación de archivos de microsoft office para detectar macros malignas o benignas a través del modelo Knn, donde los n-gramas de ejecución de código son los que están en el archivo.

Classification of Malware Using Visualisation of Similarity Matrices

El principal objetivo de este articulo era el de detectar malwares desconocidos que dan vida a ofuscaciones de x86 IA-32 (códigos de operación) para evadir métodos de detección tradicionales. Mediante minería de datos se examinaban los códigos de los malware, obteniendo Matrices de similitud, medidas de similitud, poder verlos de forma gráfica en clústeres, similitudes de contenido representadas visualmente en un plano ya sea 2d o 3d y la distancia entre cada par de puntos muestra que tan similares. (Venkatraman & Alazab, 2017)

Aportes al proyecto: Bases de datos públicas de malware y vulnerabilidades.

Malware Analytics: Review of Data Mining, Machine Learning and Big Data Perspectives

Método basado en la analítica de malware que se ocupa de los enfoques y técnicas utilizadas para generar características distintivas del malware explorando diferentes tipos de malware para encontrar patrones ocultos o asociar características utilizando técnicas de ingeniería inversa como la depuración en diferentes niveles de ejecución de código. De la ingeniería inversa se incorporan a las técnicas de minería de datos y a los modelos de aprendizaje automático para capacitar y probar el modelo de detección y análisis diseñado, minando ejecutables del malware para análisis estático o dinámico y obtener las características sin procesar iniciales (la minería de datos implica obtener datos sin procesar como un conjunto de características iniciales de muestras de malware) de la minería de datos cognitiva. (Poudyal et al., 2019)

Aportes al proyecto: El aprendizaje automático se logra mediante varios algoritmos de aprendizaje automático donde la mayoría de ellos utilizan diferentes enfoques probabilísticos. Considerar el amplio valor de cómputo, principalmente fusionado para resolver una amplia gama de problemas, desde seguridad.

Evaluation of Malware Clustering Based on its Dynamic Behaviour

El principal objetivo es el de comparar el comportamiento de dos representaciones para malware ejecutables, un conjunto de doce distancias para compararlos, y tres variantes del aglomerativo jerárquico algoritmo de agrupamiento cuando se utiliza para capturar la estructura de diferentes familias y subfamilias de malware.

Es un método basado en agrupar automáticamente el malware basado en análisis dinámico de código y luego detectar nuevos ataques, este proceso se divide en tres pasos principales: los datos deben recopilarse, representarse y almacenarse; las distancias para comparar los datos almacenados deben seleccionarse y, finalmente, debe elegirse un algoritmo de agrupamiento para ser utilizado con la representación de datos seleccionada y la distancia. (Gurrutxaga et al., 2008)

Aportes al proyecto: El análisis de costos computacionales puede ayudar a seleccionar el algoritmo más conveniente.

El análisis de virus de los antivirus se basa en análisis estático, es decir, representan el malware basado en la información estructural de un archivo. La ofuscación de código y datos también plantea considerables l desafíos para el análisis estático.

El malware tiene una estructura jerárquica (familias, subfamilias, variantes, tipo, etc). Teniendo en cuenta esto estructura, hemos seleccionado agrupamiento jerárquico algoritmos para la experimentación.

Malware Detection by Text and Data Mining

El principal objetivo es el de aplicar un método de análisis estático para detectar Malware basado en secuencias de llamadas API utilizando minería de texto y datos en tándem.

Es un método basado en calcular los valores TF-IDF con la ayuda de una herramienta de código abierto, que admite minería de texto. Para así poder obtener tokens únicos o variables para las cuales se obtienen valores tf-idf. Para estas variables, se obtiene la información para la selección de características (MIFS) Se utiliza para extraer las características más influyentes. Luego se construye una matriz de documento-término para alimentarla a las técnicas de minería de datos DT, SVM, PNN, MLP y GMDH.

Aportes al proyecto: El algoritmo de minería de texto funciona en extraer el patrón oculto en datos textuales y se realiza mediante preprocesamiento que incluye tokenización, eliminación de stop palabras y derivaciones-

La frecuencia de documento de frecuencia inversa (TF-IDF) es uno de los nuevos métodos de extracción de características en Text Mining. Se calcula mediante una fórmula la cual involucra el número de apariciones de un término en el texto y el número de ocurrencias de los términos en el texto.

Modeling Malware as a Language

El principal objetivo es hacia un enfoque para el análisis de malware estático, con el objetivo de tratar el análisis de malware como análisis de lenguaje natural. Se propone modelar malware como lenguaje y evaluar la viabilidad de encontrar semántica en instancias de ese lenguaje.

La metodología consiste en aislar similitudes semánticas entre instancias de malware de la misma clase y clasificar las instancias desconocidas por su similitud con una clase. Se usa el modelo word2vec para generar una representación computacional de documentos y elegir una distancia de documento como medida de cercanía semántica entre ellos. Se clasifican los documentos de malware aplicando el algoritmo k vecinos más cercanos (kNN) y se validando el modelo utilizando la validación cruzada de dejar uno fuera, se obtiene una precisión de clasificación de hasta el 98%. (Awad et al., 2018)

Aportes al proyecto: Viabilidad de modelar malware como lenguaje natural y encontrar similitudes entre diversas clases de malware.

5 MARCO REFERENCIAL

5.1 MARCO CONCEPTUAL

En esta sección se presenta algunos de los conceptos más citados y reconocidos en este proyecto de investigación.

Bases de datos: Las bases de datos nacen de la necesidad de obtener persistencia de datos para la humanidad, las bases de datos permiten almacenar de forma fiable, sin contradicciones y a largo plazo los datos de todos los programas usados en un sistema organizacional; son gestionadas por sistemas de gestión de bases de datos que ponen a disposición del usuario la información requerida.

Bases de datos relacionales: Son un tipo de base de datos que almacena y proporciona acceso a puntos de datos relacionados entre sí basándose en el modelo relacional, una forma intuitiva y directa de representar datos en tablas, cada fila de la tabla es un registro con un ID único llamado clave. Las columnas de la tabla contienen atributos de los datos, y cada registro generalmente tiene un valor para cada atributo. (Oracle, 2017)

Bases de datos no relacionales: Surgen en el punto de quiebre cuando las grandes empresas tenían la necesidad de guardar y extraer grandes cantidades de datos en periodos muy cortos de tiempo, una exigencia mayor, llamada big data. Una nueva solución. Están las basadas en documentos y en grafos, cada una con aplicaciones diferentes. La base de datos no relacional basada en documentos es útil para mantener los datos “vivos” en una aplicación y la de grafos es útil para encontrar relaciones grandes y complejas entre los datos y es muy usada para aplicar en machine learning. Cuando las aplicaciones que usamos se masifican (aumentar enormemente el número de datos) , se rompe el paradigma de base de datos relacionales , dando un nuevo origen llamado big data, aunque las bases de datos relacionales podían hacer de todo pero en la cantidad de datos del orden de billones presentaban problemas de consistencia , join de un tabla con otra, hacer reportes , problemas procesando el uso particular de datos para una necesidad.

Ciencia de datos: La ciencia de datos consiste en la aplicación de técnicas de ETL, Data Mining y analítica de negocios, es un campo interdisciplinario que utiliza métodos, procesos, algoritmos y sistemas científicos para extraer valor de los datos. Los científicos de datos combinan una variedad de habilidades, entre ellas estadísticas, informática y conocimiento empresarial, para analizar datos recopilados de la web, de teléfonos inteligentes, de clientes, sensores y otras fuentes. (Oracle, Oracle.com/co/data-science, 2019)

Analítica de negocios : Técnica o disciplina en la cual trata de proveer a los interesado con los datos y la información correcta para que puedan tomar buenas decisiones .Se encarga de tomar los datos donde fueron cargados en el último proceso de ETL , para aplicar business intelligence en donde se da sentido a toda la información, analizar cualquier área deseada y poder dar valor agregado al negocio al obtener por ejemplo el histórico de ventas en que mes suben y bajan , qué tipo de usuarios , jóvenes , adultos , intereses de ellos , ver los patrones que tenemos en los datos ya guardados que se han ido repitiendo en el tiempo. Es una parte muy importante en la carrera de analista de datos, es el punto final de la cadena en la que se toman datos y se transforman. (Oracle, Oracle, 2018)

5.2 MARCO TEORICO

En este capítulo, se expondrán las teorías utilizadas en el marco de este proyecto, divididas en tres secciones: Minería de datos, malware e inteligencia artificial y los términos desglosados de cada una de estas secciones principales tales como etl, inteligencia de negocio, vulnerabilidades, suite ofimática, entre otros.

Minería de datos. La minería de datos es el proceso de encontrar características o patrones útiles ocultos del conjunto de datos o conjunto de muestras dado. Es una disciplina que se dedica a minar datos, es decir extraer los datos de donde sea que estén almacenados, ya sea en la base de datos que se tiene en producción o en un almacén de datos de registros históricos para poder tratarlos, darles un sentido y así volverlos a guardar, pero de una forma más organizada, más aprovechable para la lógica del negocio. (Microsoft, 2019)

ETL: Son las siglas de extraer, transformar y cargar (Extract, Transform, Load), es una técnica usada en minería de datos para tomar datos de las bases de datos de las aplicaciones o datos de bases de datos históricas, transformarlos y guardarlos en un almacén de datos para posteriormente ser

explotados en análisis de datos, analítica, inteligencia empresarial y machine learning. Una analogía muy usada es el ejemplo de convertir un pedazo de carbón que es extraído de una mina (materia prima) y aplicar procesos para transformarlo en un diamante. (Microsoft, 2019)

Es muy aplicado a la actualidad ya que muchas veces en las empresas tienen la información en forma no coherente, sin orden real y para dar algo de sentido en la información se usa ETL, se toman los datos de alguna base de datos y procesamos (proceso de transformación) la información para posteriormente cargarla en una base de datos o datawarehouse. Se describe de forma más detallada cada sigla:

- **Extracción:** Proceso de adquirir los datos de las bases de datos Oracle, SQL, MySQL (bases de datos relacionales) o alguna base de datos no relacional, también puede tomar o extraer información desde archivos en texto plano, csv, Excel o desde otro almacén de datos.

- **Transformación:** Es la técnica en la que aplicamos limpieza como quitar espacios, eliminar símbolos, pasar textos a minúscula y quitar espacios extras (unificación de datos) para que tenga los mismos campos, es un preprocesamiento de la información.

- **Carga:** Proceso que consiste en crear una base de datos con la información previamente procesada.

Etl abarca así dos posibles casos de uso el primero es la minería a bases de datos en tiempo real, se puede usar una base de datos de documentos para llevar el estado actual de una aplicación, un juego o un sistema, datos “vivos” en mi app, y el otro caso de uso es la minería de datos a bases de datos de registros históricos o datawarehouse.

Malware: Código malicioso, es un programa cuya función es dañar un sistema o causar mal funcionamiento. (Cisco, 2016) . Existen gran cantidad de variedad de malware, entre los que se destacan para este proyecto:

Virus: Código ejecutable malicioso que se adjunta a otros archivos ejecutables que son legítimos. Requieren activación del usuario final y puede activarse en fechas o momentos específicos, pueden ser inofensivos o destructivos. Se pueden programar para mutar, en usb, discos ópticos y recursos de red. Entran en la categoría de replicadores debido a sus

comportamientos de replicación. El virus se replica dentro de la máquina donde se ha infectado. (Cisco, 2019)

Spammers: Sobrecargan su casilla de correo electrónico o sus publicaciones en la web / blog con cantidades masivas de datos, lo que puede hacer que su sitio no responda o se dañe. Mientras que las bombas lógicas se ejecutan cuando se cumplen algunas condiciones específicas por parte del escritor de malware, mientras que el caballo de Troya puede ocultarse en su aplicación (por ejemplo, calculadora) y puede registrar todas sus acciones (por ejemplo, cálculos financieros).

Ransomware: Es una categoría especial de malware capaz de comunicación remota con un servidor de comando y control para fines de intercambio de claves. Consiste en mantener captivo un sistema de computación o los datos que contienen hasta recibir un pago, encripta estos datos con una clave. Aprovecha las vulnerabilidades del sistema o de aplicaciones para bloquearlo. (Cisco, 2019).

Puertas traseras y los bots: Permiten que el atacante se conecte a la computadora de la víctima con poca o ninguna autenticación y ejecute comandos en los sistemas informáticos locales.

Adware: Presenta información publicitaria no deseada y también puede robar información de los usuarios. Incluyen rastreadores de actividades, recopilación de pulsación de teclas y captura de datos, permiten rastrear y espiar.

Rootkit: Son códigos maliciosos diseñados para ocultar la existencia de otras variantes de malware. Por ejemplo, pueden ocultar la existencia de Adwares.

Ofuscación de código: Técnica para transformar el código del programa original cambiando su flujo o estructura para dificultar la depuración o la ingeniería inversa.

La información que se analiza en este proyecto son las vulnerabilidades en las herramientas ofimáticas.

Vulnerabilidades: Una vulnerabilidad es un defecto o un fallo ocurrido por programación, diseño o configuración. Dicho fallo puede alterar el flujo normal del sistema y una persona malintencionada puede realizar algo malicioso. (Cisco, 2017). Algunas de los defectos en configuración pueden estar dados por los siguientes factores:

- **Debilidades en las prácticas de seguridad**: Dado que los sistemas y los datos son confidenciales, la falta de protección en cuanto a técnicas de autenticación, autorización y encriptación.

- **Entrada no validada**: Los programas suelen trabajar con entradas de datos, estos datos pueden contener software malicioso para que se comporten de manera no deseada. Dimensiones creadas maliciosamente podría forzar a crear búferes de tamaños improvistos. (Urbina, 2017)

- **Desbordamiento de búfer**: Cuando los datos se escriben más allá de los límites de un búfer, la aplicación accede a la memoria asignada a otros procesos, causando bloqueo del sistema y comprometiendo datos o escalar privilegios. (Ciberseguridad, 2017).

- **Condiciones de carrera**: Sucede cuando el resultado de un evento depende de resultados ordenados o temporizados y se producen en el orden o tiempos incorrectos.

El aprovechamiento de vulnerabilidades conlleva a recopilar información sobre el sistema de destino como sistema operativo, versión, listas de servicios y así poder explotar un posterior ataque.

Amenaza: Una posibilidad de violación de la seguridad que existe cuando se da una circunstancia, capacidad, acción o evento que pueda romper la seguridad y causar perjuicio. Es decir, una amenaza es un peligro posible que puede explotar una vulnerabilidad. (RFC 2828).

Existen diferentes categorías de amenazas:

- **Red**: Una red es un conjunto de dispositivos conectados por un canal de comunicación con el propósito de compartir información. Mediante este, la información viaja de un lado a otro, por lo cual un agente malicioso puede irrumpir en el mismo y robar datos sensibles.

- **Host**: Se dirige a un sistema particular en el que reside información valiosa en el cual los atacantes intentan violar la seguridad.

- **Aplicación**: Una aplicación puede ser vulnerable si no se toman las medidas de seguridad adecuadas durante el desarrollo, implementación y mantenimiento.

Ataque: Cualquier acción que comprometa la seguridad de la información de una organización. Un asalto a la seguridad del sistema, derivado de una amenaza inteligente; es decir, un acto inteligente y deliberado (especialmente en el sentido método o técnica) para eludir los servicios de seguridad y violar la política de seguridad de un sistema. (RFC 2828).

Ataque a las aplicaciones: Existen miles de piezas de software y programas de todo tipo y tamaño disponibles en el mundo que sufren una amplia gama de ataques, tales como:

- **Denegación de servicio:** Son un tipo de ataque que da como resultado cierto tipo de interrupción del servicio de la aplicación o de red a los usuarios. Existen dos variantes de este ataque que se pueden materializar en: Cantidad abrumadora de tráfico y paquetes maliciosos formateados.

- **Inyección SQL:** Es un tipo de ataque que me permite alterar el comportamiento de una sentencia que es ingresada en un espacio para que el usuario ingrese su información, enviando esa inyección al servidor de bases de datos y obteniendo información importante.

- **Directorio transversal:** El objetivo es el de conseguir acceso a ficheros o directorios que se encuentran fuera del directorio web raíz y en los que en condiciones normales un usuario sin privilegios no tendría acceso alguno

- **Cross site scripting:** Es un ataque que permite ingresar comandos con el fin de alterar el funcionamiento de la página estos comandos pueden ser de tipo JavaScript, o también cargando un script a la aplicación.

Inteligencia Artificial: Provee a un algoritmo la capacidad de aprender a través del aprendizaje supervisado y no supervisado que se le aplica a un modelo con el fin de entrenarlo y probarlo con una serie de datos que le son suministrados previamente. (Minddata, 2018).

Machine learning: Es una rama de la inteligencia artificial que consiste en una serie de técnicas que permiten a un algoritmo proveer la capacidad de aprendizaje y la detección de patrones en diferentes ámbitos, una de las principales aplicaciones es el tratamiento de datos, un paso más adelante de inteligencia empresarial (ayuda a la gente a tomar mejores decisiones). (Oracle, 2017).

Los tipos de aprendizaje que se pueden aplicar en machine learning para el tratamiento de datos puede ser:

- **Aprendizaje supervisado:** Hace uso de una información de entrada y de sus etiquetas correspondientes para cada conjunto de información.

- **Aprendizaje no supervisado:** Solo tiene acceso a información, no se asocia una etiqueta. el algoritmo aprende haciendo asociación de los datos de entrada, se requiere una exploración de estos datos para intentar descubrir patrones o relaciones, la información importante está intrínsica en la propia naturaleza de los datos.

- **Aprendizaje profundo:** Se basa en redes neuronales. Ejecutan datos a través de varias capas de algoritmos de redes neuronales, las cuales pasan a una representación simplificada de los datos a la siguiente capa. (Bernard).

El machine learning abarca dos tipos de problemas:

- **Problema de clasificación:** Está contenido en el aprendizaje supervisado, no necesita un ser humano que le vaya entrenando lo que le tiene que decidir, automáticamente la máquina sabe predecir y separar por donde va ya que nuestro modelo observa los datos históricos y a partir de esta información se puede hacer una clasificación o una regresión.

La clasificación sirve para predecir etiquetas de clase categóricas en un grupo de instancias, una nube de puntos o cualquier conjunto de datos que se le presente a este algoritmo, el modelo a partir de decisiones pasadas sabrá qué decisión es la mejor. Por ejemplo, tenemos en una página de blogs el historial de los blogs que han creado los usuarios y se desean saber cuáles temas hay, que temáticas, pero no buscando palabras claves o con filtros directos en la página de blogs, sino más bien usando el procesamiento de lenguaje natural, tomando una serie de datos (los blogs escritos) y se implementan procesos para obtener patrones útiles que se puedan encontrar. Algunos de los algoritmos de clasificación son:

1. **Regresión logística:** A diferencia de la regresión lineal, la regresión logística no unifica los puntos para intentar unificar el comportamiento de los números, lo que hace es separarlos en nubes de puntos dispersas y cuantizados, para ellos se apoya de la función Sigmoide, convirtiendo cualquier entrada de datos en ceros o unos, cuantizados y separando así entre dos clases.

2. **Support Vector Machines:** Consiste en separar los datos que se tienen de entrada, siendo el objetivo principal el desarrollo de una serie de métodos para poder separar los tipos de datos sin ningún problema en las clases. Primero mapea la información que tiene para dividirla en zonas, zonas que estarán delimitadas por vectores de soporte que son un límite entre las áreas: hiperplano positivo, hiperplano negativo y un margen que es una línea o frontera de decisión, donde se asigna a ambos lados clases diferentes.

3. **Árboles y bosques.** Los árboles son un caso particular de grafos, el algoritmo de acuerdo con una serie de preguntas, decisiones y descartes dentro de la estructura de árbol de decisión separa las clases de las flechas. Los bosques conjugan un conjunto de diferentes árboles con el fin de seguir un hilo de preguntas y respuestas para llegar a una clase final que es la decisión deseada.

4. **K – vecinos más próximos:** Algoritmo que clasifica cada dato nuevo en el grupo que corresponda, según tenga k vecinos más cerca de un grupo o de otro. Es decir, calcula la distancia del elemento nuevo a cada uno de los existentes, y ordena dichas distancias de menor a mayor para ir seleccionando el grupo al que pertenecer. Este grupo será, por tanto, el de mayor frecuencia con menores distancias.

- **Problemas de predicción:** Nos ayuda a la toma de decisiones con base a una información que tiene un comportamiento histórico. Los algoritmos usados para predicción trabajan solo con tipos de datos cuantitativos y estos son:

Regresión lineal: Permite conseguir tendencia en los datos, este es un algoritmo de tipo supervisado ya que debemos usar datos previamente etiquetados. Generando a partir de los datos, una recta y es a partir de esta que podremos encontrar la tendencia o predicción.

1. **Regresión lineal múltiple:** Cuando nuestro modelo considera más de dos variables y este trabaja sobre un sistema de referencia conocido como hiperplano.

Para el correcto entrenamiento del modelo de machine learning se deben tener en cuenta el nivel de entendimiento que este tendrá, para ello existen dos niveles :

Sobreajuste: Cuando el modelo lo obligamos a entender un caso en particular o overfiting, es decir cuando al modelo le otorgamos una cantidad de datos muy grande, pero del mismo tipo, limitándolo a un solo de entendimiento.

Subajuste: El modelo fallará en el reconocimiento por falta de muestras suficientes. Conjunto de datos muy pequeño es decir que la cantidad de datos con la que se alimenta el modelo no es óptima para entender.

Procesamiento del Lenguaje Natural: Es el campo de estudio que se enfoca en la comprensión mediante ordenador del lenguaje humano, buscando que las computadoras entiendan el lenguaje humano, no solo analizar las palabras, también la intención, el sentido, para analizar estos mensajes escritos u hablados el procesamiento del lenguaje natural integra machine learning, lingüística y ciencias de la computación.

Los usos actuales del procesamiento del lenguaje natural son:

- Máquinas de búsqueda
- Traducción de texto
- Chatbots
- Análisis de discurso
- Reconocimiento del habla
- Análisis de sentimientos
- Clasificación automática de textos
- Identificación de temas
- Identificar intención expresada por el autor del texto.

El procesamiento de lenguaje natural usa técnicas de normalización para la limpieza del texto a analizar:

- **Tokenización:** Es el proceso mediante el cual se subdivide una cadena de texto en unidades lingüísticas mínimas (palabras).

- **Lematización:** Proceso para encontrar la raíz lingüística de una palabra.

- **Segmentación:** Separación del texto en frases dependiendo de un carácter (comas).

Corpus: Colección de muchos textos

Corpora: Colección de colecciones de textos.

Lingüística computacional: Estudia el lenguaje desde una perspectiva más científica.

Medida de riqueza léxica en un texto:

Estadística del lenguaje: Los cálculos estadísticos más simples que se pueden efectuar sobre un texto o un corpus son los relacionados con frecuencias de aparición de palabras.

Web scraping: Es una de las técnicas usadas en procesamiento de lenguaje natural que permite extraer de una página web (sin ninguna restricción) el contenido que esta pueda tener dentro sus etiquetas contenidos en la estructura html de un sitio web. (Cordero, 2008). Para una mejor comprensión de cómo se encuentra estructurada la información dentro de un sitio web, se describen los siguientes componentes, contenidos en etiquetas:

- **Html:** Es una estructura para poder escribir la información, todas las páginas de internet están contenidas dentro de esta estructura inicial, se anotan las partes de información para que el navegador pueda interpretar el tipo de información allí encontrada dentro de etiquetas que siguen una jerarquía en donde se pueden encontrar los datos.

- **Css:** Hojas de estilo de cascada, nos permiten dar colores, estilos al texto y diferentes elementos de presentación.

- **JavaScript:** Nos permite añadir interactividad y cómputo a nuestra página web.

- **Json:** Es un formato para transmitir datos entre servidores y clientes, es la forma estándar en la que las páginas web y las aplicaciones se comunican con los servidores backend.

Web crawling: Proceso de mapeo e indexación de páginas web para conocer su contenido. Conocer la estructura de la web. Principalmente es para

conocer la estructura de la web que contenido tienen e indexarla (usado por motores de búsqueda como Google y Bing).

Bolsa de palabras: Un modelo de bolsa de palabras, o BoW para abreviar, es una forma de extraer características del texto para usarlas en el modelado, como con los algoritmos de aprendizaje automático. Se llama " bolsa " de palabras, porque se descarta cualquier información sobre el orden o la estructura de las palabras en el documento. El modelo solo se ocupa de si las palabras conocidas aparecen en el documento, no en qué lugar del documento.

5.3 MARCO LEGAL

Para el desarrollo del marco legal se debe tener en cuenta las siguientes consideraciones:

- La técnica de web scraping no debe ser ilegal ni incurrir en contra de alguna reglamentación local del país.
- El sitio web en el que se desea obtener la información no se le debe quebrantar alguno de sus términos y condiciones para la extracción de datos.
- No se debe acceder a algún directorio que no esté autorizado para la obtención de la información en el sitio web.
- La información debe ser usada con fines lícitos, no se debe dar un uso ilegal a la información extraída.

5.3.1 Ley 1581 de 2012 Protección de Datos

En caso de utilizar técnicas de web scraping para la implementación de sistemas de cumplimiento, en su fase de consulta de información sobre listas sancionatorias, se estaría en un escenario complejo desde el punto de vista legal ya que, en lo que se refiere a bases de datos públicas con datos personales de contenido negativo, se estarían recolectando o “raspando” de manera masiva datos personales de especial tratamiento, para su posterior clasificación y tratamiento. La técnica de scrap no es utilizada para la obtención de un dato, sino de varios, su escala es masiva y en ese sentido, a la luz de la normatividad vigente sobre protección de datos personales –

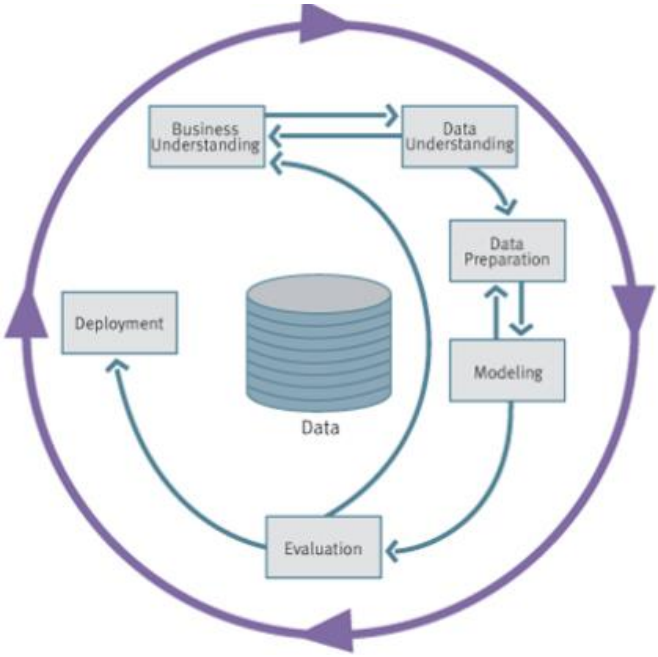
Ley 1581 de 2012-, sería violatoria del derecho de habeas data de los titulares de los datos que figuran en la base.

6 METODOLOGIA

Para el desarrollo de este proyecto, se utilizó el tipo de investigación aplicada, el cual consiste en obtener nuevos conocimientos y mejorar los existentes, con el fin de determinar un nuevo método o forma para solucionar un problema o necesidad identificada, (Colciencias, 2016).

Esta investigación aplicada tuvo un enfoque cuantitativo que, implica la recolección y análisis de datos (Sampieri, Collado, & Lucio, 2014).

Para el desarrollo del proyecto se implementa una adaptación de la metodología estándar para la minería de datos llamado CRISP-DM, es iterativo a nivel global y entre sus dependencias, puede ser enfocado en problemas de negocio real o para un análisis técnico, es muy flexible para el uso en cualquier aplicación o industria, no exige un uso especifico de herramientas y está especialmente enfocado a las fases más comunes utilizada en la minería de datos y son seis :



Fuente: https://s2.smu.edu/~mhd/8331f03/crisp.pdf

1. La primer fase es el entendimiento del negocio (Business underestanding), para este caso se abarca el entendimiento de la

tecnología y su funcionamiento a la cual se realiza el proceso de data mining para lo cual se llevó a cabo la elaboración del estado del arte sobre sistemas de procesamiento de lenguaje natural aplicado en vulnerabilidades ofimáticas (identificación y categorización), pasando por técnicas de extracción de datos y donde se debe definir el objetivo de la minería de datos.

2. La segunda fase es el entendimiento de los datos (Data underestanding) que consta de la recopilación, descripción, exploración y verificación de la calidad de los datos. Se realiza la adquisición de datos con sus características de diversas fuentes, para este caso particular de obtener los datos directamente desde páginas web se aplica una técnica de etl llamada web scraping, descrita a continuación:
    a. Se deben recopilar las paginas donde están almacenadas las bases de datos públicas de las vulnerabilidades escritas en texto plano, en lenguaje natural.
    b. Realizar una inspección a la estructura html de las paginas para obtener el tipo de etiquetas en donde está el contenido a extraer.
    c. Implementar el lenguaje de programación y librerías pertinentes que permita hacer peticiones o request a una página web con el fin de extraer la información.
3. La tercera fase consiste en la preparación de datos (data preparation). En la etapa de análisis de datos, se define un enfoque y derivado de él un objetivo exploratorio (EDA), si no se conocen los datos previamente y se quieren encontrar patrones o relaciones y se procede a limpiar la información para reestructurar estos datos y finalmente almacenarlos de forma local en un formato tipo csv o Excel.

4. La cuarta fase consiste en la implementación del modelado matemático y estadístico en donde se implementa la técnica de procesamiento de lenguaje natural a la información previamente generada y procesado con el fin de realizar diversos tipos de análisis. aplicado al procesamiento de lenguaje natural, para lo cual se deben implementar diversos de técnicas de las ramas de inteligencia artificial con el fin de adecuar y tratar el set de datos a cada modelo especifico.

5. La quinta fase es la evaluación, consiste en evaluar los gráficos y frecuencias que se han implementado y así compararlo con el objetivo inicial de la minería de datos y finalmente obteniendo la respuesta a la

aplicación del análisis de los datos. Para el éxito de la correcta implementación de la técnica se debe realizar una limpieza que son los caracteres que debemos darle, ya que es lo más difícil para la máquina que es lo que queremos que “entienda”.

6. La sexta fase es la de despliegue (deployment) donde se realiza el despliegue a producción de la herramienta, pero en este caso es un prototipo por lo cual correrá en un servidor local y no contará con ese proceso de automatización en la nube.

7 DESARROLLO Y RESULTADOS

A continuación, se presenta el desarrollo y los resultados obtenidos en cada fase de la metodología implementada, para el cumplimiento de cada uno de los objetivos propuestos.

7.1 PRIMERA FASE

Durante esta fase que es llamada entendimiento del negocio, para el desarrollo del proyecto se hace un contexto general de la seguridad en una adaptación que consiste en la identificación de las vulnerabilidades ofimáticas más comunes que usan los atacantes para aprovechar fallos de seguridad y así irrumpir en las aplicaciones ofimáticas. Cumpliendo así con el primer objetivo específico:

7.1.1 Vectores de ataque en Microsoft Office

En la actualidad son muchos los usuarios de las aplicaciones ofimáticas, las cuales se han hecho imprescindibles en el diario vivir de las personas, tanto en el ámbito, académico, empresarial o personal. Ello conlleva a que los cibercriminales están en constante búsqueda de nuevos vectores de ataque que les permitan explotar los posibles fallos en esta suite para aprovecharse y lucrarse por medio de ataques informáticos. A continuación, se describen algunos de los vectores de ataque más comunes.

Tabla 1. Vectores de ataque Microsoft Office

VECTOR DE ATAQUE	DESCRIPCIÓN
Habilitar macros	Los documentos ofimáticos de la suite de Microsoft que contienen macros (código VBA), cuya propiedad de lectura se aprueba por un usuario. Estas macros permiten la extracción de código malicioso cuando el usuario habilita este contenido dando permisos para la ejecución de scripts que lanzan el cmd o la powershell que permiten que un atacante realice cambios no deseados en el equipo de la víctima.

VECTOR DE ATAQUE	DESCRIPCIÓN
Uso de objetos incrustados o embebidos	Los documentos ofimáticos de la suite de Microsoft que contienen scripts maliciosos con vinculación e incrustación de objetos donde por ejemplo un atacante se vale del hecho de que Office permite cambiar el icono o imagen visual del contenido malicioso por uno de confianza, para así lograr que la víctima abra el contenido que llega a ser un Visual Basic Script ofuscado que ejecuta un script para incrustar y vincular otros archivos.
Uso de exploits	Los documentos ofimáticos de la suite de Microsoft que permiten explotar vulnerabilidades en las aplicaciones como: Word, Excel o Power Point. En este tipo de vector el script malicioso es ejecutado cuando el usuario abre el documento, sin la necesidad de aprobación previa y sin que se genere un mensaje de alerta o advertencia. El atacante ha ocultado un script de Visual Basic en el documento valiéndose de una vulnerabilidad cuando no se ha parcheado de manera correcta la aplicación.
Spearphishing	Los documentos ofimáticos de la suite de Microsoft son muy populares y permiten fácilmente disfrazar un software malicioso como un documento ofimático genuino para llevar a cabo ataques de phishing con archivos ofimáticos adjuntos que llegan al correo electrónico de los usuarios. El atacante engaña al usuario para que abra el adjunto ejecutando una macro maliciosa que descarga y ejecuta NetSupport Manager, la cual es una herramienta de acceso remoto legítima, pero que puede brindar acceso remoto no autorizado y ejecutar comandos de Shell.

Fuente: Adaptado de ESET

7.1.2 Vulnerabilidades en Microsoft Office

Como se mencionó anteriormente los exploits son un tipo especial de software malicioso, el cual aprovecha las vulnerabilidades de un sistema o aplicación para explotarlas en forma de ataques informáticos sin requerir de interacción con la víctima y transmitiéndose de forma discreta. Los ataques que se fundamentan en exploits son bastante potentes, ya que las vulnerabilidades dadas en la suite de Microsoft Office son bastante altas.

Para el 2013 el CSIRT-CV que es el centro de seguridad de las tecnologías de la información de la comunidad valenciana publicó dos vulnerabilidades CVE-2013-2189 y CVE-2013-4156 de office 365 que permitían realizar ataques de denegación de servicio a través de documentos de Word (csirtcv, 2013).

Igualmente, para 2014 el centro de servicios de información de noticias del estado de New York publico las vulnerabilidades CVE-2014-1757, CVE-2014-1758 y CVE-2014-1761de Microsoft Word, que permiten la ejecución de código remoto para ver, editar y eliminar datos (ITS, 2014). Para 2016 el cert-py que es el centro de respuestas ante incidentes cibernéticos de Paraguay publicó que algunas versiones de Word permitían ganar acceso y escalar privilegios en el sistema para infectar a los usuarios con malware. Estas vulnerabilidades siguen apareciendo en el último tiempo donde según Kaspersky Lab, para el periodo entre 2018 y 2019 se lograron identificar cerca de diez exploits de propagación libre en dichas plataformas. A través de los últimos años algunos ejemplos de las vulnerabilidades ofimáticas han sido altamente críticas son: la vulnerabilidad CVE-2017-11826, que fue una vulnerabilidad zero day que permitió ataques dirigidos hacia todas las versiones de Office, donde el exploit consistía en un documento tipo RTF dentro de un DOCX que explota la vulnerabilidad mediante el analizador sintáctico Office Open de XML, otra vulnerabilidad de este tipo fue la CVE-2018-8174 que permitía también ataques dirigidos. Por último, para 2019 según Incibe que es el instituto nacional de ciberseguridad de España las vulnerabilidades en aplicaciones ofimáticas que permiten la ejecución de código remoto estaban en aumento permitiendo ejecutar exploits y acceder a información confidencial de los usuarios (Incibe. 2019), A continuación, se describen algunas de las vulnerabilidades más comunes en Microsoft Office y que han sido mayormente explotadas en los últimos años según la Cisa que es la Agencia de Seguridad Cibernética e Infraestructura (CISA, 2020).

Tabla 2. Vulnerabilidades en Microsoft Office

VULNERABILIDAD	DESCRIPCIÓN
CVE-2017-11882	Esta vulnerabilidad de Microsoft Office permite la corrupción de la memoria permitiendo ejecutar código malicioso con el malware Loki, FormBook y Pony/FAREIT. Principalmente afecta Microsoft Office 2007, 2013 y 2016.
CVE-2017-0199	Esta vulnerabilidad de Microsoft Office/WordPad, permite ejecutar código malicioso con un archivo de Word modificado para infectar el sistema con el malware FINSPY, LATENTBOT, Dridex.
CVE-2019-0604	Esta vulnerabilidad de Microsoft SharePoint permite la ejecución de código cuando no se comprueba la fuente, se usa con el malware China Chopper.
CVE-2017-8759	Esta vulnerabilidad de .NET permite la ejecución de código en remoto, mediante la ejecución de una entrada que permite tomar el control del sistema después de que un usuario abre un documento de office.
CVE-2017-11882	Esta vulnerabilidad de Microsoft Office permite la ejecución de código en remoto, después de que un usuario abre un documento de office.
CVE-2015-2545	Esta vulnerabilidad de Microsoft Office permite ganar permisos y ejecutar código malicioso para recopilar información de un usuario.
Vulnerabilidad en Dynamic Data Exchange Protocol	Esta vulnerabilidad no ha sido considerada por Microsoft y permite usar la característica que brinda el protocolo de intercambio dinámico de datos en Microsoft Office, para ejecutar código malicioso sin tener que habilitar las macros u ocasionar una corrupción de memoria.
Vulnerabilidad de ejecución remota de código (RCE)	Esta vulnerabilidad está presente en la librería de Autodesk FBX que viene integrada en Microsoft Office 2016, 2019, y Office 365, donde un archivo FBX que comúnmente permite almacenar modelos 3D, activos, formas y animaciones, que además permite buffer e integer overflow, type confusion, use after free, NULL dereference para ejecutar código malicioso de desbordamiento de memoria y ataques DoS de forma remota.

Fuente: Adaptado de Kaspersky, NIST, MITRE

7.1.3 Caracterización de las Bases de datos de Vulnerabilidades

Este contexto de vulnerabilidades en Microsoft Office hace que sea de vital importancia identificar los sitios de bases de datos de vulnerabilidades como son: exploit-db, mitre o nist, para conocer de primera mano cuales vulnerabilidades están activas y estar constantemente informado sobre estas. A continuación, se hace una caracterización de estos sitos de consulta de bases de datos de vulnerabilidades.

Tabla 3. Caracterización Bases de datos de Vulnerabilidades

SITIO	DESCRIPCIÓN
Exploit-db[1]	Es una base de datos propiedad de Offensive Security donde se muestran múltiples exploits y vulnerabilidades compatibles con CVE, que además contiene una descripción detallada con la información de su explotación, En ella se puede encontrar además las herramientas y pasos para la realización de pruebas de concepto.
CVE mitre[2]	Es una base de datos donde se listan las vulnerabilidades de seguridad conocidas identificadas con un número, se hace una descripción y mínimo una referencia pública.
Nist[3]	Es una base de datos del gobierno de estados unidos para la gestión de vulnerabilidades que se basan en el Protocolo de automatización de contenido de seguridad para la automatización de la gestión de vulnerabilidades su medición incluyendo referencias de listas de verificación, fallas de software, y métricas de impacto.

Fuente: Adaptado de exploit-db, NIST, MITRE

7.2 SEGUNDA FASE

Durante esta fase que es llamada entendimiento de los datos, se realiza una técnica de web scraping para obtener las vulnerabilidades que pueden afectar a las aplicaciones ofimáticas con el fin de posteriormente hacer una limpieza y análisis de la información obtenida, estas vulnerabilidades se extraen de

[1] https://www.exploit-db.com/
[2] https://cve.mitre.org/
[3] https://nvd.nist.gov/

sitios oficiales de seguridad informática conocidos a nivel internacional por sus altos estándares de calidad y fiabilidad y que se usan como referencia para mantener un reporte y así alertar a la sociedad de que existen vulnerabilidades que afectan el software. El web scraping o rasgado de web por su traducción al español, es usado para obtener contenido de sitios web, buscándolo directamente en las diferentes etiquetas que tenga un documento html, para este caso en particular se descarga todo el documento html, ya que por políticas de seguridad y privacidad de la página web, no se permite extraer contenido de etiquetas específicas. Haciendo cumplimiento a las políticas de seguridad y derechos de autor se limita el número de vulnerabilidades que pueden ser extraídas y el tiempo que se puede implementar para cada extracción, esto viene limitado por el archivo de texto alojado en cualquier página web llamado robots.txt, que es un protocolo de exclusión de robots que impide la accesibilidad por medio de scripts a ciertas partes del sitio web.

7.3 TERCERA FASE

Durante esta fase que es llamada preparación de los datos hace referencia a la técnica que se implementa mediante un script de Python en un jupyter notebook, es una herramienta muy popular para extraer contenido de diversos sitios webs.

Cumpliendo así con el **objetivo específico número dos**, diseño y construcción del set de datos con vulnerabilidades, las cuales son extraídas de la página web y a las que se les realiza una posterior limpieza para eliminar las etiquetas html (con los símbolos asociados, signos de puntuación y las stop words), continuando así con un correcto tratamiento de los datos donde se pueden identificar y buscar las vulnerabilidades a través de la implementación de una tabla de datos(dataframe) con una librería de Python llamada pandas que permite una visualización en tabla y una búsqueda de frecuencias en una vulnerabilidad. Las filas de este dataframe contienen las palabras agrupadas por n-gramas de la información extraída. Cada tabla hace referencia a un n-grama especifico, ya sea bígama o trigrama con la especifica codificación en sus filas, en la tabla1 se puede observar la tabla de datos perteneciente a un bigrama, en la tabla2 se observa la tabla de datos perteneciente a un trigrama y en la tabla3 la tabla de datos generada para un 4-grama.

El tratamiento y limpieza de las vulnerabilidades extraídas de la página web con todo el documento html para la posterior generación de gráficas y de la data set, se hace de la siguiente manera:

Imagen 1. Funciones principales

```
aplicacion= str(input("Ingrese nombre de la aplicación para realizar el análisis ")) #1.
lista_vulnerabilidades = run_main(aplicacion)   #2.
to_chain= list_to_cadena(lista_vulnerabilidades) #3.
filtrado=cleaner(to_chain) # 4.
string_lower = filtrado.lower() #5.
stops=stop_to_words(string_lower) # 6.
frecuencias= count_words(stops) # 7.
```

Fuente: Elaboración propia

El primer paso hace referencia a la petición que se le hace al usuario para que ingrese la aplicación a la que le desea realizar el análisis vulnerabilidades.

El segundo paso hace referencia al llamado de la función principal que es donde se aloja la url de la página web de vulnerabilidades a la cual se le hace la extracción de la información y donde se modifica el nombre de la aplicación, en este caso se usa “word” para buscar para que realice el rasgado en la página web y poder tener las vulnerabilidades por aplicación especifica de forma local y así en los siguientes pasos proceder con la limpieza y posterior tratamiento de los datos.

Imagen 2. Función scraping

```
def run_main(aplicacion):
    print('Aplicación seleccionada es :{app}'.format(app=aplicacion))

    url = 'https://cve.mitre.org/cgi-bin/cvekey.cgi?keyword={app}'.format(app=aplicacion)
    response=requests.get(url)    #Objeto request
    soup = BeautifulSoup(response.content, 'html.parser')
    print(type(soup)) #<class 'bs4.BeautifulSoup'>
    secciones=soup.find(id="TableWithRules").find_all("td")
    print(type(secciones))  #<class 'bs4.element.ResultSet'>
    vulnerabilidades = []
    for tag in secciones:
        if len(vulnerabilidades)<=100:
            vulnerabilidades.append(tag)
        elif len(vulnerabilidades)>6:
            break
    return vulnerabilidades
```

Fuente: Elaboración propia

Esta función retorna una lista que se guarda en la variable llamada *lista_vulnerabilidades(imagen1)* y que contiene el documento html con todas las vulnerabilidades obtenidas del sitio web.

Se da cumplimiento con las políticas de seguridad de la página, solo se permiten realizar la extracción de 100 vulnerabilidades.

El tercer paso se realiza una conversión de la variable lista_vulnerabilidades de lista a cadena de todo el documento html extraído, guardándose en la variable llamada *to_chain(imagen1)* y que es necesario para la posterior limpieza.

En el cuarto paso se hace un llamado a la función cleaner la cual contiene la limpieza de la información mediante expresiones regulares para así eliminar los símbolos del documento html y los signos de admiración para finalmente guardar en la variable llamada **filtrado**(imagen1) una cadena de texto que contiene la estructura del documento html, pero sin los símbolos (aún se conservan etiquetas).

Imagen 3. Función expresiones regulares

```
def cleaner(paragraphs):
    Filtraddo=[re.split('[ \W\t\n]+', s) for s in paragraphs]
    flat = [x for l in Filtraddo for x in l ]
    string = " ".join(flat)
    sinruido = re.split(r'[ \W\t\n]+',string)
    String_dos = " ".join(sinruido)
    return String_dos
```

Fuente: Elaboración propia

En el quinto paso se estandariza toda la variable de tipo cadena llamada **filtrado** (anteriormente obtenida) a minúsculas, guardándose este nuevo cambio en la variable llamada string_lower(imagen1).

En el sexto paso se eliminan los stop words.

Imagen 4. Función stop words

```
def stop_to_words(texto):
    stopwords.words('english')
    token_with=main(texto)
    clean_toke = token_with[:]
    for token in token_with:
        if token in stopwords.words('english'):
            clean_toke.remove(token)
    return clean_toke
```

Fuente: Elaboración propia

En el séptimo paso se hace llamado a la función count_word que contiene el conteo de palabras únicas y la frecuencia con la que aparecen en la cadena llamada stops.

Imagen 5. Función frecuencia de palabras

```
def count_words(stop):
    dic = {}
    for palabra in set(stop):
        dic[palabra] = stop.count(palabra)
    long_words = [palabra for palabra in stops if len(palabra)>=3]
    vocabulario_filtrado = sorted(set(long_words))
    vocabulario_filtrado[:]
    fdist = FreqDist(stops)
    return fdist.most_common(100)
```

Fuente: Elaboración propia

En la siguiente imagen se observan las primeras 15 palabras más frecuentes, no se discrimina las palabras que son etiquetas del documento html.

7.4 CUARTA FASE

Imagen 6. Frecuencias de palabras

```
[('td', 202),
 ('cve', 164),
 ('microsoft', 112),
 ('nowrap', 102),
 ('cgi', 102),
 ('valign', 101),
 ('top', 101),
 ('vulnerability', 93),
 ('word', 90),
 ('2020', 70),
 ('code', 56),
 ('execution', 56),
 ('remote', 54),
 ('name', 53),
 ('href', 51),
```

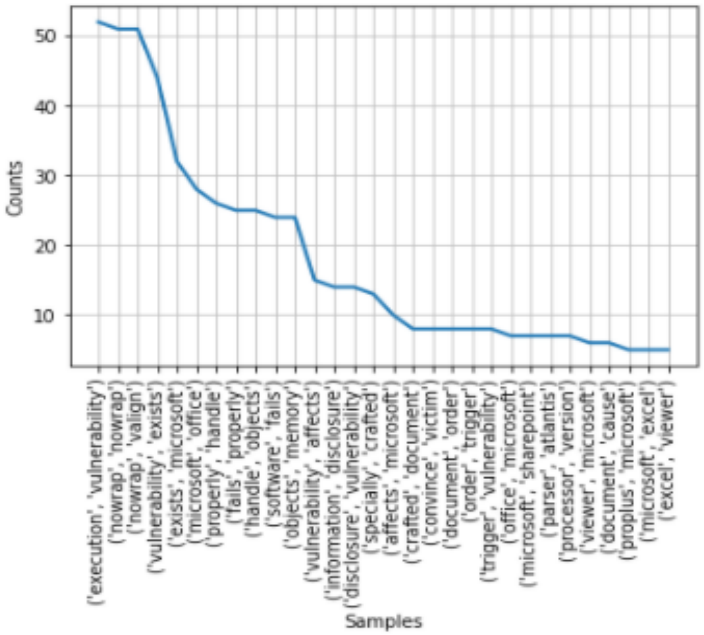
Fuente: Elaboración propia

El tratamiento anteriormente realizado se emplea para realizar el posterior agrupamiento de palabras por n-gramas, donde se cuenta la frecuencia de cada n-grama y se puede observar una gráfica que muestre la distribución de cada n-grama.

Las gráficas se generaron para cierta cantidad de n-gramas (4), empezando en orden ascendente y configurando la longitud de la palabra que se desea graficar eliminado así palabras propias del contenido html y que por el método de stop word no se pudieron eliminar.

En la gráfica 1 se puede observar la frecuencia de los primeros 30 bigramas.

Grafica1. Frecuencia de bigrama



Fuente: Elaboración propia

En la imagen 7 se observan los 15 primeros bigramas más frecuentes

Imagen 7. Frecuencia de bigramas

```
[(('valign', 'top'), 101),
 (('td', 'td'), 100),
 (('cve', '2020'), 70),
 (('microsoft', 'word'), 63),
 (('code', 'execution'), 56),
 (('remote', 'code'), 52),
 (('execution', 'vulnerability'), 52),
 (('td', 'nowrap'), 51),
 (('nowrap', 'nowrap'), 51),
 (('nowrap', 'valign'), 51),
 (('top', 'href'), 51),
 (('href', 'cgi'), 51),
 (('cgi', 'bin'), 51),
 (('bin', 'cvename'), 51),
 (('cvename', 'cgi'), 51),
```

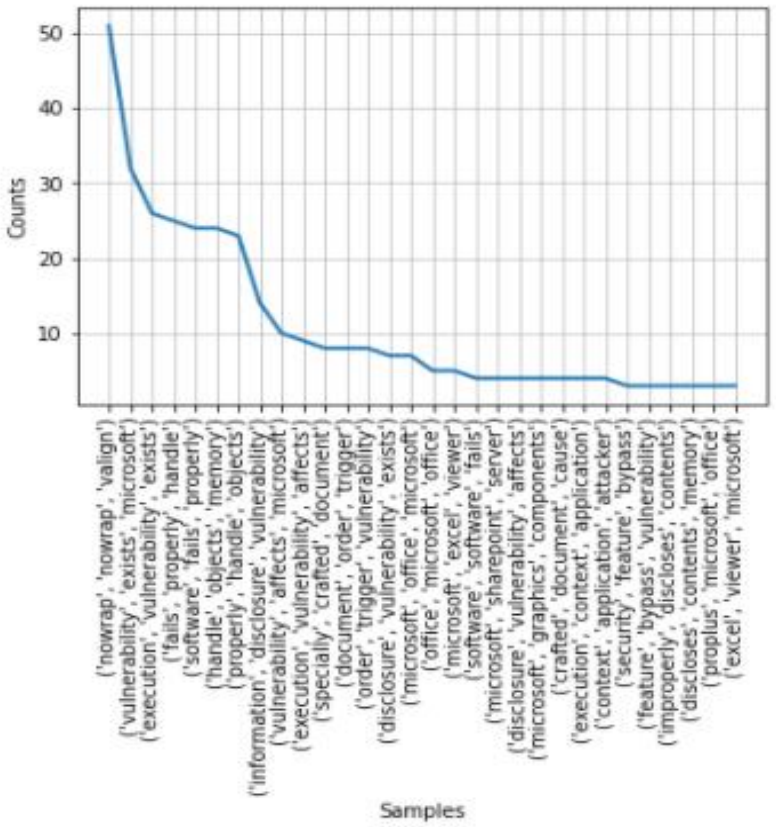
Fuente: Elaboración propia

Con la grafica1 se puede identificar que el bigrama que aparece con mayor frecuencia es el de “execution vulnerability” con mas de 50 apariciones lo cual indica que es una ejecución de código remota. En la imagen 7 se puede observar que gran parte de los bigramas encontrados hacen referencia al

contenido del documento html, a excepción de la aparición de las vulnerabilidades que presenta Microsoft word que son la ejecución de código y el código remoto, la cual indica que a estas dos se le puede ejecutar una vulnerabilidad. En el posterior análisis de ngramas se identifica si se refiere a una vulnerabilidad mas especifica.

Para el análisis de un trigrama se realiza con la grafica2 donde se filtra la longitud de las palabras (mayor o igual a 5) y se muestran los primeros 30 trigramas

Grafica 2. Frecuencia de trigrama



Fuente: Elaboración propia

En la imagen 8 se observan los 15 primeros trigramas más frecuentes

Imagen 8. Frecuencia de trigramas

```
[(('remote', 'code', 'execution'), 52),
 (('code', 'execution', 'vulnerability'), 52),
 (('td', 'nowrap', 'nowrap'), 51),
 (('nowrap', 'nowrap', 'valign'), 51),
 (('nowrap', 'valign', 'top'), 51),
 (('valign', 'top', 'href'), 51),
 (('top', 'href', 'cgi'), 51),
 (('href', 'cgi', 'bin'), 51),
 (('cgi', 'bin', 'cvename'), 51),
 (('bin', 'cvename', 'cgi'), 51),
 (('cvename', 'cgi', 'name'), 51),
 (('cgi', 'name', 'cve'), 51),
 (('td', 'td', 'valign'), 50),
 (('td', 'valign', 'top'), 50),
 (('td', 'td', 'nowrap'), 50),
 (('vulnerability', 'exists', 'microsoft'), 32),
```

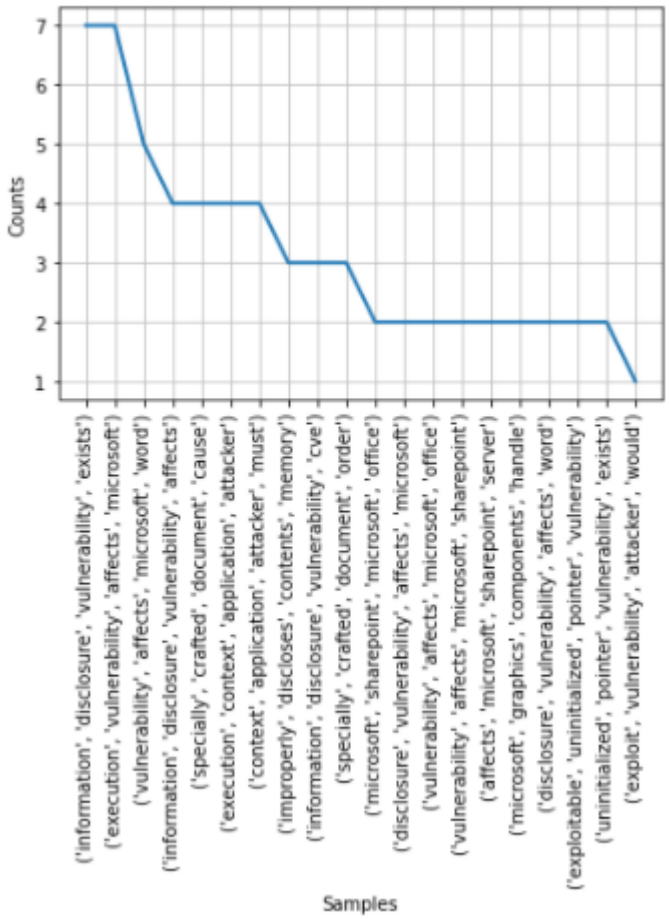
Fuente: Elaboración propia

El análisis obtenido a través de los trigramas en la gráfica es que se puede observar que la mayor frecuencia de aparición es para las etiquetas html que hacen referencia a la característica de una celda en una tabla , se puede observar que el segundo con mas frecuencia es una vulnerabilidad que existe en Microsoft , seguido de una ejecución de vulnerabilidad pero sin saber específicamente a cual se hace referencia, sin embargo con el cuarto y quinto trigrama se puede considerar que se trata de una vulnerabilidad que se presenta en memoria al almacenar los allí los objetos. En la imagen de frecuencias se puede observar que la mayor ocurrencia es para una vulnerabilidad que se llama remote code execution.

Y finalmente el análisis obtenido para un 4-grama es el siguiente:

En la gráfica 3 se puede observar la frecuencia de los primeros 30 trigramas.

Grafica 3. Frecuencia de 4-gramas



Fuente: Elaboración propia

Grafica 3. Frecuencia de 4-gramas

```
[(('remote', 'code', 'execution', 'vulnerability'), 52),
 (('td', 'nowrap', 'nowrap', 'valign'), 51),
 (('nowrap', 'nowrap', 'valign', 'top'), 51),
 (('nowrap', 'valign', 'top', 'href'), 51),
 (('valign', 'top', 'href', 'cgi'), 51),
 (('top', 'href', 'cgi', 'bin'), 51),
 (('href', 'cgi', 'bin', 'cvename'), 51),
 (('cgi', 'bin', 'cvename', 'cgi'), 51),
 (('bin', 'cvename', 'cgi', 'name'), 51),
 (('cvename', 'cgi', 'name', 'cve'), 51),
 (('td', 'td', 'valign', 'top'), 50),
 (('td', 'td', 'nowrap', 'nowrap'), 50),
 (('vulnerability', 'exists', 'microsoft', 'word'), 28),
 (('td', 'valign', 'top', 'remote'), 26),
 (('valign', 'top', 'remote', 'code'), 26),
 (('top', 'remote', 'code', 'execution'), 26),
 (('code', 'execution', 'vulnerability', 'exists'), 26),
 (('software', 'fails', 'properly', 'handle'), 24),
```

Fuente: Elaboración propia

Y el análisis obtenido del ultimo n-grama se observa que ya son frases mas específicas que son de mayor utilidad para identificar una vulnerabilidad específica, donde se puede identificar con una aparición de 52 veces “remote code execution vulnerabilidad” que hace referencia a la vulnerabilidad de ejecución de código remoto.

En esta misma fase se da cumplimiento al **objetivo específico 3** que consiste en el reconocimiento de tendencias útiles y estadísticas que son obtenidas desde los data set. En la imagen8, imagen9 e imagen10 se puede observar la visualización de estadísticas y frecuencias para cada tabla de datos generada respectivamente (bigrama, trigrama, 4grama).

Los data set se generaron a partir de las colocaciones(collocations) de las palabras que son la combinación de palabras con el fin de obtener un significado completamente nuevo y declarada por la forma en la que se ubica esa palabra en la secuencia de palabras. Para cada n-grama se obtienen una serie de estadísticas para comprender cuales son los más frecuentes y su comportamiento.

Para la siguiente elaboración del data set de bigramas se hizo un filtrado de la longitud de las palabras para que no aparecieran palabras con longitud menor 4, eliminando así la aparición de palabras que son etiquetas propias del contenido html. Se muestran los 15 bigramas con mayor frecuencia de aparición.

Imagen 8. Data set bigramas

	bi_gram	word_0	word_1	bi_gram_freq	word_0_freq	word_1_freq	PMI	log(bi_gram_freq)
27	(execution, vulnerability)	execution	vulnerability	52	0	0	inf	5.700440
28	(vulnerability, affects)	vulnerability	affects	15	0	0	inf	3.906891
0	(disclosure, vulnerability)	disclosure	vulnerability	14	0	0	inf	3.807355
59	(information, disclosure)	information	disclosure	14	2	0	inf	3.807355
26	(specially, crafted)	specially	crafted	13	8	0	inf	3.700440
57	(affects, microsoft)	affects	microsoft	10	0	0	inf	3.321928
24	(trigger, vulnerability)	trigger	vulnerability	8	0	0	inf	3.000000
69	(crafted, document)	crafted	document	8	0	0	inf	3.000000
103	(processor, version)	processor	version	7	0	0	inf	2.807355
15	(microsoft, sharepoint)	microsoft	sharepoint	7	0	0	inf	2.807355
29	(proplus, microsoft)	proplus	microsoft	5	0	0	inf	2.321928
8	(application, attacker)	application	attacker	4	5	0	inf	2.000000
2	(execution, context)	execution	context	4	0	2	inf	2.000000
45	(graphics, components)	graphics	components	4	0	0	inf	2.000000
98	(microsoft, graphics)	microsoft	graphics	4	0	0	inf	2.000000

Fuente: Elaboración propia

La columna PMI hace referencia a una métrica basada en teoría de la información para encontrar Collocations, a través de la siguiente formula:

$$PMI = \log\left(\frac{P(w_1, w_2)}{P(w_1)P(w_2)}\right)$$

Con el data set generado se pueden realizar una búsqueda de una vulnerabilidad especifica que el usuario desee y ya sea que aparezca en la primera o en la segunda palabra, esta concatenación se hace realizando dos búsquedas por aparte y uniendo estas búsquedas mediante un merge. En la imagen 9 se describe el proceso de buscar una palabra especifica que el usuario puede ingresar y observar las colocaciones que tiene en el data set y saber si una vulnerabilidad es identificada y la frecuencia con la que aparece.

Imagen 9. Búsqueda merge

```
word_to_search="remote"
searched=dataset_b[dataset_b["word_0"]==word_to_search]
searched_one=dataset_b[dataset_b["word_1"]==word_to_search]
join_one=pd.merge(searched, searched_one, how='outer')
join_one
```

	bi_gram	word_0	word_1	bi_gram_freq	word_0_freq	word_1_freq	PMI	log(bi_gram_freq)
0	(remote, attackers)	remote	attackers	1	10	0	inf	0.0
1	(allows, remote)	allows	remote	2	0	10	inf	1.0
2	(components, remote)	components	remote	1	0	10	inf	0.0
3	(engine, remote)	engine	remote	1	1	10	-3.321928	0.0
4	(office, remote)	office	remote	2	8	10	-5.321928	1.0
5	(android, remote)	android	remote	1	0	10	inf	0.0

Fuente: Elaboración propia

La búsqueda que la palabra remote que hace referencia a una vulnerabilidad, aparece en 5 diferentes bigramas, colada en diferentes partes , se puede identificar que la frecuencia del bigrama que mas apariciones tiene es la de “allows ,remote” y “office remote” lo cual nos indica que en microsoft office si existe esta vulnerabilidad y que puede ser explotada por medio de un exploit.

Imagen 10. Data set trigrama

	Tri_gram	word_0	word_1	word_2	Tri_gram_freq	word_0_freq	word_1_freq	word_2_freq
31	(nowrap, nowrap, valign)	nowrap	nowrap	valign	51	0	0	0
122	(vulnerability, exists, microsoft)	vulnerability	exists	microsoft	32	0	4	0
1	(execution, vulnerability, exists)	execution	vulnerability	exists	26	0	0	4
228	(fails, properly, handle)	fails	properly	handle	25	0	9	14
88	(handle, objects, memory)	handle	objects	memory	24	14	11	6
20	(software, fails, properly)	software	fails	properly	24	0	0	9
9	(properly, handle, objects)	properly	handle	objects	23	9	14	11
77	(information, disclosure, vulnerability)	information	disclosure	vulnerability	14	2	0	0
58	(vulnerability, affects, microsoft)	vulnerability	affects	microsoft	10	0	0	0
110	(execution, vulnerability, affects)	execution	vulnerability	affects	9	0	0	0
93	(specially, crafted, document)	specially	crafted	document	8	8	0	0
117	(order, trigger, vulnerability)	order	trigger	vulnerability	8	64	0	0
219	(document, order, trigger)	document	order	trigger	8	0	64	0
149	(microsoft, office, microsoft)	microsoft	office	microsoft	7	0	8	0
214	(disclosure, vulnerability, exists)	disclosure	vulnerability	exists	7	0	0	4
197	(office, microsoft, office)	office	microsoft	office	5	8	0	8
220	(microsoft, excel, viewer)	microsoft	excel	viewer	5	0	1	0

Fuente: Elaboración propia

Imagen 11. Data set 4grama

	4_gram	word_0	word_1	word_2	word_3	4-gram_freq	word_0_freq	word_1_freq	word_2_freq	word_3_freq
42	(execution, vulnerability, affects, microsoft)	execution	vulnerability	affects	microsoft	7	0	0	0	0
26	(information, disclosure, vulnerability, exists)	information	disclosure	vulnerability	exists	7	2	0	0	0
35	(vulnerability, affects, microsoft, word)	vulnerability	affects	microsoft	word	5	0	0	0	0
58	(information, disclosure, vulnerability, affects)	information	disclosure	vulnerability	affects	4	2	0	0	0
0	(improperly, discloses, contents, memory)	improperly	discloses	contents	memory	3	0	0	7	0
23	(execution, context, application, attacker)	execution	context	application	attacker	3	0	2	5	0
1	(specially, crafted, document, cause)	specially	crafted	document	cause	3	8	0	0	0
32	(information, disclosure, vulnerability, cve)	information	disclosure	vulnerability	cve	3	2	0	0	0
19	(context, application, attacker, must)	context	application	attacker	must	3	2	5	0	0
66	(specially, crafted, document, order)	specially	crafted	document	order	3	8	0	0	0
31	(microsoft, sharepoint, microsoft, office)	microsoft	sharepoint	microsoft	office	2	0	0	0	0
18	(disclosure, vulnerability, affects, word)	disclosure	vulnerability	affects	word	2	0	0	0	0
11	(microsoft, graphics, components, handle)	microsoft	graphics	components	handle	2	0	0	0	0
9	(affects, microsoft, sharepoint, server)	affects	microsoft	sharepoint	server	2	0	0	0	0
45	(vulnerability, affects, microsoft, sharepoint)	vulnerability	affects	microsoft	sharepoint	2	0	0	0	0

Fuente: Elaboración propia

7.5 **QUINTA FASE**

Por último, en la fase de deployment, se da cumplimiento al objetivo específico 4, que consiste en la implementación de una interfaz gráfica para la visualización de las estadísticas y tablas de datos generadas con una interacción del usuario que permita seleccionar parámetros tales como: Seleccionar de cual página desea realizar el web scraping, cual aplicación ofimática quiere analizar. Después de esto se genera la tabla de datos o dataframe donde el usuario puede realizar búsquedas específicas de una vulnerabilidad para esa aplicación.

En la parte de selección aplicación se puede escoger cual aplicación ofimática se desea realizar el análisis previamente mostrado y por lo cual el usuario esta en la capacidad de obtener de una manera rápida las vulnerabilidades especificas de cada aplicación y la frecuencia que aparecen. Posteriormente se desea implementar un filtrado por n-grama para que muestre los bigramas n-gramas deseados.

CONCLUSIONES

A partir de una adecuada identificación de las vulnerabilidades de seguridad en Microsoft Office, se logra establecer un contexto de ciberseguridad donde los usuarios y administradores de aplicaciones ofimáticas adquieren conocimientos generales para una comprensión asertiva de las implicaciones de seguridad informática que tiene el uso de herramientas de la suite de Microsoft Office y la información que allí se procesa o genera.

La seguridad informática es un proceso de niveles por consiguiente en el ámbito de los usuarios ya sean personales o empresariales es importante contar con herramientas que permitan establecer un nivel de madurez básico de la seguridad de la información.

Se debe tener en cuenta el archivo robots.txt en cualquier página web ya que es el encargado de restringir la extracción de información allí contenida para cualquier proyecto que implemente técnicas de web scraping.

La generación de bigramas permite dar claramente una idea de cuales son las vulnerabilidades que tiene la aplicación, pero se confirma en n-gramas posteriores.

El código implementado se puede usar para buscar palabras típicas en scripts maliciosos y poder buscar malware en páginas web. También se puede implementar para buscar malware en sitios de cryptojacking y búsqueda de palabras binarias de códigos maliciosos mientras se están ejecutando.

Dar una cultura y que se vaya alfabetizando a las personas que usan herramientas ofimáticas en los diferentes ambientes , ya sea de uso cotidiano , empresarial , de uso diario Y para auditores de sistemas otorgarle una base de donde debe empezar, es muy importante ya que puede identificar vulnerabilidades que no sabia que existían.

Recomendaciones a futuro

La implementación de diferentes tipos de n-gramas para establecer filtros mas precisos para la producción de graficas sin agrupación de palabras que generen ruido.

El procesamiento de lenguaje natural es un buen enfoque dependiendo de la fuente de datos, es decir, de las cantidades, se puede realizar dos tipos de aproximaciones (supervisada y no supervisada), en el primer caso, se necesitarías un buen conjunto de datos de vulnerabilidades reportadas para construir un bag of words de estas como un mecanismo de firmas que se pueda utilizar como insumo para un método de detección y si la construcción del anterior insumo de datos incluye palabras que dependen de sus consecuentes y antecedentes entonces se puede utilizar modelos más avanzados (grafos, redes neuronales profundas, regresiones, etc.) para identificar desde un nuevo texto vulnerabilidades que se han estudiado previamente.

Y si es un enfoque no supervisado, se puede tener una aproximación basada en anomalías, donde este enfoque sería construir un bag of words de cosas comunes (benignas - buenas líneas de código) de las cosas que se van a analizar, lo que serviría para construir un modelo que permita definir un umbral donde se pueda identificar cosas que no son particulares, una aproximación que se ha utilizado es Latent Dirichlet Allocation para construir un conjunto de tópicos que incluyen las palabras más frecuentes.

El análisis con NLP se ha explorado en la literatura con buenos resultados en modelos, sería recomendable en primer lugar evaluar los datos que se van a trabajar y luego ver las aproximaciones que se han mencionado, además, de centrar el alcance a una tecnología especifica ya sea web, móvil o una aplicación en concreto.

La técnica de web scraping puede ser muy útil para automatizar el proceso de conocer la aparición de una nueva vulnerabilidad en tiempos muy cortos y ayudando a estar preparado al instante.

REFERENCIAS BIBLIOGRAFICAS

Awad, Y., Nassar, M., & Safa, H. (2018). Modeling Malware as a Language. 2018 IEEE International Conference on Communications (ICC), 1–6.

Bernard. (s.f.). Obtenido de Forbes.com: https://www.forbes.com/sites/bernardmarr/2018/10/01/what-is-deep-learning-ai-a-simple-guide-with-8-practical-examples/#32389f338d4b

Ciberseguridad. (2017). Obtenido de https://ciberseguridad.com/amenzas/vulnerabilidades/desbordamiento-buffer/

Cisco. (2016). Obtenido de https://www.cisco.com/c/en/us/products/security/advanced-malware-protection/what-is-malware.html

Cisco. (2017). Obtenido de www.netacad.com/es/security/introduction-cybersecurity/vulnerability

Cisco. (2019). Obtenido de https://tools.cisco.com/security/center/resources/virus_differences

Cordero, P. (2008). *academa.edu.* Obtenido de https://www.academia.edu/12029545/APLICACIONES_PRÁCTICAS_UTILIZANDO_MICROSOFT_EXCEL_Y_WEKA

Microsoft. (2019). Obtenido de https://docs.microsoft.com/es-es/analysis-services/data-mining/data-mining-concepts?view=asallproducts-allversions

Microsoft. (2019). Obtenido de https://docs.microsoft.com/en-us/azure/architecture/data-guide/relational-data/etl

Minddata. (2018). Obtenido de https://minddata.org/what-is-ai-mit-stanford-harvard-cmu-Brian-Ka-Chan-AI

Oracle. (2017). Obtenido de https://www.oracle.com/co/database/what-is-a-relational-database/

Oracle. (2017). Obtenido de https://www.oracle.com/artificial-intelligence/what-is-machine-learning.html

Oracle. (2017). *Oracle.* Obtenido de https://www.oracle.com/co/database/what-is-a-relational-database/

Oracle. (2018). *Oracle.* Obtenido de https://www.oracle.com/co/business-analytics/

Oracle. (2018). *oracle.com.* Obtenido de https://www.oracle.com/co/business-analytics/

Oracle. (2018). *Oracle.com.* Obtenido de https://www.oracle.com/co/business-analytics/

Oracle. (2019). *Oracle.com/co/data-science.* Obtenido de https://www.oracle.com/co/data-science/what-is-data-science.html

Urbina, G. B. (2017). *Google Books.* Obtenido de https://books.google.com.co/books?id=IhUhDgAAQBAJ&pg=PA160&lpg=PA160&dq=ataque+entrada+no+validada+de+datos&source=bl&ots=0WSD1CrgKq&sig=ACfU3U2OdoJ2GMi419SlPhRDFrOHTBkqlw&hl=es&sa=X&ved=2ahUKEwjO_pr936fpAhWEmeAKHThLDF0Q6AEwBHoECAoQAQ#v=onepage&q=ataque%2