TRABAJO FIN DE GRADO

INGENIERÍA EN INGENIERÍA INFORMÁTICA

Analizador de mensajes de correo

Subtítulo del proyecto

Autor
Pedro Luis Fuertes Moreno

Directores
Alberto Guillén Perales
Gabriel Maciá Fernández

Escuela Técnica Superior de Ingenierías Informática y de Telecomunicación

—

Granada, mes de 201

Analizador de mensajes de correo: Subtítulo del proyecto

Pedro Luis Fuertes Moreno

Palabras clave: Correo electrónico, ciberseguridad, phishing, malware, virus

Resumen

Este proyecto surge debido a la falta de herramientas, tanto para usuarios técnicos como domésticos, para analizar un correo sospechoso una vez que llega a la bandeja de entrada.

El objetivo de este proyecto es, por tanto, intentar proporcionar una ayuda extra en la identificación de correos maliciosos mediante la extracción y relación de características comunes. Por otro lado, se ofrecerá un servicio público y funcional para que los usuarios puedan aprovechar toda la información recopilada y analizar sus propios correos.

A lo largo del documento se hablará de los distintos tipos de correos maliciosos, de algunas técnicas usadas por los atacantes para engañar o manipular a sus víctimas, de los patrones que se van a extraer y cómo, de las relaciones que se van a hacer.

En la parte de diseño se analizarán y compararán tanto lenguajes de programación como bases de datos, teniendo la idea en mente de migrar todo el servicio a la nube para tener un SaaS, siendo especialmente relevante el servicio de Azure Functions para ejecutar el código. En la parte funcional, el servicio debe permitir tanto analizar como buscar resultados, así como mostrar información relevante derivada del análisis.

También se indicarán problemas encontrados, soluciones aplicadas y posibles mejoras.

Finalmente, y como objetivo último se intentarán obtener ingresos económicos por parte del servicio de cara a crear una posible startup o venta del servicio.

Analizador de mensajes de correo: Subtítulo del proyecto

Pedro Luis Fuertes Moreno

Keywords: e-mail, cybersecurity, phishing, malware, virus

Abstract

Write here the abstract in English.

Yo, **Pedro Luis Fuertes Moreno**, alumno de la titulación Grado en ingeniería informática de la **Escuela Técnica Superior de Ingenierías Informática y de Telecomunicación de la Universidad de Granada**, con DNI XXXXXXXXX, autorizo la ubicación de la siguiente copia de mi Trabajo Fin de Grado en la biblioteca del centro para que pueda ser consultada por las personas que lo deseen.

Fdo: Pedro Luis Fuertes Moreno

Granada a X de mes de 201 .

D. **Alberto Guillén Perales**, Profesor del Área de XXXX del Departamento Departamento de ... de la Universidad de Granada.

D. **Gabriel Maciá Fernández**, Profesor del Área de XXXX del Departamento Departamento de ... de la Universidad de Granada.

Informan:

Que el presente trabajo, titulado ***Analizador de mensajes de correo, Subtítulo del proyecto***, ha sido realizado bajo su supervisión por **Pedro Luis Fuertes Moreno**, y autorizamos la defensa de dicho trabajo ante el tribunal que corresponda.

Y para que conste, expiden y firman el presente informe en Granada a X de mes de 201 .

Los directores:

Alberto Guillén Perales **Gabriel Maciá Fernández**

Agradecimientos

A mi familia, en especial a mi padre, a mi madre, a mi hermano y a mi hermana.

A mis amigos.

A mis profesores.

Índice general

Capítulo 1

Introducción

Aunque en la actualidad poseemos una gran cantidad de aplicaciones con las que comunicarnos con otras personas, el correo electrónico sigue siendo una de las herramientas de comunicación más usadas, especialmente en el mundo empresarial.

Esto es debido a que fue uno de los primeros servicios en permitirnos enviar y recibir tanto texto como otros tipos de archivos de manera rápida y sencilla.

Según un estudio de hace tres años de The Radicati Group el correo electrónico tenía 3.7 mil millones de usuarios, que enviaban 269 mil millones de correos cada día. [1]

Viendo las cifras anteriores se puede entender la importancia de tener un servicio de este tipo lo más seguro posible, ya que un ataque bien diseñado puede afectar a millones de personas.

Sin embargo, debido a cómo y cuándo crea, la seguridad nunca ha sido uno de sus puntos fuertes y eso ha llegado hasta nuestros días.

1.1. Breve historia del correo electrónico

El correo electrónico se remonta a 1962 en el MIT, cuando compraron a IBM un ordenador que permitía que distintos usuarios iniciaran sesión y guardaran archivos en él. Estos lo aprovecharon para intercambiar mensajes, lo que provocó que para 1965 se desarrollara un servicio que facilitase esa comunicación entre los distintos usuarios y lo llamaron MAIL.

Hay que tener en cuenta que, en ese servicio, los mensajes no salían de dicho ordenador. Habría que esperar hasta 1971 para ver lo que sería el primer "correo electrónico" enviado a través de una red, en concreto de ARPANET y fue gracias a Ray Tomlinson, que adaptó un programa que permitía enviar mensajes a distintos terminales de distintos usuarios de un mismo ordenador, para poder enviar mensajes entre distintos terminales, aunque no estuviesen en el mismo ordenador. Precisamente el "@" del co-

rreo electrónico viene de la necesidad de Tomlinson de tener que separar al usuario del equipo, ya que anteriormente esto no era necesario.

No es hasta 1977 cuando se crea el primer rfc del correo electrónico, concretamente el rfc733 [2], aunque, este protocolo no es usado en la actualidad. El primer rfc del primer protocolo que aún se usa es el rfc821 [3] de SMTP de 1982 y el rfc918 [4] de POP de 1984.

La situación por aquél entonces de lo que ahora conocemos como Internet, era muy distinta. Internet estaba reservado a universidades, centros de investigación e instituciones gubernamentales. Esto hizo que cuando se desarrollasen estos protocolos, no se pensara en la seguridad de ellos.

Y este es uno de los grandes problemas que tiene el correo en la actualidad, ya que, aunque tanto SMTP como POP (E IMAP [5] aunque no se ha mencionado antes) han ido recibiendo actualizaciones, están basados en unos protocolos diseñados y pensado para un entorno radicalmente diferente en el que se siguen utilizando.

1.2. Origen del proyecto

Este proyecto surge debido a la gran cantidad de demandas que he recibido en los últimos años por parte de familiares y amigos para que, les ayudase a verificar si un correo sospechoso que les había llegado a su bandeja de entrada era o no malicioso.

Y esta tarea, que, para mí era trivial en la mayoría de los casos, no lo era para ellos. Aunque algunos estaban tan bien preparados que incluso a mí me costaba diferenciarlos.

Todo esto me llevó a pensar dos cosas. La primera, que una vez que un correo llega a la bandeja de entrada del usuario, sólo le queda su intuición para confiar o no en el mensaje, intuición que puede fallar incluso si se tienen conocimientos técnicos.

Por otro, que en caso de querer investigar dicho mensaje y obtener más información, no existe ninguna herramienta específica para ello, por lo que se tiene que hacer todo a mano y siendo imposible obtener algún tipo de relación con otro mensaje parecido, lo cual dificulta mucho el proceso.

1.3. El proyecto

El objetivo de este proyecto es tratar de paliar esta carencia de herramientas tanto para la identificación de correos maliciosos una vez que han pasado los filtros de spam, como para la investigación de un mensaje en los casos más complicados.

Por este motivo se va a crear por un lado, un servicio que permita analizar mensajes *on-line*, para que cuando se reciba un mensaje sospechoso, el usuario tenga una segunda validación con más información y por otro lado,

se va a permitir al usuario poder "navegar" entre los datos encontrados en el correo para poder obtener más información, lo que puede facilitar en gran medida un análisis más profundo del correo por parte de una persona más técnica en caso de ser necesario.

Para llevar esto acabo se debe crear una gran base de datos con todos los correos analizados, así como los distintos datos que se han extraído de los mismos y sus relaciones.

También se deben identificar patrones maliciosos conocidos, para poder alertar a los usuarios menos especializados sobre los patrones encontrados sin necesidad de que ellos sepan en qué consisten.

Finalmente, [como ejercicio académico] se presentará un estudio de modelo de negocio para poder hacer económicamente viable esta propuesta de servicio...

Capítulo 2

Motivación

En la actualidad los ataques por correo electrónico siguen siendo un hecho y prácticamente todo el mundo ha recibido algún correo de este tipo alguna vez, lo que demuestra que, las herramientas actuales no son capaces de solucionar de manera efectiva este problema.

Esto se suma a que cada vez los ataques son más y más sofisticados, y, por tanto, complicados de detectar, ya no solo por estas herramientas, sino, por las propias personas, sean o no profesionales del sector. Y es que cuando un usuario recibe un mensaje de este tipo no tiene ninguna herramienta extra que le ayude a comprobar si es o no malicioso.

Esto es así hasta tal punto, que la solución para identificar estos correos de grandes empresas antivirus dedicadas a la ciberseguridad es que los usuarios sigan su intuición [6], otras ofrecen pequeños cursos para identificarlos [7].

Todo esto refleja como los usuarios están en una clara situación de vulnerabilidad, especialmente los usuarios menos técnicos que no conocen cómo funciona realmente la tecnología y que están usando ¿Y es que acaso deberían?

Bajo mi punto de vista, el enfoque de las grandes compañías sobre cómo atacar los problemas de seguridad que tienen el correo electrónico, está dirigido a personas con unos conocimientos que la mayoría de las personas no tienen y por ta nto sólo es útil para una minoría de usuarios del servicio.

Pero además, es que dichas compañías tampoco están exentas de ataques de esta naturaleza, esto se pone de manifiesto en algunos ataques llevados a cabo con éxito a empresas tecnológicas de máximo nivel, como pueden ser Google o Facebook, a las que un atacante les consiguió robar 121 millones de euros [8]

Todo esto lleva a pensar que actualmente sigue habiendo un gran agujero de seguridad en el correo electrónico y que las herramientas existentes no son suficientes ni siquiera para los expertos del sector.

2.1. Aplicaciones relacionadas

A continuación, se van a analizar un conjunto de herramientas que, si bien no ofrecen soluciones completas a este problema, pueden ser buenos servicios en los que apoyarse para tratar de dar una solución más amplia y completa que las actuales.

2.1.1. Herramientas específicas del correo electrónico

Las herramientas descritas en esta sección únicamente son válidas en correos electrónicos, por lo que, si en el futuro también se desean analizar otro tipo de mensajes, o no se tienen todas las cabeceras asociadas al mensaje, no serán válidas.

Filtros de correo no deseado

Tal vez sea la mejor solución que se tiene actualmente para mitigar este tipo de problemas. Son filtros que analizan cada uno de los mensajes que se reciben y en base a distintos criterios informan al usuario si el correo es o no legítimo.

Aunque, este tipo de herramientas tienen varios problemas asociados:

- Su efectividad: Conseguir una herramienta con una efectividad del 100 % es prácticamente imposible y esto es algo que se debe asumir, sin embargo, hay ciertos filtros cuya efectividad bastante reducida.
- La imposibilidad de poner un filtro de este tipo: hay muchos servicios de correo electrónico que no permiten al usuario poner un filtro personalizado de correo no deseado. Ejemplos de estos servicios son Gmail u Outlook, 1500 [9] y 400 [10] millones de usuarios respectivamente (¡Son muchos!)
- Una vez que el mensaje pasa el filtro, esta herramienta deja de ser efectiva, lo cual puede ser muy sencillo para un atacante, simplemente va probando con distintos correos hasta que consigue uno que lo pase.

Un servicio de este tipo puede ser muy útil para un primer análisis si el mensaje es compatible. Normalmente tienen una larga trayectoria, son rápidas y tienen una efectividad comprobada.

2.1.2. Analizador de cabeceras

Un analizador de cabeceras de correo electrónico da información relevante y normalmente invisible al usuario sobre dicho correo. Estas cabeceras pueden aportar información muy útil como por ejemplo la ruta seguida por

el mensaje, la ip del servidor de correo desde donde se envió o si ha pasado o no los filtros antispam del proveedor de correo.

Actualmente hay varias páginas que ofrecen este servicio, como por ejemplo Google [11], Mircrosoft [12] o Mxtoolbox [13].

Ofrecer este tipo de análisis es muy interesante, ya que permite de manera sencilla hacer un análisis más profundo por parte de un profesional del sector.

2.2. Herramientas generales

Las herramientas que se describen a continuación, si bien no están destinadas al correo electrónico como tal, se pueden usar de manera efectiva para analizar los distintos mensajes que se reciban, sean o no correos electrónicos.

Virus Total

Virus Total [14] es una web propiedad de Google que permite analizar archivos, y direcciones web en busca de programas malignos y aunque no está directamente relacionada con el correo electrónico puede servir de ayuda para analizar posibles archivos adjuntos, así como posibles direcciones sospechosas.

Tener una herramienta que enlace directamente con el servicio puede ser de gran ayuda tanto para profesionales del sector como para usuarios menos especializados, que lo único que tendrán que hacer es clicar en un botón para analizar una dirección de su correo electrónico.

Virus total ofrece una api rest que permite analizar tanto archivos como urls, dominios e ips. [15]

Metadefender

Metadefender [16] es un analizador de archivos, url's, dominios e ips similar a Virus Total de la empresa de ciberseguridad Opswat.

Puede ser una buena alternativa a Virus Total y al igual que este tiene una api pública [17] en la que realizar consultas.

Have I been pwned

Have I been pwned [18] es una web que permite saber si una dirección de correo electrónico ha aparecido en alguna brecha de seguridad, y en caso de que haya aparecido te dice en qué brecha ha sido.

Saber si el mensaje proviene de una dirección comprometida puede ser de relevancia, siendo más probable que un mensaje malicioso provenga de una dirección comprometida que de una dirección que no lo sea.

Have I been pwned tiene una api [19] donde realizar consultas sobre direcciones de correo electrónico, además su base de datos se va actualizando

con las últimas brechas de seguridad que van surgiendo y contando ya con más de 9.500 millones de cuentas de correo.

Capítulo 3

Objetivos y requisitos

3.1. Obtención y relación de patrones

Sacar mediante expresiones regulares datos de interés de correos electrónicos y relacionarlos entre sí. Esto permitirá a los expertos del sector contar con una herramienta para poder realizar análisis de mayor profundidad al poder relacionar datos comunes entre los distintos correos de la base de datos.

Un ejemplo sencillo de esto puede ser el análisis de un correo electrónico nuevo, pero con una url ya conocida y presente en otros correos electrónicos.

Un ejemplo más complejo podría ser, el análisis de un mensaje con una url nueva, pero con un dominio de cuya IP sí se tienen registros, lo que puede dar lugar a una nueva línea de investigación sobre si dicho dominio pertenece (O no) a un cibercriminal, aunque no se tengan registros previos ni del dominio ni de la url en cuestión.

Se debe extraer al menos los siguientes tipos de datos:

- Direcciones IP
- Dominios
- Enlaces
- Direcciones de correo electrónico

Aunque el servicio se debe pensar para que en un futuro se puedan extraer más tipos de datos como carteras de criptomonedas o números de teléfono.

3.2. Evaluar de manera relativa la maliciosidad

Se deben detectar ciertas técnicas comúnmente usadas por los ciberdelincuentes para atacar a sus víctimas y así asignar un valor de maliciosidad tanto a los datos extraídos como a los mensajes en sí.

Algunas de estar técnicas podrían ser:

- Mostrar un enlace distinto al que se redirige.
- Usar una gran cantidad de subdominios de subdominios
- Mostrar un correo electrónico distinto del real

3.3. Ofrecer un servicio de análisis público

La segunda parte del proyecto será ofrecer a todos los usuarios la posibilidad de analizar sus mensajes mediante una web donde podrán o bien copiar y pegar el mensaje, o bien subir un archivo eml para un análisis más completo.

La página devolverá varias listas con todos los tipos datos extraídos del análisis, así como enlaces a cada uno de ellos donde se muestre un informe más detallado.

Esto permitirá que a un usuario que le llegue un correo y no sepa si es o no malicioso, pueda analizarlo en la página y obtener más información sobre este.

3.4. Cumplir de manera efectiva con la ley de protección de datos

Es de vital importancia diseñar el servicio pensando en las leyes de protección de datos, especialmente en la europea por ser la más estricta hasta el momento.

Los usuarios deben poder eliminar, tanto los mensajes, como los datos obtenidos de ellos si así lo desean de manera automática y sin necesidad de que nadie intervenga en el proceso.

Aunque esta característica no se implemente en este proyecto, sí que se debe pensar el sistema para que se pueda implementar en el futuro de manera sencilla y modificando la mínima cantidad de código posible.

3.5. Monetización

Como parte complementaria se intentará pensar cómo obtener ingresos económicos del desarrollo. Estos ingresos nunca deben impedir que usuarios particulares puedan usar el servicio, aunque sí pueden impedir acceder a todas las funcionalidades de este.

3.6. Integración con terceros

Aunque esto está un poco al margen del TFG, sería muy interesante la integración directa desde la página con otros servicios como el de Virus Total o la de Have I been pwned.

Esto permitiría por un lado facilitar a los usuarios menos técnicos un análisis rápido desde la propia página y por otro puede dar información relevante a los expertos.

3.7. Crear un servicio funcional

Al final del proyecto se debe crear un servicio que dé un soporte real y online, no se debe limitar a tener un servicio local únicamente de prueba. Por lo que durante el apartado del diseño se deben elegir tecnologías factibles para su posterior despliegue en internet y por tanto no limitadas a un entorno de local pruebas.

Capítulo 4

Análisis y diseño

4.1. Análisis previo

Durante esta sección se analizarán cuestiones importantes que afectarán a múltiples secciones posteriores y que es necesario plantearse antes de comenzar, ya que pueden afectar de manera muy significativa a la elección y posterior diseño de futuros apartados.

De manera general, se deben tener en cuenta los siguientes factores:

- El tipo de datos que se van a guardar.
- El tipo y la cantidad relaciones entre los datos.
- La forma de identificar los datos.
- La forma de mostrar los datos.
- Cómo se va a buscar información y qué búsquedas se van a realizar.
- La cantidad de información que se va a almacenar.

4.1.1. Tipos de datos

Dentro de los tipos de datos se tienen que distinguir dos tipos, por un lado, están los mensajes que el usuario envía y por otro los datos que se extraen de ellos.

Tipos de datos a analizar

Dentro de los mensajes, este proyecto se centrará en mensajes de correo electrónico, por lo que hay fundamentalmente de tres tipos de datos, en texto plano, en HTML [20] y en formato EML.

A la hora de elegir un lenguaje, contar con alguna librería que sea capaz de analizar dichos tipos de archivos es algo crítico, puesto que hacer un

analizador de dichos tipos sería muy costoso y llevaría demasiado tiempo. Mientras la mayoría de los lenguajes modernos tienen librarías para leer archivos en HTML, no ocurre los mismo para el formato EML.

Sin embargo, el diseño no debe limitar la implementación de otros tipos de archivos tales como CSV, XML o JSON.

Tipos de datos que se van a extraer

En un principio los datos que se van a extraer son enlaces, dominios, direcciones de correo, direcciones IP y en el caso de los archivos en formato EML, sus cabeceras.

4.1.2. Tipo y cantidad de relaciones

El tipo de relaciones serán normalmente de muchos a muchos, ya que de un único mensaje se pueden extraer varios datos de un mismo tipo, que a vez pueden estar múltiples mensajes.

Respecto a la cantidad de relaciones, mientras que de un correo no se deben extraer demasiados datos, un dato concreto puede aparecer en una gran cantidad de correos, por lo que en función del diseño que se haga, recuperar todos los correos en los que aparece un determinado dato podría llegar a ser una operación muy costosa.

4.1.3. Forma de identificar los datos

En este proyecto y debido a la naturaleza de los datos que se tratan, algo que pudiera ser trivial como es el hecho de identificar un dato concreto, se vuelve una tarea más compleja, pues estos pueden ser relativamente grandes y no se deben tratar de la misma manera que otros de menor tamaño, como tal vez puede ser un número o una fecha.

También es importante el hecho de poder "navegar" mediante enlaces, ya que no tendría sentido poner como dirección de un mensaje su propio contenido.

Este problema se puede solucionar de múltiples formas, una de ellas, consistiría en asignar a cada dato un valor numérico e incremental, de modo que se podría acceder al dato número 1, 2, …, n. Este sistema permite tener un control del número de elementos que se han analizado, además ocupa muy poco (con 4 bytes por elemento se podrían numerar 4.294.967.296 elementos de dicho tipo), permitiría hacer búsquedas rápidas, al poder guardar en memoria una gran cantidad de elementos y no guarda relación alguna con el elemento al que identifica.

Sin embargo, también presenta los siguientes problemas, y es que, al no guardar relación con el elemento, no se puede obtener el identificador únicamente con el valor del dato, por lo que requiere de una consulta del

valor completo a la base de datos, que, en el caso de un mensaje, puede ser de gran tamaño.

Para evitar esta falta de relación entre el valor de un elemento y su identificación se puede utilizar una función hash, que sea rápida y que devuelva valores pequeños. Esto solucionaría la falta de correlación entre identificador y valor, facilitaría la navegación, permitiría búsquedas rápidas, (Constantes en teoría) y simplificaría las búsquedas de datos grandes como los mensajes.

Aunque también presenta problemas, el mayor de ellos es que ocupa mucho más espacio que una identificación numérica, lo que, en un sistema relacional, puede hacer que las uniones de tablas sean mucho más lentas. Además, también se pierde el orden de inserción, pero tampoco es importante en este caso.

Por tanto, una solución interesante puede ser adoptar ambos modelos, por un lado, tener un índice numérico e incremental y por otro un índice de tipo hash para buscar en caso de no tener el identificador numérico. Por ejemplo, para comprobar si un mensaje ya ha sido o no analizado se podría utilizar su hash, mientras que para las uniones de tablas de podría utilizar el identificador numérico.

Esta solución también presenta el problema de sobrecoste que conlleva guardar el hash y que puede ser mayor o menor en función de los bits que ocupe la función elegida.

Para ello, ha hecho un pequeño estudio del sobrecoste generado por cada millón de elementos insertados según algunas de las funciones hash más conocidas en este momento. No se va a tener en cuenta el coste computacional de dicha función al considerarlo relativamente bajo en todas ellas.

También se va a considerar la probabilidad de colisiones suponiendo que todos los valores son equiprobables. Para hacer este cálculo y teniendo en cuenta la paradoja del cumpleaños, se va a calcular la probabilidad de colisiones mediante la siguiente fórmula $k = \sqrt{2^n}$ siendo n el tamaño del hash en bits y k la probabilidad de que haya una colisión.

Función hash	Tamaño del hash (Bytes)	Probabilidad de colisión	Sobrecoste generado (Por cada millón) MB
MD5	16	1.84467×10^{19}	15.25879
SHA1	20	1.20893×10^{24}	19.07349
SHA256	32	3.40282×10^{38}	30.51758
SHA512	64	1.15792×10^{77}	61.03516

Tabla 4.1: Comparativa de las distintas funciones hash

En base a la tabla 4.1, puede verse que el sobrecoste por cada millón de documentos es completamente asumible en todos los casos, pues incluso con 10 millones de elementos, en el peor de los casos, es decir usando la función SHA512, los hashes tendrían un coste de tan solo 610MB, lo que no es algo exagerado teniendo en cuenta la cantidad de elementos que se tendrían y las ventajas que se obtienen.

Sin embargo, y dado que el hash se va a utilizar únicamente como identificador, es mucho más interesante usar una función que genere un hash de menor tamaño como podría ser MD5 o SHA1 ya que el tamaño es similar.

Finalmente se va a optar por SHA1 debido a que MD5 en la actualidad está roto, y aunque esto no debería afectar directamente al servicio, pues no se pretender obtener ningún tipo de seguridad, sí que se podría aprovechar esta vulnerabilidad por parte de algún ciberdelincuente añadiendo él mismo un mensaje falso que genere el mismo hash que un posible correo malicioso enviado por el mismo ciberdelincuente, evitando de esta manera que sea analizado en la plataforma.

A la hora de la elección de un lenguaje de programación, será necesario que este cuente con dicha función o en su defecto, que haya alguna implementación de esta ya desarrollada, pues no es objetivo de este trabajo desarrollarla.

4.1.4. Forma de mostrar los datos

La forma de mostrar los datos puede indicar o al menos sugerir de qué manera se deben almacenar para facilitar su posterior visualización.

En el caso de este proyecto, dado un elemento cualquiera debe existir un enlace a todos los elementos que estén relacionados con él y cuyo texto será su valor o en caso de ser muy extenso, su hash.

De este modo la visualización de los distintos elementos queda como sigue:

Visualización de un mensaje

En la visualización de un mensaje se tendrá que poder ver, toda la información referente a él (Tipo de archivo, fecha de análisis, score, …), y todos los datos extraídos, así como un enlace a los mismos (Enlaces, dominios, direcciones IP, …)

Visualización de un dato cualquiera extraído de un mensaje

En la visualización de cualquier dato extraído de un mensaje será importante que además de la información relacionada con el mismo (Valor, fecha de análisis, score, …), se puedan listar todos los mensajes en donde ha aparecido dicho dato, así como un enlace a cada uno de ellos. El texto mostrado en este caso será el hash dl mensaje.

El poder listar los mensajes donde aparece un dato concreto es importante para poder llevar a cabo investigaciones, o identificar nuevos mensajes perniciosos en base a elementos comunes con otros mensajes ya analizados.

4.1.5. Cómo se va a buscar información y qué búsquedas se van a realizar

Saber qué búsquedas van a hacer y con qué datos se cuenta para llevarlas a cabo es necesario para determinar por un lado cómo se guarda la información y, por otro lado, las relaciones necesarias para que se puedan realizar las dichas consultas.

En este caso, dado un documento se debe poder obtener a cada uno de los datos extraídos de él y dado un dato cualquiera, tienen que poder obtenerse tanto los correos en los que aparece, como otros elementos relacionados con él. Por ejemplo, dado un domino, además de los mensajes donde está presente, también se debe obtener información sobre todos los enlaces analizados pertenecientes a dicho dominio.

Las búsquedas generalmente serán de elementos concretos, es decir, no será común realizar búsquedas por rango. Tampoco será común usar los operadores de mayor o menor.

Se debe poder buscar un elemento tanto por su valor, o por su hash en caso de ser un elemento de gran tamaño, así como por su identificador numérico, ya que su búsqueda puede ser mucho más rápida.

En un principio no será habitual buscar elementos por información relativa a ellos, por ejemplo, elementos analizados en una determinada fecha o con una determinada característica.

También es importante señalar que al principio las operaciones de inserción serán las más comunes y que a medida que se vaya haciendo uso del servicio, las operaciones de consulta serán las que prevalecerán. Además, es necesario que el servicio no sea demasiado lento, pues no sería práctico, por este motivo, debe prevalecer la velocidad de consulta sobre la inserción.

4.1.6. Cantidad de información que se va a almacenar

Respecto a la cantidad de información a almacenar, el sistema debe estar preparado para guardar un gran volumen de información, de decenas o cientos de millones de correos electrónicos, por lo que contar con un sistema escalable es crítico.

A la hora de realizar este informe, se cuentan con más de 8.5 millones de correos electrónicos perniciosos.

4.2. Información sobre los datos que se va a extraer

Como ya se ha comentado con anterioridad, uno de os objetivos del proyecto es extraer distintos tipos de datos y relacionarlos con otros correos electrónicos.

En concreto se quiere extraer:

- Direcciones IP
- Dominios
- Enlaces
- Direcciones de correo electrónico

Una forma sencilla de identificar y extraer este tipo de datos relativamente bien definidos es mediante expresiones regulares. El hacerlo de esta manera requiere que se sepa de manera muy precisa cómo están formados, qué símbolos tienen o no, qué tamaño,...

Por este motivo, se va a hacer un análisis exhaustivo de los distintos tipos de datos que se quieren obtener, especialmente en lo que a estructura, formato, tamaño y símbolos se refiere. Sin embargo, el análisis será eminentemente práctico, sin entrar detalles técnicos que no tengan relevancia a la hora de crear una expresión regular para evitar que la longitud de la memoria crezca en exceso.

De esta manera se podrán crear unas expresiones regulares muy precisas que generen el mínimo número de falsos negativos posibles.

4.2.1. Direcciones IP

Una dirección IP es un conjunto de números y/o letras que identifican de manera única a cada uno de los dispositivos de una red.

Existen dos versiones, la versión 4 y la versión 6, cada una tiene un formato distinto. [21]

IP versión 4 (IPv4)

Cuando se habla de IPv4 es importante mencionar que su formato no se ha especificado como tal en ningún RFC, quedando definido por el uso y por cómo fue descrito en otros RFCs. Con esto en mente se va a usar el RFC790 [22], donde se realiza la asignación de clases de redes, a distintos grupos de direcciones IP, para definir su formato.

Como se puede observar en dicho RFC, se podría decir que por convención una dirección IP en versión 4 está definida por un conjunto de cuatro números, separados entre sí por un punto y cuyo valor del 0 al 255. Pueden

ser escritos tanto con ceros a la izquierda o sin ellos, o lo que es lo mismo, tanto el 3 como el 03, como el 003 son válidos y tienen exactamente el mismo valor. Esto hace que una misma dirección pueda ser escrita de múltiples formas, por ejemplo 192.168.0.1 podría ser escrita como 192.168.000.001 ò 192.168.00.01 ò 192.168.000.01, etc.

IP versión 6 (IPv6)

La representación textual de una dirección IPv6 es muy flexible lo que hace que sea mucho más complicado crear una expresión regular para identificarlas. Su formato de definió en el RFC 4291, en la sección 2.2 [23], aunque debido a que ciertos operadores estaban teniendo problemas por su flexibilidad, la IEFT publicó el RFC 5952 [24] con ciertas recomendaciones sobre su formato escrito para facilitar la implementación del protocolo.

En líneas generales una dirección IP en versión 6 está representada por 8 números hexadecimales de cuatro cifras, separados entre sí por dos puntos verticales (":") y escritos en normalmente en minúsculas, aunque también pueden estar escritos en mayúsculas.

A continuación, se van a comentar algunas de las posibles variaciones a la hora de representar textualmente una dirección de este tipo.

Omitir ceros a la izquierda Los 0 a la izquierda pueden (o no) ser omitidos y el valor de la dirección no se altera. Por ejemplo, las siguientes direcciones, pese a estar escritas de distinta forma, tienen el mismo valor.

2001:db8:aaaa:bbbb:cccc:dddd:eeee:0001
2001:db8:aaaa:bbbb:cccc:dddd:eeee:001
2001:db8:aaaa:bbbb:cccc:dddd:eeee:01
2001:db8:aaaa:bbbb:cccc:dddd:eeee:1

Contraer los ceros Si en una dirección uno o más números consecutivos son cero, se pueden eliminar. Esto puede hacerse una única vez.

Por ejemplo:
2001:db8:aaaa:bbbb:0:0:0:1
2001:db8:aaaa:bbbb::1

Precisamente esta característica es la que hace sea complicado analizar una dirección IP en su versión 6, ya que múltiples opciones son posibles y todas correctas.

Direcciones IPv4 embebidas Una dirección IPv6 puede tener embebida una dirección IPv4 al final de esta, que será escrita con el mismo formato de una dirección IPv4.

Por ejemplo
0:0:0:0:0:0:13.1.68.3
0:0:0:0:0:FFFF:129.144.52.38

Además, si se tiene en cuenta la propiedad anterior, podrían contraerse los ceros, quedado como sigue:
::13.1.68.3
::FFFF:129.144.52.38

4.2.2. Dominios

Un dominio es una cadena de caracteres que está asociada a una o varias direcciones IP. Está regulados por un organismo internacional llamado ICANN del inglés (Internet Corporation for Assigned Names and Numbers; ICANN) [25] y su especificación viene recogida en el RFC 1035 [26]

Formato

Según este RFC un dominio está formado por dos o más partes separadas entre sí por un punto «.».

Siendo el formato como sigue:
[<Subdominio>.]<nombre de dominio>.[<SLD/2LD>.]<TLD>
Donde:

- Subdominio: Los subdominios subpartes del dominio al que preceden. Puede haber tantos como se quiera.
- Nombre de dominio: Se registra en el ICANN o la autoridad competente dependiendo del TLD o del SLD.
- SLD/2LD: Son conocidos como dominios de segundo nivel y permiten una especificación de un dominio de primer nivel. Por ejemplo, «co.es» podría estar destinado a empresas comerciales españolas. Todos los SLDs registrados pueden ser consultados en Publicsuffix [27].
- TLD: Son conocidos como dominios de primer nivel y son los gestionados por el ICANN. Todos los TLDs registrados se pueden encontrar en la web del IANA [28]

Tener en cuenta los TLDs y los SLDs es muy importante para asignar correctamente los dominios y los subdominios, ya que no tener en cuenta los SLDs podría hacer que se considerasen nombres de dominio que no lo son.

Además, es una buena forma de detectar falsos positivos cuando se están buscando dominios en un documento de texto.

Tamaño

En la sección “2.3.4. Size limits” del RFC 1035, se indica que el tamaño máximo para el dominio son 255 completo caracteres y el de casa sección de 63 caracteres, aunque en la actualidad no suelen ser tan largos.

Caracteres permitidos

Así mismo en la sección "2.3.3. Character Case" se indica que la codificación de un dominio debe ser ASCII e insensible a mayúsculas y minúsculas, aunque no obliga a ello. En el RFC 3490 [29], se recoge la posibilidad de usar caracteres Unicode aunque estos sean representados internamente como caracteres ASCII, en la práctica, los caracteres Unicode no son muy usados.

Además, y según se especifica en el RFC 952 [30] y en el RFC 1123 [31], los únicos caracteres válidos dentro de un dominio son los caracteres alfanuméricos (A-Z) y (0-9), el guion medio (-) y el punto (.), aunque algunos servidores también permiten el guion bajo (_).

4.2.3. Enlaces (URLs)

Los enlaces o URLs (del inglés Uniform Resource Locator) vienen definidas por el RFC 1738 [32], de especial interés la sección "3. Specific Schemes". Una URL es a su vez un tipo de URI (del inglés Uniform Resource Identifier) definido en el RFC 3986 [33], de este último el apartado más interesante para el proyecto es la sección "3.3. Path" en el cual se especifica el formato que debe tener y también en el "Apéndice A: Collected ABNF for URI".

También es interesante el RFC 3696 en la sección "4.1. URI syntax definitions and issues" [34], ya que se centra en las URLs cuyo esquema sea HTTP o HTTPS, y por tanto el más interesante para el proyecto .

Formato

El formato viene definido como sigue:
http/https://<dominio>[:<puerto>][/<ruta[<parámetros>]>]
Donde:

- http/https: Define el protocolo y aparecerá uno u otro.
- Dominio: su formato ya ha sido definido.
- Puerto: No es muy usado, es un valor numérico que va del 0 al 65535.
- Ruta: son distintas cadenas de texto que dan acceso a cada una de las páginas de un mismo dominio.
- Parámetros: Son posibles valores que envían información extra al servidor.

Caracteres permitidos

Los caracteres permitidos en una URL son:

- Cualquier carácter alfanumérico.

- Cualquiera de los símbolos siguientes: - . _ ~ ! $ & ' ( ) [ ] * + , ; = : @ # ?
- El carácter % precedido de un número hexadecimal de dos cifras.

Tamaño máximo

En principio una URL no tiene límite de tamaño tal y como se recoge en el RFC 7230 al final de la sección "3.1.1. Request Line" [35], sin embargo y como también se menciona en el mismo RFC, sí hay ciertas restricciones por parte de los navegadores. Por ejemplo, la longitud máxima de una URL si se usa Internet Explorer es de 2083 caracteres en total [36].

4.2.4. Dirección de correo electrónico

La sintaxis de una dirección de correo electrónico viene definida en el rfc 5322 en la sección "3.4.1. AddrSpec Specification" [37], sin embargo, en la definición de este RFC se centra mucho en la parte del dominio, que ya ha sido definida, por lo que es más útil el RFC 3696, concretamente su sección "3. Restrictions on email addresses" [38]

Formato

El formato de una dirección de correo electrónico es de la siguiente forma:
<Parte local>@<dominio>
Siendo:

- Parte local: es la parte de la dirección que define al usuario.
- Dominio: indica el dominio al que pertenece la dirección.

Caracteres permitidos

Los caracteres permitidos en el dominio ya han sido definidos.

Los caracteres permitidos en la parte local pueden ser, cualquier carácter ASCII siempre que vaya entre comillas dobles o escapeado mediante el símbolo "\". No es necesario ni escapar, ni entrecomillar los caracteres alfanuméricos, ni los siguientes caracteres especiales entre paréntesis (! # $ % & ' * + - / = ? ^ _ ` . ~ { } |).

El punto ("."), pese a estar permitido, no puede estar ni en la primera, ni en la última posición, tampoco puede haber más de dos seguidos.

Tamaño máximo

El tamaño máximo de la parte local es de 64 caracteres, por lo que sumados a los 255 de la parte del dominio y el "@", da un máximo de 320 caracteres.

4.1. Elección del lenguaje de programación para la extracción de datos

4.1.1. Java

4.1.2. C++

4.1.3. Pyhton

4.1.4. Node.js (JavaScript)

4.1.5. PHP

4.2. Elección de la base de datos en la que almacenar los datos

4.0.1. NO-SQL

MongoBD

Neo4j

4.0.2. SQL

MariaBD

PostgreSQL

4.1. Visualización de los datos

4.2. Adecuación a la ley de protección de datos

Capítulo 5

Implementación

5.1. Expresiones Regulares

5.2. MySQL

5.3. PHP

Capítulo 6

Pruebas

Capítulo 7

Conclusiones y vías futuras

Bibliografía

[1] The Radica ti Group. Email statistics report, 2017-2021. http://www.radicati.com/wp/wp-content/uploads/2017/01/Email-Statistics-Report-2017-2021-Executive-Summary.pdf, 2017.

[2] Internet Engineering Task Force. Standard for the format of arpa network text messages. https://tools.ietf.org/html/rfc733, 1977.

[3] Internet Engineering Task Force. Simple mail transfer protocol. https://tools.ietf.org/html/rfc821, 1982.

[4] Internet Engineering Task Force. Post office protocol. https://tools.ietf.org/html/rfc918, 1984.

[5] Internet Engineering Task Force. Interactive mail access protocol - version 2. https://tools.ietf.org/html/rfc1064, 1988.

[6] Malwarebytes. What is phishing? https://www.malwarebytes.com/phishing/.

[7] Google. ¿puedes detectar cuándo te están engañando? https://phishingquiz.withgoogle.com/?hl=es.

[8] ABC. El hombre que logró robar 121 millones de dólares a facebook y google con facturas falsas. https://www.abc.es/tecnologia/redes/abci-hombre-logro-robar-121-millones-dolares-facebook-y-google-facturas-falsas-201903261035_noticia.html, 2019.

[9] Profesional Review. Gmail tiene ya 1.500 millones de usuarios activos. https://www.profesionalreview.com/2018/10/28/gmail-tiene-ya-1-500-millones-de-usuarios-activos/, 2018.

[10] Lifewire. How many people use email worldwide? https://www.lifewire.com/how-many-email-users-are-there-1171213, 2019.

[11] Google. Message header. https://toolbox.googleapps.com/apps/messageheader/.

[12] Microsoft. Message header analyzer. https://mha.azurewebsites.net/.

[13] Mxtoolbox. Email header analyzer. https://mxtoolbox.com/EmailHeaders.aspx.

[14] Virus Total. https://www.virustotal.com/gui/.

[15] Virus Total. Api. https://developers.virustotal.com/reference.

[16] Metadefender. https://metadefender.opswat.com/?lang=en.

[17] Metadefender. Api. https://onlinehelp.opswat.com/mdcloud/.

[18] Have I been pwned. Eldfsfds. https://haveibeenpwned.com/.

[19] Have I been pwned. Api. https://haveibeenpwned.com/API/v3.

[20] Wikipedia. Html. https://es.wikipedia.org/wiki/HTML.

[21] Internet Engineering Task Force. Textual representation of ipv4 and ipv6 addresses. https://tools.ietf.org/html/draft-main-ipaddr-text-rep-00, 2003.

[22] Internet Engineering Task Force. Assigned numbers. https://tools.ietf.org/html/rfc790, 1981.

[23] Internet Engineering Task Force. Ip version 6 addressing architecture (2.2.text representation of addresses). https://tools.ietf.org/html/rfc4291#section-2.2, 2006.

[24] Internet Engineering Task Force. A recommendation for ipv6 address text representation. https://tools.ietf.org/html/rfc5952, 2010.

[25] Internet corporation for assigned names and numbers; icann. https://www.icann.org/es.

[26] Internet Engineering Task Force. Domain names - implementation and specification. https://tools.ietf.org/html/rfc1035, 1987.

[27] Publicsuffix. Lista de slds. https://publicsuffix.org/list/public_suffix_list.dat.

[28] IANA. Lista de tlds. http://data.iana.org/TLD/tlds-alpha-by-domain.txt.

[29] Internet Engineering Task Force. Internationalizing domain names in applications (idna). https://tools.ietf.org/html/rfc3490, 2003.

[30] Internet Engineering Task Force. Dod internet host table specification. https://tools.ietf.org/html/rfc952, 1985.

[31] Internet Engineering Task Force. Requirements for internet hosts – application and support. https://tools.ietf.org/html/rfc1123, 1989.

[32] Internet Engineering Task Force. Uniform resource locators (url). https://tools.ietf.org/html/rfc1738, 1994.

[33] Internet Engineering Task Force. Uniform resource identifier (uri): Generic syntax. https://tools.ietf.org/html/rfc3986, 2005.

[34] Internet Engineering Task Force. Application techniques for checking and transformation of names. 4. urls and uris. 4.1. uri syntax definitions and issues. https://tools.ietf.org/html/rfc3696#section-4.1, 2004.

[35] Internet Engineering Task Force. Hypertext transfer protocol (http/1.1): Message syntax and routing. 3.message format. 3.1.start line. 3.1.1.request line. https://tools.ietf.org/html/rfc7230#section-3.1.1, 2014.

[36] Microsoft. La longitud máxima de la dirección url es de 2083 caracteres en internet explorer. https://support.microsoft.com/es-es/help/208427/maximum-url-length-is-2-083-characters-in-internet-explorer, 2019.

[37] Internet Engineering Task Force. Internet message format. 3.syntax. 3.4.address specification. 3.4.1.addr-spec specification. https://tools.ietf.org/html/rfc5322#section-3.4.1, 2008.

[38] Internet Engineering Task Force. Application techniques for checking and transformation of names. 3.restrictions on email addresses. https://tools.ietf.org/html/rfc3696#section-3, 2004.