Recibido: 20 de setiembre del 2016
Aprobado: 21 de octubre del 2016

Filogenia de *malware* orientada al análisis de librerías

Juan Manuel Gutiérrez Cárdenas
Jmgutier@ulima.edu.pe
Universidad de Lima. Lima, Perú

Leopoldo Lenin Orihuela
leopoldo.lenin@gmail.com
Caltec. Lima, Perú

Resumen

En el campo de la biología computacional se emplea la filogenia con el objetivo de reconocer la semejanza existente entre diversas especies, así como el motor de evolución que ha permitido que estas muestren modificaciones con el paso del tiempo. El empleo de estas técnicas de filogenia, orientadas hacia los virus computacionales, ha demostrado ser una alternativa que permite encontrar similitudes entre diversas familias de malware. En el presente trabajo se presenta la implementación de una prueba de concepto, mediante la aplicación de una técnica utilizada en el campo de la bioinformática, como es el caso del algoritmo de Neighbor-Joining, dirigido al análisis de un grupo de muestras de virus de computadoras. La finalidad será la detección de semejanzas entre las muestras recopiladas, tomando en consideración la similitud entre las llamadas "librerías del sistema" que realizan este tipo de programas.

Palabras clave: malware / similitud de cadenas / filogenia

Abstract

In the field of computational biology, phylogeny is used to recognize the existing similarity between various species as well as the evolution engine (force) that has enabled these species to show modifications as time goes by. The use of these phylogeny techniques with a focus on computer viruses has allowed the finding of similarities between different malware families. This study presents the implementation of a proof of concept through the application of a technique used in the bioinformatics field, as is the case of the Neighbor Joining algorithm designed for the analysis of a group of computer samples. The aim will be to detect the similarity between the gathered samples, considering the similarity between the libraries of the system that uses this kind of programs.

Keywords: malware / chains similarity / phylogeny

1. Introducción

En esta investigación se tratará de exponer las semejanzas que podrían encontrarse entre distintos tipos de *software* malicioso o *malware*, de tal manera que se agrupen diversas especies de virus informáticos con una técnica ampliamente utilizada en el campo de la bio-informática, como es el caso de la filogenia. Con esta técnica podemos agrupar especies o taxas en ramas de un grafo tipo árbol, con el fin de poder observar si han existido cambios evolutivos a través del tiempo, o simplemente determinar si algunas especies comparten características similares entre sí.

La utilización de la filogenia, usada principalmente en la clasificación de *software* no es nueva. Existen diversas investigaciones como los trabajos de Goldberg L., Goldberg P., Phillips y Sorkin, (1998) con su modelo conocido como phyloDAG. Este modelo se basaba en la hipótesis de que muchas especies virales son construidas como modificaciones de otras ya existentes.

También se destaca el trabajo de Karim, Walenstein, Lakhotia y Parida (2005), que apuntaba a construir un modelo filogenético de virus basándose en los denominados "virus polimórficos". En ese modelo la solución fue construir permutaciones o intercambios en el código para así realizar una comparativa basada en los *n-grams* de los códigos alterados.

Entre otros enfoques relacionados a nuestra propuesta se encuentran los trabajos de Carrera y Erdélyi (2004) en los cuales se diseñó un *plugin* realizado en Python conjuntamente con el programa llamado IDA Pro, que es un desensamblador. En los citados trabajos se elaboró un gráfico en donde las funciones estarían representadas por los vértices y las llamadas entre estas serían los lados del grafo; brevemente se hace referencia a la probable utilización del algoritmo de Neighbor-Joining a fin de encontrar agrupamientos de taxas entre los virus (Carrera y Erdélyi, 2004).

Otra investigación similar fue la elaborada por Gheorghescu (2005) en la que se crea un flujograma o CFG (Control Flow Graph), para hacer un seguimiento del código donde los vértices son bloques de instrucciones con longitudes entre 12 y 14 *bytes*, los lados representan interacciones entre estos bloques. La medida de distancia para comparar bloques de código fue una modificación de la distancia de Levenshtein. Para finalizar este breve recuento, en la investigación propuesta por Khoo y Lió (2011) se utilizan conceptos de filogenia con el objeto de realizar ingeniería inversa en la estructura de los virus y así determinar familiaridades entre ellos. En la investigación se explica que ciertas operaciones, como son: entrada/salida, manejo de librerías dinámicas e hilos, se mantienen constantes entre diversas familias de virus. El artículo despliega diversas técnicas de biología computacional para lograr esta clasificación como son árboles filogenéticos, análisis de motifs, logos, entre otras.

Lo expuesto en el párrafo anterior es tan solo una muestra reducida de las diversas técnicas que se han desarrollado en esta área, las cuales orientan sus esfuerzos hacia una mejora de la detección de virus o *malware*, a pesar del trabajo seminal de Fred Cohen donde se estableció la indecidibilidad o imposibilidad de saber si un *software* puede ser detectado como malicioso (Filiol, 2005).

El presente trabajo de investigación está organizado de la siguiente manera: se presenta una descripción sucinta acerca de conceptos básicos sobre *malware* y similitud entre secuencias, las cuales se aplicarán para comparar semejanzas en el código de distintos tipos de *software* malicioso; se finalizará este apartado con una introducción al campo de la filogenia y la descripción de un algoritmo basado en distancias para la construcción de árboles filogenéticos. Posteriormente, se describen los métodos usados para extraer información acerca de las llamadas a librerías al sistema, hechas por códigos de *malware*; de igual manera se muestra la aplicación de un algoritmo de distancias para determinar la similitud entre estos códigos. Finalmente, se presentan los resultados y las correspondientes conclusiones.

2. Conceptos previos

2.1. Malware

El término *malware* se refiere a todo aquel tipo de *software* que es utilizado con fines de perjudicar o causar efectos dañinos a un determinado objetivo (Aycpl, 2006). Otro término empleado vendría a ser el de un *computer infection program*, que podría ser un programa simple o uno con características de replicación. Un *malware* encuentra la forma de instalarse de forma transparente en un sistema de datos, es decir, sin que el usuario tenga conocimiento de esta acción y con la potencialidad de causar un daño a la confidencialidad, disponibilidad e integridad del sistema (Filiol, 2005).

Dentro del *malware* existen diversas variantes; por ejemplo los virus, las bombas lógicas, los troyanos y los *worms* o 'gusanos'. Las diferencias entre cada una de estas consisten en la forma cómo actúan. Por ejemplo, un troyano realiza la instalación de una funcionalidad de servidor al interior de la máquina víctima y esta permitirá que los puertos de la víctima se vuelvan vulnerables (Filiol, 2005). Un virus, por el contrario, es un programa que trata de replicarse dentro de un código de tipo ejecutable (Aycock, 2006); en este caso el símil con los virus biológicos es innegable. El último tipo vendrían a ser los *worms* o 'gusanos', los cuales poseen ciertas características como los virus pero no requieren de un archivo ejecutable para subsistir y pueden ser transmitidos por medio de las redes (Aycock, 2006). En la presente investigación se utilizarán indistintamente ambos términos: *worms* o virus.

Los antivirus han sido soluciones de *software* planteadas con la finalidad de detectar o suprimir estas amenazas cuando ingresan a un sistema. Una técnica bastante utilizada en este tipo de *software* ha sido la búsqueda, a través de la utilización de una base de datos interna actualizable, de firmas o secciones de código común entre las diferentes familias de *malware* (Goldberg et al., 1998). Esta labor se dificulta más cuando los virus toman otras formas o se encuentran encriptados, como es el caso de los virus polimórficos. La utilización de técnicas antivirales basadas en heurísticas, o la de encontrar similitudes entre cadenas de virus, como en este trabajo, corresponde a otro conjunto de técnicas de prevención contra este tipo de ataques.

2.2. Similitud entre secuencias

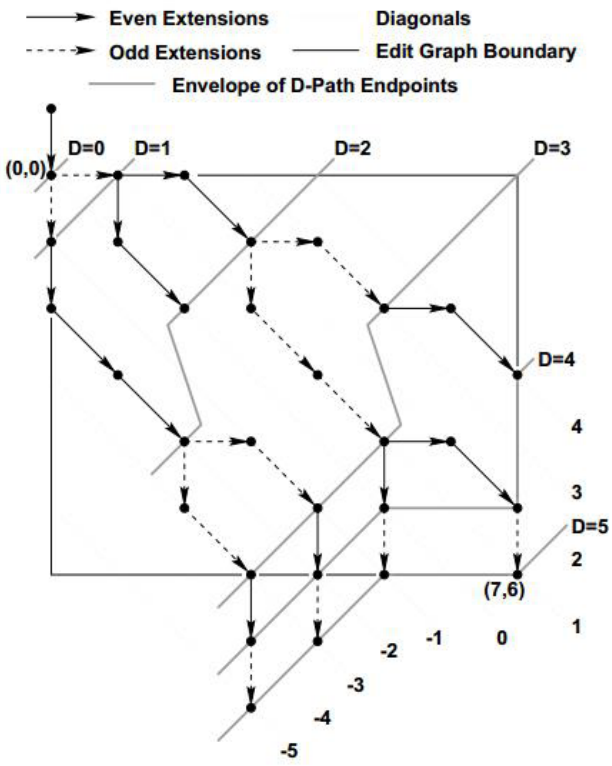
La similitud entre secuencias consiste, por ejemplo, desde el punto de vista de la biología molecular, en saber qué tan parecidas son dos secuencias o en el número de cambios que se deben de hacer en una de ellas, a fin de que una se parezca a la otra. Si se considera que las secuencias biológicas, como es el caso del ADN, son una secuencia de nucleótidos representados por los caracteres A, C, T y G, se puede extender estas técnicas de comparación hacia el hallazgo de similitudes entre secuencias basadas en cualquier carácter textual.

Una forma bastante sencilla de comparar dos secuencias es de colocar los caracteres a manera de índices en una matriz y comparar carácter por carácter. Cada vez que exista una coincidencia se marca un punto en la matriz; esta técnica es conocida como Dot Plot (Mount, 2011). Una variante de esta técnica consiste en realizar operaciones de inserción y de borrado entre las secuencias, de tal manera que se trata de observar cuántos cambios son necesarios para que una cadena sea semejante a la otra (Needleman y Wunsch, 1970). Esta técnica nos da un puntaje de alineación que determina el nivel de exactitud del alineamiento o de semejanza entre cadenas. Para este proceso se pueden utilizar técnicas basadas en programación dinámica.

En algunas ocasiones se requiere encontrar si entre un par de cadenas de ADN se encuentra una subsecuencia que esté contenida en otra. Cuando la cadena que se pretende buscar debe de ser la de mayor longitud, nuestro problema se reduce a hallar el denominado LCS (Longest Common Subsequence).

La tarea de hallar esta cadena común tiene un problema dual: encontrar un LCS utilizando una matriz como en la técnica de Needleman y Wunsch (1970), pero que tenga el menor número de operaciones de inserción y borrado, o lo que se reduce a hallar un SES (Shortest Edit Script). En resumen, se busca encontrar el mayor número de lados diagonales con el menor número de lados no diagonales; donde un lado diagonal representa una coincidencia entre caracteres y uno no-diagonal representa la situación opuesta (Myers, 1986). Una explicación de esta técnica se representa en la figura 1.

Figura 1. Caminos más largos



Desde el punto (0,0) hasta el punto (7,6) se calcula la cantidad de operaciones de edición, inserciones y borrados (edit), necesarias para que una cadena colocada en la parte izquierda sea parecida a una cadena colocada en la parte superior.

Fuente: Myers (1986)

Se forman distintos caminos entre ambos puntos, donde una línea diagonal representa una coincidencia entre caracteres. Este problema puede ser reducido a cómo encontrar el camino más corto. Como se observa, existen diversos caminos que representan operaciones de edición realizadas para que una cadena se asemeje a la otra.

2.3. Filogenia y método del Neighbor-Joining

La filogenia es el estudio que nos permite determinar los probables ancestros evolutivos de las especies. Su símil vendría a ser el árbol genealógico utilizado para sistematizar nuestros vínculos familiares. La forma de representar esta secuencia de ancestros y sucesores es usando una estructura de árbol como las usadas en teoría de grafos. En este tipo de representación las hojas representan a las especies, mientras que los nodos

internos son los antecesores de estas secuencias; donde las distancias entre estas representarían el llamado "reloj biológico" o aquel factor de mutación que se mantiene constante entre las ramas de un árbol filogenético (Haubold y Wiehe, 2006). Existen diversas formas de obtener esta estructura de árbol y la que se usará en este artículo es una basada en distancias denominada el algoritmo de Neighbor-Joining.

El método de Neighbor-Joining (Saitou y Nei, 1987) consiste en unir dos especies o taxas en un árbol que no posee raíz. Para este caso se calcula primero una matriz de distancias; esta matriz de distancias tiene la característica de ser una matriz simétrica en la cual la diagonal está formada por valores iguales a cero. El algoritmo base sería el siguiente:

Algoritmo Neighbour-Joining (M[ ][ ])

{M es una matriz simétrica de distancias entre secuencias de *m* filas y *n* columnas, *T* es un árbol vacío}

1. Repetir mientras existan elementos sin unir en M[ ][ ]
   2. Buscar mínima distancia en M[ ][ ]
   3. $taxa_{s1,s2} \leftarrow taxa_{s1} \cup taxa_{s2}$ //La taxa resultante formaría la pseudo-raíz de ambas hojas
   4. $T \leftarrow T \cup taxa_{s1,s2}$
   5. Calcular las distancias entre las taxas sobrantes usando la fórmula:

      $dist[taxa_{i}, taxa_{s1,s2}] = (dist[taxa_{i}, taxa_{s1}] + dist[taxa_{i}, taxa_{s2}])/2$
6. fin de repetir

El algoritmo descrito va uniendo aquellos valores de la matriz de similitud que tengan el menor valor entre ellos. Este proceso se repite al eliminar aquellos valores que han sido unidos y que ahora forman parte del árbol, recalculando al mismo tiempo la distancia entre los antiguos valores de la matriz con los nuevos formados por la unión de distintas taxas. Este método es bastante directo para encontrar árboles filogenéticos y será el que se probará en la presente propuesta.

3. Metodología

El primer paso consiste en la selección de un conjunto de muestras virales de una misma familia. Estas muestras serán comparadas posteriormente con otros virus de similares características pero pertenecientes a otros grupos; de esta forma se puede apreciar si existen características que los hace comunes a ambos, lo cual permitiría agruparlos en una misma rama.

Las muestras virales serán recopiladas de Offensive Malware[1]. Este repositorio contiene diversas muestras de *malware* catalogadas por familias de virus y permite la descarga de estos archivos para su estudio respectivo. En este acápite es conveniente recordar las medidas de seguridad al trabajar con este tipo de archivos, como por ejemplo el uso de máquinas virtuales, de preferencia Linux como sistema operativo, entre otras.

A continuación se describe el proceso seguido con el propósito de extraer de las librerías información a ser comparada, en nuestro caso, llamadas al sistema entre las diversas formas virales. Para esto se tiene que considerar que todo código que quiera usar los recursos de *hardware* (procesador y memoria) debe ser planificado para ser ejecutado. Hay programas que son residentes y corren automáticamente al cargar el sistema operativo, como los servicios; por el contrario, existen otros que se ejecutan dinámicamente y que son gestionados por el usuario. Estos últimos son ejecutados de forma manual mediante una interfaz o *shell*.

Microsoft usa la extensión de archivos ".exe", los cuales poseen una cabecera o firma en particular. Esta cabecera actúa como una firma que indica al planificador la labor a ejecutar con este tipo de archivos. En caso de ser un archivo del tipo ".com" el procesador ejecutará el código desde el primer *byte* sin buscar las cabeceras. Un virus puede tomar, indistintamente, cualquiera de ambas formas. Se debe tener en cuenta que generalmente un archivo que contiene estos códigos a ejecutarse se encuentra compuesto de las siguientes partes:

- Código o texto
- Pila
- Datos

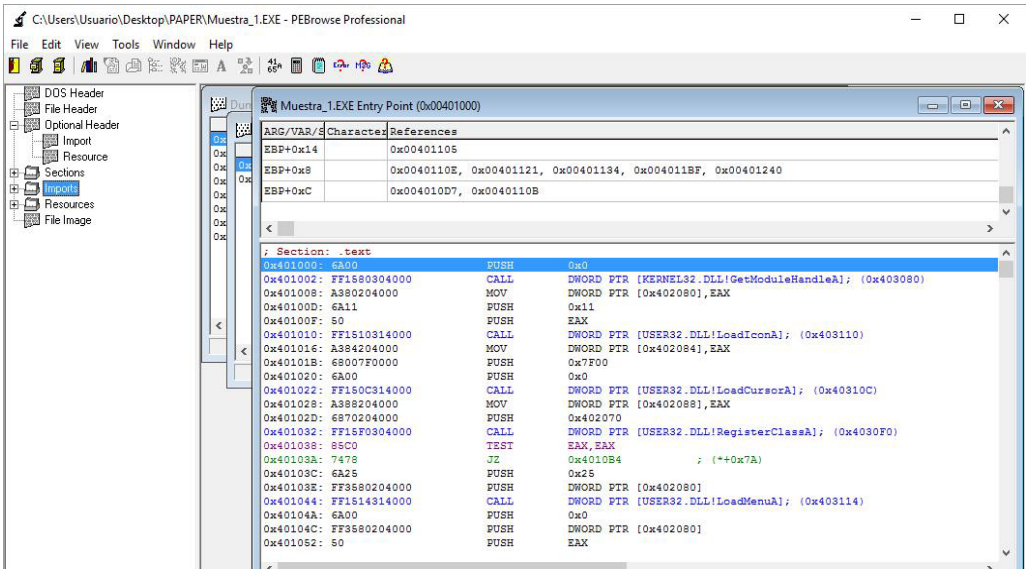
El código es marcado en el archivo para indicarle al procesador desde dónde debe empezar la ejecución del programa. Estos códigos se encuentran en valores hexadecimales. Para que un programa pueda ejecutarse requiere llamar a este código, que se encuentra

1. http://offensivecomputing.net/

en el segmento de texto. Del mismo modo, necesita datos para que el código trabaje, los cuales se hallan en el denominado segmento de datos. Los datos que van produciéndose dinámicamente pueden ser almacenados en un segmento que cuando se carga en memoria RAM recibe los valores y hace las veces de memoria temporal.

Ahora bien, en algunas circunstancias se necesita investigar sobre la estructura interna de un programa; para realizar un análisis forense de datos o, en nuestro caso, para averiguar sobre la estructura interna de un virus. Para este proceso se utilizaron dos herramientas: Winhex (Winhex, 2016), potente editor para análisis forense de datos y PEBrowser (PEBrowse 64 Professional, 2016), que servirá como un desensamblador para seleccionar e identificar los inicios de segmento de pila, datos, código como se puede apreciar en la figura 2.

Figura 2. Desensamblado de un programa

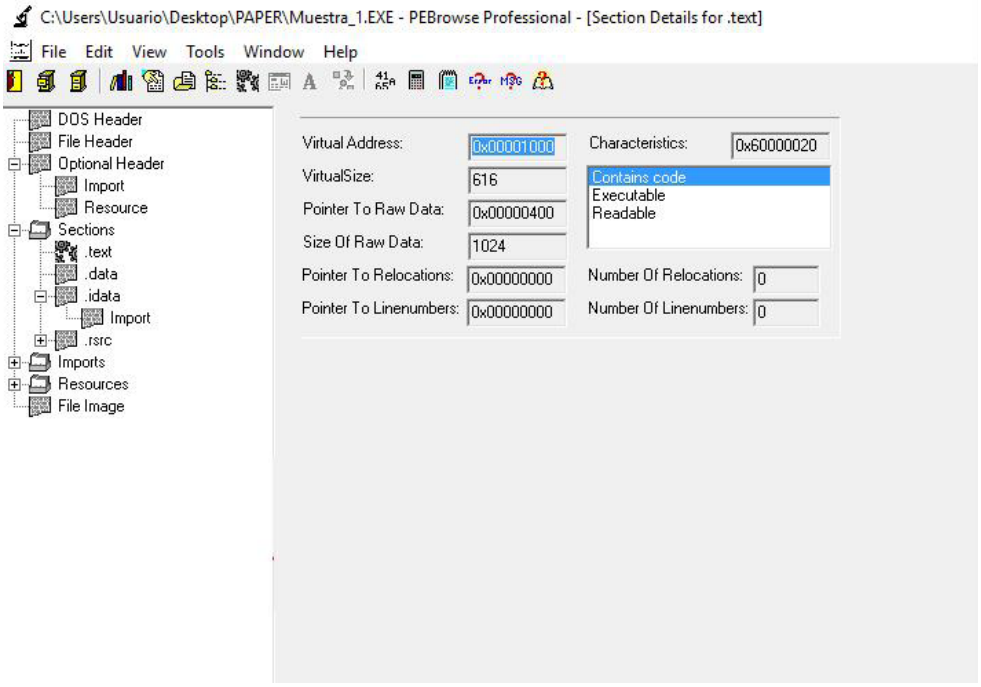


Se muestran los bloques de código conjuntamente con sus direcciones de memoria haciendo uso de (PEBrowse 64 Professional, 2016).

Elaboración propia

La herramienta de la figura 2 trabaja a 32 o 64 *bits*, identificando los segmentos de cada tipo con el objeto de poder desensamblarlos y hacer un proceso de ingeniería inversa. Es preciso comentar que este proceso no se puede hacer con todas las muestras, ya que algunas no poseen la misma estructura. En estos casos se presume la existencia de una encriptación de código; dando como consecuencia que la estructura del mismo, pilas y datos no sean tan aparentes al intentar diferenciar sus segmentos. Esta situación en particular se puede apreciar en la figura 3.

Figura 3. Caso de encriptación de código

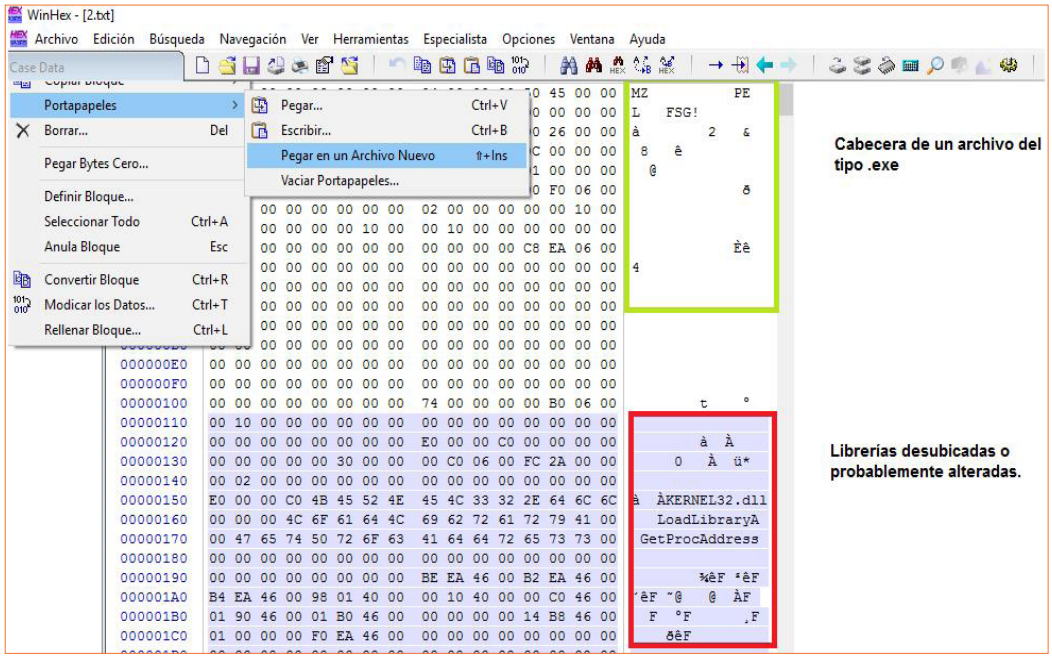


Haciendo uso de un desensamblador se aprecian las direcciones de los distintos segmentos de datos de un programa determinado.

Elaboración propia

Como se observa en la figura 3, esta herramienta nos permite visualizar los distintos segmentos con sus respectivas firmas cuando la muestra de código no se encuentra encriptada; esto mismo sucede cuando se desea encontrar la información correspondiente a las librerías para nuestra investigación. Cuando la muestra no está encriptada se podrá diferenciar los segmentos; sus respectivas firmas del código o de las librerías y se podrán visualizar como dato. Un caso en particular en el cual no se ha respetado el orden de las diversas secciones de un código se expone en la figura 4.

Figura 4. Segmento de programa desensamblado



Se aprecia que el orden ha sido alterado, particularmente en lo que respecta a la ubicación de las librerías.

Elaboración propia

Un programa que cuente con una estructura normal se encontrará bien estructurado, es decir, las librerías y los datos van después del código. En formas virales este orden no se aprecia claramente, pues algunas librerías pueden estar desubicadas (figura 4), o incluso en algunos casos pueden no estar, ser inexistentes. Todos estos indicios advertirían la presencia de un código que podría ser un *malware*. Para los propósitos de la investigación se centrará en la extracción de la data de estas librerías, de tal manera que se pueda llevar a cabo un proceso de comparación de las similitudes de estas en diversas familias de virus.

Una vez obtenidas las librerías de cada de una de estas muestras, se procedió a utilizar el HexEdit (HexEdit, 2012), editor binario, para convertir dicha información a un formato de texto y de esa manera compararlas haciendo uso de una herramienta de similitud basada en la librería de PHP denominada Similar_text[2]. Esta herramienta está basada en el trabajo de investigación de Myers (1986).

2. Similar_text puede encontrarse en la página web: https://www.tools4noobs.com/online_tools/string_similarity/

Los porcentajes de similitud hallados nos darán una medida de similitud entre cada par de secuencias de *malware*. En este punto se plantea como hipótesis que dicha herramienta utiliza el LCS, a fin de observar que tantas operaciones de edición o *indel (insertion-deletion*), deberían realizar estas secuencias para poder coincidir con el LCS; hipótesis que se reafirma al revisar el trabajo de Myers (1986) en el cual se basa la herramienta usada. A partir de la matriz de similitud hallada se aplica el algoritmo del Neighbor-Joining (Saitou y Nei, 1987) para obtener un árbol filogenético.

4. Resultados

Se analizará primeramente la familia de virus conocida como Mimail. Este virus es un troyano esparcido a través de mensajes por correo electrónico (Mimail-The Virus Encyclopedia, s.f.). A este virus, así como a los demás que se presentarán líneas abajo, se les efectuó primero un proceso de desensamblaje, para ubicar las llamadas a las librerías del sistema y luego realizar el proceso de comparación de cadenas especificado en el punto anterior.

Con respecto al Mimail se descargaron nueve variantes de este virus:

Email-Worm.Win32.Mimail.o

Email-Worm.Win32.Mimail.s

Email-Worm.Win32.Mimail.u

Email-Worm.Win32.Mimail.j

Email-Worm.Win32.Mimail.f

Email-Worm.Win32.Mimail.k

Email-Worm.Win32.Mimail.a

Email-Worm.Win32.Mimail.e

Email-Worm.Win32.Mimail.l

Entre estas variantes se realizó un análisis de la similitud de cadenas presentes en su codificación hexadecimal. Los resultados obtenidos fueron los siguientes:

Tabla 1. ***Distancias obtenidas entre diversos miembros de la familia Mimail***

Mimail variantes									
Email-Worm. Win32.Mimail.o	0	0,3005	0,457	0,4537	0,4169	0,3561	0,3997	0,4065	0,2271
Email-Worm. Win32.Mimail.s	0,3005	0	0,4512	0,4496	0,1761	0,0934	0,4951	0,468	0,7709
Email-Worm. Win32.Mimail.u	0,457	0,4512	0	0,9343	0,3998	0,1992	0,8673	0,8943	0,3992

(Continúa)

(Continuación)

Email-Worm. Win32.Mimail.j	0,4537	0,4496	0,9343	0	0,349	0,2004	0,9265	0,9056	0,3992
Email-Worm. Win32.Mimail.f	0,4169	0,1761	0,3998	0,349	0	0,2463	0,3997	0,3996	0,1484
Email-Worm. Win32.Mimail.k	0,3561	0,0934	0,1992	0,2004	0,2463	0	0,1565	0,1752	0,075
Email-Worm. Win32.Mimail.a	0,3997	0,4951	0,8673	0,9265	0,3997	0,1565	0	0,9455	0,3833
Email-Worm. Win32.Mimail.e	0,4065	0,468	0,8943	0,9056	0,3996	0,1752	0,9455	0	0,392
Email-Worm. Win32.Mimail.l	0,2271	0,7709	0,3992	0,3992	0,1484	0,075	0,3833	0,392	0

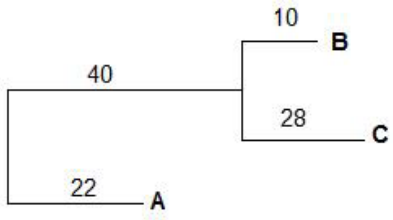
Elaboración propia

Los datos de la tabla 1 serán utilizados en la construcción del árbol filogenético de esta especie de *malware*; para ello se utilizó el algoritmo del Neighbor-Joining de Saitou y Nei (Saitou y Nei, 1987). La data obtenida por este algoritmo y expresada en el formato de Newick es la siguiente:

(Email-Worm.Win32.Mimail.j:3.5883,((Email-Worm.Win32.Mimail.k:0.0000,Email-Worm.Win32.Mimail.a:3.5390):0.0000,(Email-Worm.Win32.Mimail.l:0.7894,(Email-Worm.Win32.Mimail.o:0.9112,Email-Worm.Win32.Mimail.e:3.1530):0.2741):0.4874):0.1018,(E-mail-Worm.Win32.Mimail.f:0.4359,(Email-Worm.Win32.Mimail.s:1.2579,Email-Worm.Win32.Mimail.u:3.2542):0.1873):0.4139).

El formato de Newick (The Newick tree format, 2016) permite hacer la representación de un árbol haciendo uso de una expresión parentizada, como el caso mostrado en la figura 5.

Figura 5. Representación en forma de árbol de tres especies relacionadas entre sí.



Los valores de los lados representan las distancias de la especie hacia la probable raíz de una rama arbitraria. En la mayoría de casos, estos árboles presentan una estructura de árbol sin raíz.

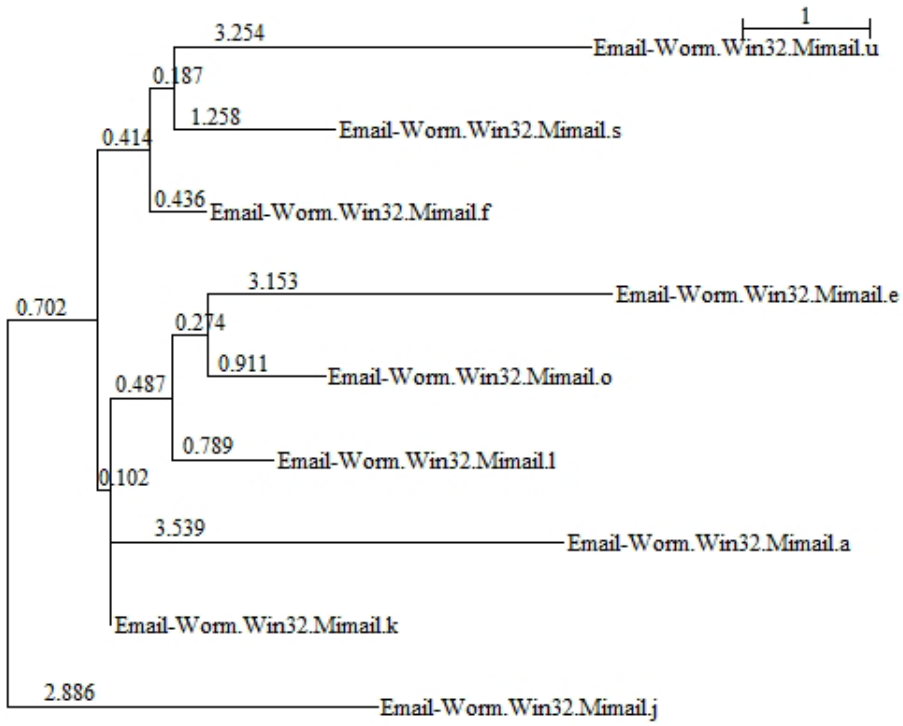
Elaboración propia

El árbol de la figura 5 puede ser transformado a la representación Newick, al colocar el nombre de la especie seguida de la distancia hacia la subrama superior. Para el caso anterior su representación sería:

(A:22,(C:28,B:10):40).

Una vez lograda la representación en el formato de Newick, se procede a usar la herramienta conocida como NJPlot (NJPlot, 2015) para graficar el árbol obtenido. En el caso de las especies del virus Mimail, el árbol filogenético obtenido por la técnica descrita anteriormente sería el siguiente:

Figura 6. Árbol filogenético obtenido de la muestra del virus Mimail.



Se observa la semejanza existente entre las diversas especies, lo que indicaría una similitud en las llamadas a librerías.

Elaboración propia

En la figura anterior se aprecian algunos aspectos interesantes en la agrupación como, por ejemplo, que la muestra viral del Win32.Mimail.k se encuentra dentro de la misma rama del Win32.Mimail.e. Esto último es justificable debido a que el Mimail.k es una muestra

variada del Mimail.e (Podrezov, 2003). Un análisis similar se podría emplear con las demás secuencias.

A continuación se compara la familia del *worm* Mimail con otro virus que también se propaga a través de los mensajes de correo electrónico: el *worm* MyDoom. En la siguiente tabla se muestra parte de la similitud entre ambas familias:

Tabla 2. ***Matriz de distancias de similitud entre virus de la familia Mimail y Mydoom***

Email-Worm.Win32.Mimail.o	0	0,3005	0,457	0,4537	0,4169	0,3561	0,3997	0,4065
Email-Worm.Win32.Mimail.s	0,3005	0	0,4512	0,4496	0,1761	0,0934	0,4951	0,468
Email-Worm.Win32.Mimail.u	0,457	0,4512	0	0,9343	0,3998	0,1992	0,8673	0,8943
Email-Worm.Win32.Mimail.j	0,4537	0,4496	0,9343	0	0,349	0,2004	0,9265	0,9056
Email-Worm.Win32.Mimail.f	0,4169	0,1761	0,3998	0,349	0	0,2463	0,3997	0,3996
Email-Worm.Win32.Mimail.k	0,3561	0,0934	0,1992	0,2004	0,2463	0	0,1565	0,1752
Email-Worm.Win32.Mimail.a	0,3997	0,4951	0,8673	0,9265	0,3997	0,1565	0	0,9455
Email-Worm.Win32.Mimail.e	0,4065	0,468	0,8943	0,9056	0,3996	0,1752	0,9455	0
Email-Worm.Win32.Mimail.l	0,2271	0,7709	0,3992	0,3992	0,1484	0,075	0,3833	0,392
Email-Worm.Win32. Mydoom.l	0,4116	0,2173	0,3993	0,3976	0,447	0,3999	0,3823	0,3839
Email-Worm.Win32. Mydoom.o	0,3996	0,5428	0,6889	0,7096	0,3831	0,1405	0,7721	0,737
Email-Worm.Win32. Mydoom.t	0,1844	0,6785	0,3989	0,3994	0,1808	0,0816	0,5057	0,5131
Email-Worm.Win32. Mydoom.q	0,2898	0,4717	0,2754	0,2858	0,1773	0,0446	0,399	0,3994
Email-Worm.Win32. Mydoom.b	0,2555	0,5522	0,3991	0,3986	0,1992	0,0681	0,4085	0,4142

Nota: Se ha colocado solamente una parte de dicha información. Esta medida de similitud será procesada con la finalidad de obtener el árbol filogenético entre ambas variantes.

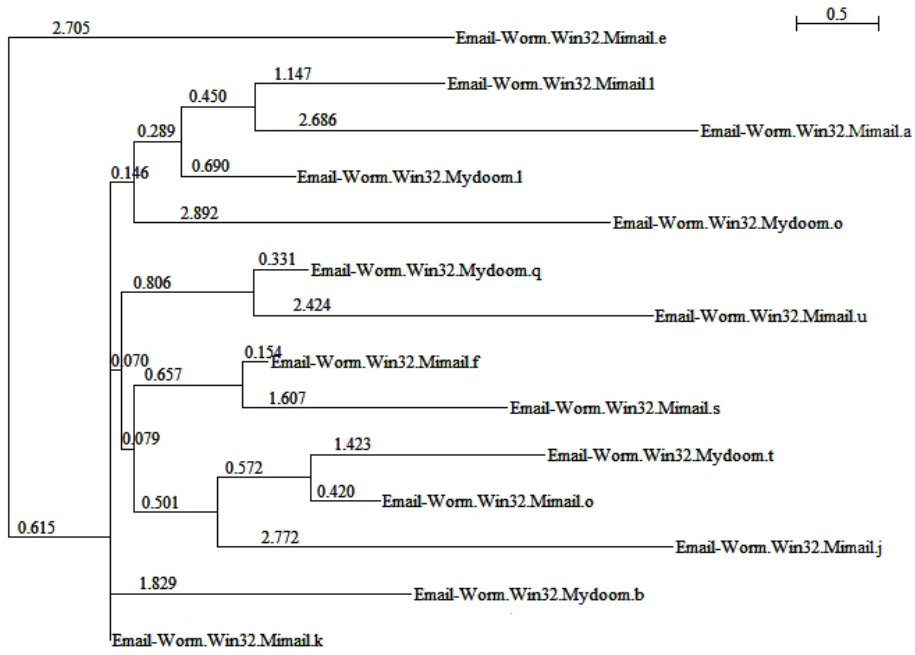
Elaboración propia

Al comparar este conjunto de secuencias mediante el algoritmo del Neighbour-Joining se obtiene la siguiente representación en el formato de Newick:

((((Email-Worm.Win32.Mimail.j:2.7723,(Email-Worm.Win32.Mimail.o:0.4198,Email-Worm.Win32.Mydoom.t:1.4229):0.5719):0.5014,(Email-Worm.Win32.Mimail.s:1.6075,Email-Worm.Win32.Mimail.f:0.1540):0.6569):0.0786,(Email-Worm.Win32.Mimail.u:2.4236,Email-Worm.Win32.Mydoom.q:0.3306):0.8057):0.0696,(Email-Worm.Win32.Mydoom.o:2.8919,(Email-Worm.Win32.Mydoom.l:0.6898,(Email-Worm.Win32.Mimail.a:2.6855,Email-Worm.Win32.Mimail.l:1.1473):0.4496):0.2891):0.1462,(Email-Worm.Win32.Mydoom.b:1.8288,(Email-Worm.Win32.Mimail.k:0.0000,Email-Worm.Win32.Mimail.e:3.3203):0.0000):0.0000);

Información que nos dará la siguiente representación de árbol filogenético:

Figura 7. Representación de la filogenia de las especies virales conocidas como Mimail y Mydoom



Elaboración propia

Una información resaltante que se observa en la figura 7 es cómo algunas especies de la familia Mimail presentan un alto grado de similitud con otras de la familia Mydoom. En este caso, dicha característica no sería de extrañar puesto que ambas muestras virales utilizan casi los mismos medios de transmisión y, por tanto, es de suponer que tendrán similares llamadas también a las librerías del sistema.

Para el caso siguiente se compara ambas familias, Mimail y Mydoom, con una muestra del virus conocido como Blaster. Este virus intenta explorar una vulnerabilidad en las llamadas a los procedimientos remotos (Remote Procedure Call, RPC) de los sistemas basados en Windows usando el puerto 135 mediante TCP (Techopedia, s.f.) Los resultados obtenidos de esta comparativa se expresan a continuación.

Tabla 3. ***Matriz de distancias entre los virus de tipo gusano de las familias Mimail, MyDoom y Blaster***

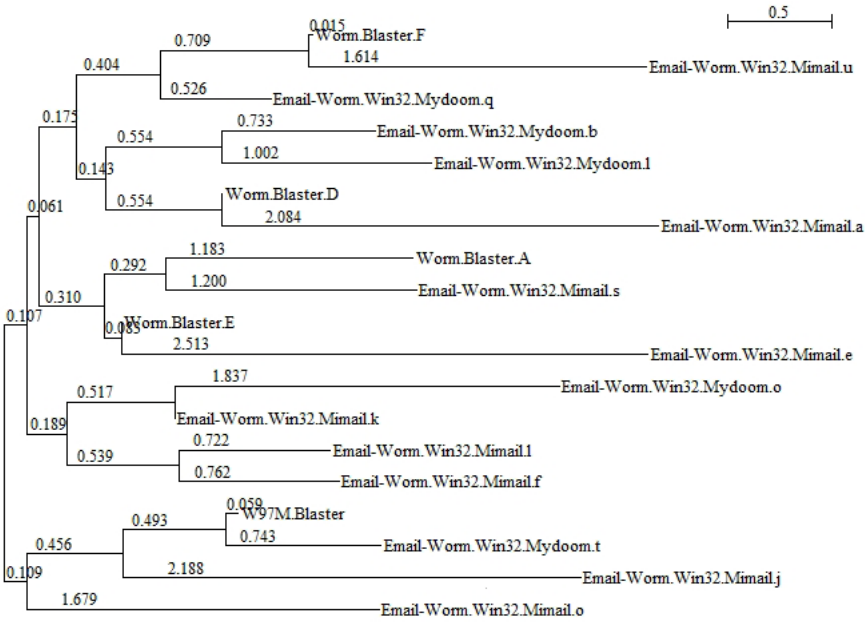
Email-Worm.Win32.Mimail.o	0	0,3005	0,457	0,4537	0,4169	0,3561	0,3997	0,4065
Email-Worm.Win32.Mimail.s	0,3005	0	0,4512	0,4496	0,1761	0,0934	0,4951	0,468
Email-Worm.Win32.Mimail.u	0,457	0,4512	0	0,9343	0,3998	0,1992	0,8673	0,8943
Email-Worm.Win32.Mimail.j	0,4537	0,4496	0,9343	0	0,349	0,2004	0,9265	0,9056
Email-Worm.Win32.Mimail.f	0,4169	0,1761	0,3998	0,349	0	0,2463	0,3997	0,3996
Email-Worm.Win32.Mimail.k	0,3561	0,0934	0,1992	0,2004	0,2463	0	0,1565	0,1752
Email-Worm.Win32.Mimail.a	0,3997	0,4951	0,8673	0,9265	0,3997	0,1565	0	0,9455
Email-Worm.Win32.Mimail.e	0,4065	0,468	0,8943	0,9056	0,3996	0,1752	0,9455	0
Email-Worm.Win32.Mimail.l	0,2271	0,7709	0,3992	0,3992	0,1484	0,075	0,3833	0,392
Email-Worm.Win32. Mydoom.l	0,4116	0,2173	0,3993	0,3976	0,447	0,3999	0,3823	0,3839
Email-Worm.Win32. Mydoom.o	0,3996	0,5428	0,6889	0,7096	0,3831	0,1405	0,7721	0,737
Email-Worm.Win32. Mydoom.t	0,1844	0,6785	0,3989	0,3994	0,1808	0,0816	0,5057	0,5131
Email-Worm.Win32. Mydoom.q	0,2898	0,4717	0,2754	0,2858	0,1773	0,0446	0,399	0,3994
Email-Worm.Win32. Mydoom.b	0,2555	0,5522	0,3991	0,3986	0,1992	0,0681	0,4085	0,4142
Worm.Blaster.E	0,2703	0,0619	0,1458	0,1475	0,1982	0,4533	0,1341	0,1334
W97M.Blaster	0,3999	0,0773	0,1629	0,2169	0,3636	0,3999	0,1923	0,1987
Worm.Blaster.A	0,5184	0,2383	0,3998	0,3998	0,4871	0,3872	0,3274	0,3998
Worm.Blaster.D	0,3704	0,103	0,1904	0,1843	0,3999	0,4445	0,1376	0,1785
Worm.Blaster.F	0,3999	0,0773	0,1629	0,2169	0,3636	0,3999	0,1923	0,1987

Elaboración propia

Las distancias obtenidas que se observan en la tabla 3, al ser procesadas haciendo uso del algoritmo de Neighbor-Joining, nos dieron el siguiente árbol filogenético:

(((Email-Worm.Win32.Mimail.o:1.6792,(Email-Worm.Win32.Mimail.j:2.1881,(Email-Worm.Win32.Mydoom.t:0.7434,W97M.Blaster:0.0587):0.4926):0.4555):0.2167,((Email-Worm.Win32.Mimail.f:0.7619,Email-Worm.Win32.Mimail.l:0.7223):0.5386,(Email-Worm.Win32.Mimail.k:0.0000,Email-Worm.Win32.Mydoom.o:1.8367):0.5169):0.1892):0.0608,((Email-Worm.Win32.Mimail.e:2.5130,Worm.Blaster.E:0.0000):0.0830,(Email-Worm.Win32.Mimail.s:1.2004,Worm.Blaster.A:1.1826):0.2918):0.3096,(((Email-Worm.Win32.Mimail.a:2.0840,Worm.Blaster.D:0.0000):0.5540,(Email-Worm.Win32.Mydoom.l:1.0023,Email-Worm.Win32.Mydoom.b:0.7326):0.5536):0.1432,(Email-Worm.Win32.Mydoom.q:0.5259,(Email-Worm.Win32.Mimail.u:1.6140,Worm.Blaster.F:0.0149):0.7086):0.4042):0.1747);

Figura 8. Árbol filogenético de la familia de gusanos MiMail, MyDoom y Blaster



Se observan características similares entre algunas muestras de estas familias, a pesar de pertenecer a grupos virales distintos.

Elaboración propia

En el árbol de la figura 8 se puede apreciar el agrupamiento de los virus de la familia Mimail y Mydoom en sus respectivas ramas. El agrupamiento de muestras virales pertenecientes a la misma familia y agrupadas en una misma rama, por ejemplo el caso del Mimail.l y del Mimail.f, es porque poseen casi las mismas características en las partes de codificación y llamadas a librerías.

En otros casos se observa que existen agrupamientos no tan explícitos, como el virus de la familia Blaster.f que utiliza el puerto TCP 135 (Symantec, 2007a) y se aloja en la misma rama que algunas variantes del Mydoom, como por ejemplo el Mydoom.q, que también tiene la capacidad de manipulación de los puertos TCP en los rangos del 3127 al 3198 (Symantec, 2007b).

De la misma manera, otros tipos de asociaciones entre familias de virus serían factibles de ser analizadas. Esto podría permitir saber qué características similares, tanto en codificación como en llamadas al sistema, se encuentran entre unas y otras. El motivo fundamental sería el poder reconocer nuevas especies víricas, las que podrían ser comparadas y clasificadas a fin de determinar si un programa es en realidad un *malware* o algún código de tipo benigno.

5. Conclusiones

La filogenia aplicable a la comparativa de muestras virales ha sido una técnica usada a fin de poder determinar semejanzas entre diversas familias de *malware*. Las técnicas de preprocesamiento de estas cadenas también son diversas y se basan mayormente en la construcción de matrices de similitud para luego aplicar algoritmos de construcción de árboles filogenéticos.

En nuestra propuesta se ha implementado una comparativa de *malware* basándonos en la extracción de las llamadas a las librerías que se han podido encontrar al desensamblar estas muestras. Posteriormente, la aplicación de un algoritmo de similitud entre cadenas, conjuntamente con un algoritmo de Neighbour-Joining, ha permitido encontrar semejanzas interesantes entre las familias de virus analizadas.

Se recomendaría para futuros estudios una mayor cantidad de muestras y observar el caso en el cual las librerías se encuentran encriptadas. Del mismo modo, sería interesante la aplicabilidad de estas técnicas en el campo de la construcción de *software* antivirus.

Referencias

Aycock, J. (2006). *Computer Viruses and Malware*. Nueva York: Springer.

Carrera, E., Erdélyi, G. (2004). Digital Genome Mapping-Advanced Binary Malware Analysis. *Virus Bulletin Conference, (pp.*187-197).

Filiol, E. (2005). *Computer Viruses: From Theory to Applications.* París: Springer-Verlag.

Gheorghescu, M. (2005). An Automated Virus Classification system. *Virus Bulletin Conference, (pp.*294-300).

Goldberg, L., Goldberg, P., Phillips, C., y Sorkin, G. (1998). Constructing Computer Virus Phylogenies. *Journal of Algorithms, 26*(1), 188-208.

Haubold, B., y Wiehe, T. (2006). *Introduction to Computational Biology: An Evolutionary Approach*. Basel, Suiza: Birkhäuser Verlag.

HexEdit (Versión 4.0) [Software] (2012). Recuperado de http://www.hexedit.com/

Karim, E., Walenstein, A., Lakhotia, A, Parida, L. (2005). Malware Phylogeny Generation Using Permutations of Code. *European Research Journal of Computer Virology, 1*(1), 13-23.

Khoo, W. y Lió, P. (2011). Unity in Diversity: Phylogenetic-Inspired Techniques for Reverse Engineering and Detection of Malware Families. *2011 First SysSec Workshop*, 3-10. IEEE Xplore. DOI: 10.1109/SysSec.2011.24

Mimail.(s.f.). En *The Virus Encyclopedia.* Recuperado de http://virus.wikidot.com/mimail

Mount, D. (2001). *Bioinformatics: Sequence and Genome Analysis*. Nueva York: Cold Spring Harbor Laboratory Press.

Myers, E. (1986). *An O(ND) Difference Algorithm and its Variations, Algorithmica, 1,* 251-266.

Needleman, S. y Wunsch, C. (1970). A General Method Applicable to the Search for Similarities in the Amino Acid Sequence of Two Proteins. *Journal of Molecular Biology, 48*(3), 443-453.

NJPlot (Versión 2.3) [Software] (2015). Recuperado de http://njplot.software. informer.com/download/

PEBrowse64 Professional [Software] (2016). Recuperado de: http://www.smidgeonsoft.prohosting.com/pebrowse-pro-file-viewer.html

Podrezov, A. (2003). Mimail.K. Threat Description. F-Secure. Recuperado de https://www.f-secure.com/v-descs/mimail_k.shtml

Saitou, N. y Nei, M. (1987). The Neighbor-Joining Method: a New Method for Reconstructing Phylogenetic Trees. *Molecular Biology and Evolution, 4*(4), 406-425.

Symantec (2007a). W32.Blaster.F.Worm. Recuperado de https://www.symantec.com/security_response/writeup.jsp?docid=2003-090105-2513-99

Symantec (2007b). W32.Mydoom.A@mm. Recuperado de https://www.symantec.com / security_response/writeup.jsp?docid=2004-012612-5422-99&tabid=2

What is the Blaster Worm? (s.f.). En *Techopedia*. Recuperado de https://www.techopedia.com/definition/27295/blaster-worm

The Newick tree format. (s.f.). En *Phylip*. Recuperado de http://evolution.genetics.washington.edu/phylip/newicktree.html

WinHex (2016) [Software]. Recuperado de https://www.x-ways.net/winhex/