Instituto Tecnológico y de Estudios Superiores de Monterrey
Análisis y diseño de algoritmos avanzados
TC2038
Grupo 604
Reflexión de trabajo
Andrea Catalina Fernández Mena
A01197705
Catedrático:
Dr. Eduardo Arturo Rodríguez Tello

Dentro de este proyecto en particular se solicitó que por medio de la lectura de 2 archivo de transiciones de código se busque algún tipo de código malicioso, con el fin de comparar si el código dentro de dichos dos archivos es malicioso se hizo una comparación con otros tres archivos que contaban con información de posibles códigos maliciosos. Esta búsqueda se logró por medio de la comparación caracter por caracter así como la lectura de dichos archivos como si se tratara de la comparación de dos vectores y los caracteres, así como ciertos patrones de palíndromo, como si cada letra del string se tratara de un elemento separado desde la vista lógica y programabilística.

Esta primera actividad integradora resultó compleja de descifrar al inicio, ya que desde mi perspectiva, me era complicado identificar cuál método e implementación de algoritmos serían implementados para lograr detectar el código malicioso dentro de la transmisiones de código. No obstante, una vez que logré entender un poco más a fondo la lógica solicitada para detectar el malware logré comprender cómo implementar algunos de los algoritmos todo a base del análisis de strings y substrings así como comparación de caracteres con el fin de identificar que ciertos patrones como el tener características palindrométricas era indicio de que se trataba de un tipo de código malicioso.

Un factor sumamente importante que se tomó a consideración al momento de desarrollar el código fue el uso de estructuras algorítmicas eficientes, esto con el fin que la naturaleza de las complejidades les permita obtener tiempos de respuesta menores, una forma utilizada como métodos y técnica durante el programa fue el desarrollo de funciones anidadas para cada parte de la actividad por ejemplo:
En la **parte 1** cuyo principal objetivo es reconocer patrones se implementaron dos algoritmos de órden **O(n)** (uno de tipo Knuth-Morris-Pratt y el otro identificando el longest proper prefix para la lectura de caracteres), así como un algoritmo de tipo **O(n*m)** con el fin de mostrar el productor de ambas comparaciones de reconocimiento.

Después, en la **parte dos** de igual forma se implementaron dos algoritmos de órden **O(n)** ambos con el fin de revisar si dichas similitudes encontradas con palíndromos y en caso de serlas identificarlas como virus, uno de los algoritmos usando el método de manacher y el otro utilizando Palindrome Application.

Por último, en la **parte 3** se implementa un algoritmo de tipo longest common substring con uan complejidad de **O(n*m)** y de la mano se implementa un algoritmo de complejidad **O(1)** el cuál nos ayuda con el output del otro algoritmo antes mencionado para la parte 3.

Por último, considerando el uso de algoritmos de las complejidades antes mencionadas es posible definir la complejidad de nuestro programa como 3O(n) + 2O(n*m) + O(1), dejando como resultado aquella complejidad más dominante la cuál en este caso sería **O(n)**, por lo que podemos decir que es

un programa en su mayoría de complejidad de tiempo linear y que es característico debido a su naturaleza propia de implementación donde la rapidez del algoritmo el proporcional al representado como el número de iteraciones a alcanzar.

Por último, considero que fue un ejercicio sumamente enriquecedor, y que me ayudó a comprender de mejor manera la lógica detrás de los algoritmos así como a buscar formas de facilitar procesos de búsqueda o detección de nuestro caso. Así también, a partir de los algoritmos y lectura implementados actividad integradora me ha permitido idear que es posible hacer un productor más tangible por medio múltiples procesos de la vida diaria enfocados a la ciberseguridad, como aquellas funciones de análisis que un software de antivirus cada que se ingresa desde el buscador a un sitio desconocido o bien cuándo se descarga un archivo, siempre y cuándo las parametrizaciones aumentan dependiendo del tipo de objetivo o virus que se solicita encontrar.

Fuentes bibliográficas para desarrollo de algoritmos:

Laakmann Mcdowell. G. (2015). *Crack in the code interview* . Palo Alto, CA, USA: Published by CareerCup.