Instituto Tecnológico y de Estudios Superiores de Monterrey Escuela de Ingeniería

Programación de estructuras de datos y algoritmos fundamentales (Gpo 602)

TC1031.602

Act 6.2 - Reflexión Final de Actividades Integradoras de la Unidad de Formación TC1031 (Evidencia Competencia)

Sofía Moreno López — A01028251

Campus Santa Fe

1 diciembre, 2023

Actividad 1.3:

En Actividad 1.3, se aborda la tarea de organizar la bitácora de accesos a un servidor mediante la aplicación del algoritmo de ordenamiento Insertion Sort. Este método implica la comparación de cada elemento con su predecesor, insertándolo en la posición adecuada. Entre sus ventajas se destacan su eficiencia en listas de pequeño tamaño, su implementación simple, capacidad de ordenamiento en tiempo real, estabilidad y la característica de ordenamiento in situ, lo que implica que solo requiere un espacio constante. No obstante, es importante mencionar que su eficiencia es notablemente inferior en comparación con otros algoritmos de complejidad O(n^2), como Quick Sort o Merge Sort. Estos últimos, al emplear estrategias de "divide y vencerás", logran complejidades más bajas, específicamente O(n log n), aunque QuickSort puede presentar una complejidad más alta en su peor caso, que se manifiesta cuando los datos ya están ordenados o casi ordenados. En el ámbito de la ciberseguridad, la rapidez en la detección de infecciones y vulnerabilidades es crucial para el mantenimiento de servicios y servidores. Esto incluye la capacidad de ordenar la información rápidamente con diferentes criterios para detectar actividades sospechosas o maliciosas.

Actividad 2.3:

En la segunda actividad, los datos de la misma bitácora se gestionan mediante una lista ligada, organizándolos por dirección IP. Una lista ligada se compone de nodos que contienen datos y enlaces al siguiente nodo. A pesar de que las listas ligadas facilitan la inserción y eliminación con complejidad O(1), no permiten el acceso directo a elementos individuales como los arreglos o el acceso aleatorio. Se sugiere mejorar la eficiencia de esta tarea mediante el uso de una lista doblemente ligada, que ofrece ventajas como la navegación bidireccional, inserción y eliminación eficientes en ambos extremos. En este contexto, se emplearon librerías preexistentes, destacando la importancia de comprender la aplicación práctica de estos conceptos en el código diario. Específicamente, se utilizó la librería "list" con la función de "sort", que implementa una combinación de QuickSort y Merge Sort, los dos algoritmos previamente mencionados.

En el contexto de accesos de la bitácora el manejo de estos datos– el uso de estructuras de datos, como las listas ligadas, adecuados para las necesidades que presenta está problemática es crucial para mantener la integridad de un sistema.

Actividad 3.4:

En la tercera actividad, los datos de la bitácora ya organizados por la actividad 2.3 se almacenan en un árbol binario, en particular, decidimos implementarlo en un max heap o árbol binario con prioridad ordenado por dirección IP como indicado. Un árbol binario es una estructura jerárquica que consta de nodos con un máximo de dos hijos, siendo útil para representar relaciones de orden y realizar búsquedas e inserciones eficientes. La elección de un max heap implica que cada nodo tiene un valor mayor o igual que sus hijos, logrando así una complejidad de ordenamiento O(n log n). En este caso, se recurrió a una librería para aprovechar la utilidad de esta estructura en el rápido ordenamiento y búsqueda en la bitácora. La identificación de las cinco direcciones IP con más repeticiones se logra simplemente tomando los primeros cinco valores del montículo.
Repeticiones de las direcciones ip en el contexto de la ciberseguridad podrían constituir un comportamiento sospechoso y una eficiente identificacion de estas direcciones de ip pueden probar ser la diferencia entre si se detecta una visita maliciosa al sistema o si pasa desapercibida.

Evaluación de Competencias:
Las cuatro competencias se usaron durante el desarrollo y reflexion sobre las actividades integrales:

- SICT0301B (Evaluación del problema): El abordaje exitoso de estas actividades demandó una evaluación meticulosa del problema desde el principio, identificando necesidades, datos y contexto. Se destacó la comprensión del problema de prevención de infecciones en un sistema, reconociendo los desafíos a abordar en función de las características y necesidades planteadas.

- SICT0302B (Toma de decisiones): La aplicación de conocimientos computacionales y teóricos fue esencial para analizar y seleccionar la solución ideal a los problemas de ciberseguridad. Aunque no siempre se optó por la opción más eficiente, comprender las limitaciones de la solución facilitó la proposición de algoritmos y estructuras de datos más adecuadas durante la toma de decisiones.

- SICT0303B (Implementación de acciones): La correcta implementación de la teoría computacional fue crucial para abordar las necesidades de ciberseguridad planteadas en estas actividades. Esto implicó la habilidad para implementar correctamente los sistemas propuestos durante la toma de decisiones mediante un pensamiento computacional efectivo.

- SEG0702A (Tecnologías de Vanguardia): La capacidad para adoptar y aplicar tecnologías nuevas, como librerías y estructuras específicamente en C++, previamente desconocidas, fue fundamental para implementar las acciones de manera innovadora y eficiente. Este enfoque permitió conocer las tecnologías utilizadas actualmente para resolver problemáticas similares y completar con éxito las tareas integrales.