Tecnológico de Monterrey, Campus Monterrey
Programación de estructuras de datos y algoritmos fundamentales
Rodrigo Llaguno Cárdenas, A01198067
Grupo 206

Reflexión Act 5.2

Para la entrega de la quinta evidencia de nuestra actividad integral, teníamos que utilizar la estructura de datos de Hashmap o tabla de hash para almacenar la información de ataques cibernéticos en forma de Botnet por la red de su IP donde dicha IP sea la llave de la tabla hash. Como atributos, se almacenarán las IP únicas que intentaron acceder a la IP junto con el número de veces que lo intentaron hacer. Finalmente se creará un método en el cual se entregue una dirección IP y se despliegue el elemento asociado a ella y su información.

Existen diferentes tipos de ciberataques pero el tipo con el cual estamos trabajando en esta evidencia es un botnet. Las botnets vienen de las palabras robot y network y sirven para describir dichos ataques los cuales son conectados por un malware y controlado por un propietario de los robots. Primero, estos robots infectan a las víctimas como computadoras mediante malware o incluso ingeniería social, término que describe la acción de infiltrarse usando técnicas sociales. Después se debe ampliar la botnet para infectar aún más otros equipos donde finalmente se activa la red. Con esto se pueden hacer desde acciones muy simples como leer y escribir datos de un sistema, enviar mensajes, instalar aplicaciones, recopilar información personal, hasta cosas más complejas como espiar en víctimas, crear ataques DDoS o hacer actividades de cripto minería (Belcic, 2021). Es importante identificar una botnet y poder tomar acción contra ella.

La tabla hash es una estructura de datos la cual consiste en ligar *llaves* con valores. Este proceso se hace mediante una función llamada hash. Lo que hace esta función es insertar información en un arreglo donde cada valor de información tiene una llave única. Esto permite que la recolección de datos sea segura (GeeksForGeeks, 2022). Dentro de las tablas hash existe un concepto llamado colisiones, esto es cuando un valor produce un mismo valor llave o de *index*. Existen dos maneras principales de lidiar con esto, el hashing cerrado y el hashing abierto. El cerrado nos dice que se crea una lista enlazada o arreglo por medio del encadenamiento en donde se pone toda la información que tiene una misma llave. El abierto

utiliza el *probe* para encontrar una ubicación en donde pueda almacenar la información cuando ya esté ocupada la llave. Las tablas de hash se utilizan en diferentes áreas como en bases de datos para indexar datos, en lenguajes de programación como Python para crear objetos, para la verificación de contraseñas, etc (GeeksForGeeks, 2022). Los algoritmos principales de una tabla hash se mostrarán a continuación junto con su complejidad:

- show(), esta función recorre la tabla hash y muestra los elementos actuales. Complejidad O(n)
- insert(), esta función inserta un valor en una llave especificada anteriormente. Complejidad O(n)
- find(), esta función busca un valor dentro de la tabla hash por medio de la llave. Complejidad O(n)
- delete(), esta función elimina el valor de una llave de la tabla hash dejándola disponible para nuevas inserciones. Complejidad O(n)
- update(), esta función cambia el valor dentro de una llave especificada. Complejidad O(n)

Algunas ventajas de las tablas hash son que son una estructura de datos eficiente, aun más que árboles de búsqueda debido a su sistema de llaves. Además la complejidad se mantiene constante para todas sus funciones como se puede observar anteriormente, es por eso que es una estructura de datos sincronizada. Sin embargo, hay que tener en cuenta que la tabla hash se vuelve cada vez más ineficiente conforme se vayan creando colisiones. Además no acepta valores nulos (GeeksForGeeks, 2022).

En esta evidencia, es de gran utilidad la tabla hash puesto que cada ip, es una llave para la tabla hash. Con esta información se puede almacenar fácilmente mediante hashing cerrado o abierto los valores relacionados a esta llave. Así pudiendo analizar más eficientemente la información de la red de bots.

Para la quinta evidencia, decidimos crear una tabla hash donde se insertaran los ip nodo como ‘llaves’ de la tabla hash. Esto permite que se pueda acceder a un valor. Después en la tabla hash para cada llave, se tendrá un struct de tipo Element donde cada uno contenga el IP y el número de veces que fue destino y origen. Así cada llave tendrá su arreglo de información con la que podemos después crear la función de búsqueda que se nos pidió. Para esa función solamente solicitamos que el usuario ingrese el IP a buscar y con la función hash, se busca la

llave y se muestra toda la información almacenada. Finalmente desplegamos en un archivo nuevo todos los datos de cada IP.

Referencias:

Belcic, I. (2021, October 8). *¿Qué es una botnet? | Buscador gratuito de botnets*. Avast. https://www.avast.com/es-es/c-botnet

GeeksForGeeks. (2022, June 7). *Applications, Advantages and Disadvantages of Hash Data Structure*. GeeksforGeeks. Retrieved November 21, 2022, from https://www.geeksforgeeks.org/applications-advantages-and-disadvantages-of-hash-data-structure/