



**FACULTAD DE INGENIERÍA, ARQUITECTURA Y
URBANISMO**

**ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMAS
TESIS**

**EVALUACION DE LA EFICIENCIA DEL ALGORITMO TOKEN BUCKET
PARA MITIGAR LA DENEGACION DE SERVICIOS DISTRIBUIDA EN
MUNICIPALIDADES PERUANAS. CASO DE ESTUDIO MUNICIPALIDAD DE
SANTIAGO DE SURCO**

**PARA OPTAR EL TÍTULO PROFESIONAL DE INGENIERO
DE SISTEMAS**

Autor(a) (es):

Bach. Gonzales Guevara Rommel Andres

ORCID: <https://orcid.org/0000-0001-6052-0444>

Asesor(a):

Mg. Bances Saavedra David Enrique

ORCID: <https://orcid.org/0000-0002-7164-8918>

Línea de Investigación:

Infraestructura, Tecnología y Medio Ambiente

Pimentel – Perú 2021

APROBACIÓN DEL JURADO

EVALUACION DE LA EFICIENCIA DEL ALGORITMO TOKEN BUCKET PARA MITIGAR LA DENEGACION DE SERVICIOS DISTRIBUIDA EN MUNICIPALIDADES PERUANAS. CASO DE ESTUDIO MUNICIPALIDAD DE SANTIAGO DE SURCO

Bachiller, Gonzales Guevara Rommel Andres
Autor

Mg. Bances Saavedra David Enrique
Asesor

Grado, apellidos y nombres
Presidente de Jurado

Grado, apellidos y nombres
Secretario de Jurado

Grado, apellidos y nombres
Vocal de Jurado

Dedicatorias

Agradezco a mis padres José Luis Gonzales Pacheco y Jenny Miluska Guevara Valdez, por estar siempre conmigo en cada paso que doy, por fortalecerme cada día con sus sabios consejos y que con su gran amor incondicional guiaron mi vida durante todo este largo camino.

A mi hermano Alberto Bairon Gonzales Guevara y a mí adorado hijo Marco Andree Gonzales Flores que me dieron la fuerza y fortaleza para poder realizar y culminar con este trabajo.

También agradezco este trabajo a Patricia Rodríguez Carrasco quien me acompañó durante todo este tiempo para el desarrollo de la presente tesis.

Agradecimientos

Dedico este trabajo a mi hermosa familia que siempre están apoyándome continuamente con mis estudios, mis objetivos y mis metas.

A mi hijo hermoso y a mí adorada novia que son la razón de mí ser y la bendición más grande que la vida me ha dado, es que me esfuerzo a diario.

Resumen

En el presente Trabajo de Investigación se determinaron los mejores algoritmos para defenderse de los ataques de Denegación de Servicios Distribuida, con un conjunto de datos de 494 registros de paquetes de datos, el diseño de la investigación fue cuasiexperimental, la técnica de recolección de datos fue con instrumentos mecánicos o electrónicos y el instrumento fue el registro electrónico.

Se detalle los aspectos teóricos de los Algoritmos Token Bucket y Leaky Bucket, para su comparación y debidas técnicas de evaluación.

Para terminar, se demuestra que el algoritmo Token Bucket es más eficiente que el algoritmo Leaky Bucket para nuestro caso de estudio en la Municipalidad de Santiago de Surco, ya que obtiene una precisión de 96.9%, sin embargo, también se observa que combinando el algoritmo Token Bucket con otros algoritmos o sistemas de clasificación se puede mejorar la precisión que se obtiene.

Palabras Clave:

Ataque de Denegación de Servicio Distribuida

Token Bucket

Leaky Bucket

Abstract

In this Research Work, the best algorithms to defend against Distributed Denial of Services attacks were determined, with a data set of 494 records of data packets, the research design was quasi-experimental, the data collection technique was with mechanical or electronic instruments and the instrument was the electronic record.

The theoretical aspects of the Token Bucket and Leaky Bucket Algorithms are detailed, for their comparison and due evaluation techniques.

Finally, it is demonstrated that the Token Bucket algorithm is more efficient than the Leaky Bucket algorithm for our case study in the Municipality of Santiago de Surco, since it obtains a precision of 96.9%, however, it is also observed that combining the algorithm Token Bucket with other algorithms or classification systems can improve the precision obtained.

Keywords:

Distributed denial of service attack

Token Bucket

Leaky Bucket

Índice

I. INTRODUCCIÓN	8
1.1. Realidad Problemática.	8
1.2. Trabajos previos.	11
1.3. Teorías relacionadas al tema.	15
1.4. Formulación del Problema.	19
1.5. Justificación e importancia del estudio.	19
1.6. Hipótesis.	20
1.7. Objetivos.	20
1.7.1. Objetivo general.	20
1.7.2. Objetivos específicos.	20
II. MATERIAL Y MÉTODO	20
2.1. Tipo y Diseño de Investigación.	20
2.2. Población y muestra.	21
2.3. Variables, Operacionalización.	22
2.4. Técnicas e instrumentos de recolección de datos, validez y confiabilidad.	23
2.5. Procedimiento de análisis de datos.	23
2.6. Criterios éticos.	24
2.7. Criterios de Rigor Científico.	24
III. RESULTADOS.	25
3.1. Resultados en Tablas y Figuras.	25
3.2. Discusión de resultados.	31
3.3. Aporte práctico.	32
IV. CONCLUSIONES Y RECOMENDACIONES	41
4.1. Conclusiones.	41
4.2. Recomendaciones.	42
REFERENCIAS.....	42
ANEXOS.	45

I. INTRODUCCIÓN

En este mundo cada vez más globalizado e interconectado, se considera primordial que los sistemas de comunicación sean más eficientes en el transporte y difusión de la información.

Como resultado de estos esfuerzos se desarrolló una red de computadoras por encargo del Departamento Defensa de los Estados Unidos, conocida como ARPANET (Advanced Research Project Agency Network) en 1972. Después de muchos años de pruebas y errores, nació en 1989 la WWW (Word Wide Web), y comercializado en 1992 para todo el público en general. Al inicio el tamaño de la red era reducido, pero con el paso del tiempo se fueron añadiendo nuevas conexiones e infraestructuras, haciendo que la red sea de un tamaño considerablemente grande, que actualmente se conoce como INTERNET.

Si bien es cierto, el internet fue desarrollado para resolver los problemas de comunicación entre lugares distantes. Pero ni bien ha ido creciendo la red de redes (INTERNET), se han ido presentando diversos problemas que se deben afrontar. Uno de ellos es el tema de la seguridad.

Para el presente proyecto de Investigación vamos a tratar uno de los tipos de ataques más destacadas en la ciberseguridad: los ataques de Denegación de Servicio Distribuido (DDoS).

1.1. Realidad Problemática.

Los programas de ataques de Denegación de Servicios Distribuido (DDoS), han existido durante un largo tiempo y hasta la fecha, ningún sistema de seguridad ofrece un método de defensa eficaz para los ataques de Denegación de Servicios Distribuido (DDoS).

Los ataques de Denegación de Servicio Distribuido (DDoS), han evolucionado de una manera muy rápida con el pasar de los años, y se han vuelto más peligrosos y son muchos más avanzado, técnicamente hablando.

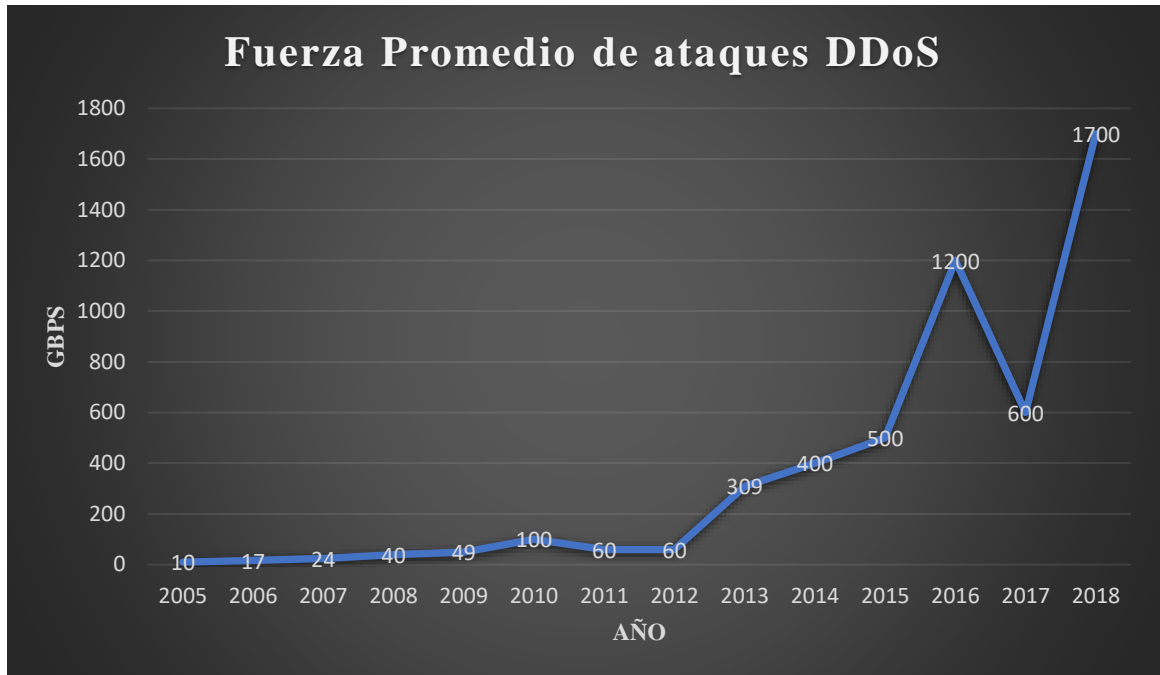


Figura 1 Fuerza Promedio de Ataques DDoS

Fuente: Elaboración Propia

Como se muestra en la Figura 1, se puede visualizar el aumento notable de los ataques DDoS realizados en el mundo, lo que ocasiona una alerta para todas las comunidades de ciberseguridad.

Según Martín Fuentes, security business manager sénior de CenturyLink, señaló a El Peruano que estos ciberataques se han elevado durante los últimos años. En el 2018, el tamaño y complejidad de los ataques DDoS aumentó en 273%, según el último informe de Century Link. La tendencia se mantiene en el 2019. (El Peruano, 2019)

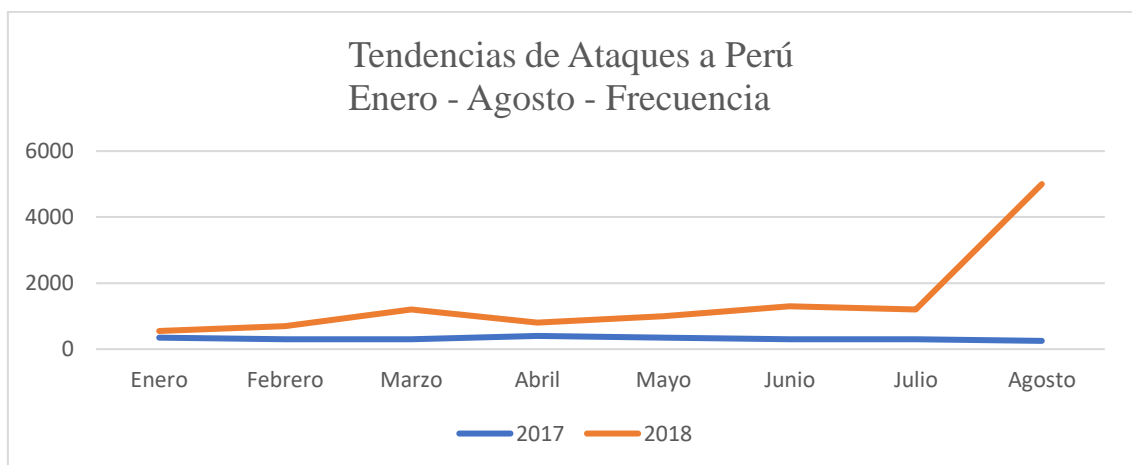


Figura 2 Tendencias de Ataque a Perú - 1er Trimestre

Fuente: Elaboración Propia

Como se muestra en la Figura 2, los ataques realizados a Perú en el primer trimestre del 2017, fueron un total de 2200, y durante el primer trimestre del 2018 fueron 12000 ataques en total, lo que significa un aumento de 550% de ataques realizados a nuestro País, según el informe realizado por la empresa NETSCOUT SYSTEMS, INC.

En nuestro caso de estudio que es la Municipalidad Distrital de Santiago de Surco, se ha podido validar que también ha recibido ataques de Denegación de Servicios Distribuido (DDoS). Según el reporte de su ISP como se muestra en la Figura 3, se obtiene que por lo menos es atacado una vez al mes.

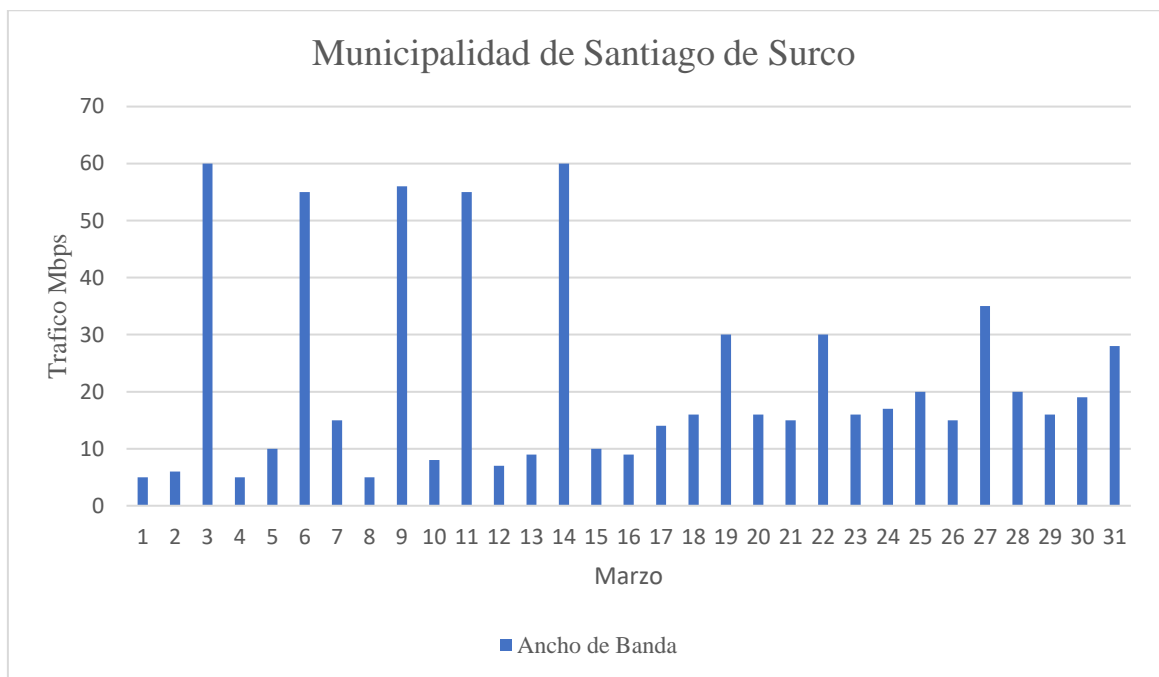


Figura 3 Trafico entrante de Ancho de Banda en la Municipalidad de Santiago de Surco en el mes de Marzo del 2020

Fuente: Elaboración Propia

1.2. Trabajos previos.

Anselme Ndikumana, Saeed Ullah, Kyi Thar, Nguyen H. Tran, Bang Ju Park, Choong Seon Hong (2017), realizó la investigación Novedoso mecanismo de control de congestión totalmente distribuido y cooperativo para redes centradas en el contenido, en Corea del Sur. Propuesta de un mecanismo de control de congestión para la red centrada en el contenido. Desbordamiento de búfer el cual provoca la pérdida de paquetes y que es una señal de congestión de la red. Por lo tanto, los mecanismos de control de congestión existentes propuestos principalmente para el Protocolo de Control de Transmisión (TCP) no se puede implementar en la red centrada en contenido. Se previene la congestión antes de que ocurra mediante el monitoreo del tamaño del buffer, por lo tanto, el nodo notifica a su nodo o nodos descendientes. Al recibir la notificación, el nodo inferior ajusta la tasa de tráfico asignando nuevos Intereses entrantes. cuando el nodo inferiro no logra reducir la velocidad del tráfico, el mismo procedimiento continúa hasta que el nodo superior reduce la velocidad de envío. Los resultados de la simulación muestran que el mecanismo propuesto es capaz de mejorar significativamente el rendimiento y evitar el congestionamiento de la red. El artículo en mención muestra un nuevo mecanismo cooperativo y de control de congestión, que nos avisa antes que la congestión ocurra. Por lo cual mezcla CMTB salto por salto, que es parecido a una modificación de Dynamic Token Bucket, y con la gestión controladabadao en FDCC (modificación de Additive-Increase / Multiplicative-Decrease). De esta forma se logra aumentar el rendimiento de la red y el retraso experimentado en los que paquetes de salida se reducen.

AQEEL SAHI, DAVID LAI, YAN LI, (Member, IEEE), AND MOHAMMED DIYKH (2017), realizó la investigación de Un sistema eficiente de detección y prevención de ataques de inundación DDoS TCP en un entorno de nube, en Iraq. Los ataques DDoS que son de tipo inundación TCP, se usan para inundar la máquina de una víctima con paquetes de datos, y consumir sus recursos o ancho de banda. En la actualidad, la computación en la nube está en un auge muy rápido en diversos sectores, como por ejemplo el sector salud, debido a su disponibilidad y el pedido de sus servicios. Por lo tanto, casi todas las personas

que van haciendo uso de la computación en la nube, empieza a pensar en esto como una red virtual que va a traer mucha demanda de servicios de forma accesible y flexible. Este nuevo sistema de clasificador propuesto para la detección y prevención de ataques de inundación DDoS TCP (CS_DDoS), está basado en la clasificación y puede identificar estos datos de ataques independientemente de la forma en que llegan al sistema en la nube. En el sistema que se ha propuesto, también se definen 2 subsistemas, uno de ellos es el subsistema de prevención y el otro es el subsistema de detección. En la etapa de detección, este subsistema toma todos los paquetes entrantes en un periodo de tiempo. Y estos paquetes capturados pasan por una revisión para validar que sus fuentes no están en ninguna lista negra de atacantes. Si la fuente del paquete llega a aparecer en una lista negra, el subsistema de detección va a enviar esos paquetes inmediatamente al subsistema de prevención, donde termina todo el proceso. Y si la fuente del paquete no aparece en ninguna lista negra, entonces dicho paquete va a pasar a un clasificador para determinar si es un paquete normal o no. Cuando los paquetes llegan al sistema de prevención, el subsistema de detección considera que están atacando paquetes. El subsistema de prevención empieza mandando un aviso al administrador. Para luego agregar la dirección de la fuente del ataque, a la lista negra que utiliza el subsistema de detección. Finalmente, el paquete atacante será descartado. el sistema CS_DDoS, es eficaz y estable para resistir ataques de múltiples fuentes y de una sola fuente cuando se usa el clasificador LS-SVM. En conclusión, este sistema CS_DDOS ayuda a prevenir un ataque DDoS con una precisión del 94% y es muy estable, por lo que se puede implementar es un proyecto en la nube de gran tamaño, así como también en proyecto pequeños. Usando LS-SVM el sistema CS_DDoS puede identificar los ataques con precisión, mejorando de manera eficiente la seguridad de los registros, reduciendo el consumo de ancho de banda y mitigando el agotamiento de los recursos.

MARCOS V. O. . NOVAES, CINARA B. ZERBINI, LUIZ F. CARVALHO, TAUFIK ABRÃO, AND MARIO L. PROENÇA Jr. (2018), realizó la investigación de Sistema de defensa rápida contra ataques en redes definidas por software, en Brasil. Los ataques DoS intentan agotar los recursos de la red y es más poderoso cuando se realiza de forma distribuida (DoS distribuido o DDoS). Cuando se atacan los servidores, el objetivo del atacante es que un servicio ya no se encuentre disponible enviando diversas solicitudes al mismo tiempo, en cambio los ataques de infraestructura, el atacante congestiona un enlace de red. La gestión de la seguridad de la información de la red es una tarea de alta complejidad, ya que es necesario garantizar la disponibilidad, fiabilidad e integridad de los servicios de red prestados a los usuarios finales. Se presenta un sistema de defensa SDN, cuyo fin es proporcionar ayuda defendiéndonos contra los ataques de escaneo de puertos, y los DDoS. El sistema que se está presentando de Defensa SDN, integra un exportador de flujo IP y tres módulos. Para el desarrollo de este sistema el exportador de flujo utilizado es OpenFlow. Según los resultados obtenidos, se aprecia que la mitigación de los ataques DDoS fue eficiente para restablecer a su funcionamiento normal el sistema SDN. Por lo cual, se tiene en mente estudiar el comportamiento de otras anomalías en entornos SDN. Además, tenemos la intención de mejorar el enfoque de mitigación con el objetivo de reducir aún más el impacto sufrido por los usuarios legítimos en el proceso.

O. Boyar, M. E. Özen and B. Metin (2018), realizó la investigación Detección de ataques de denegación de servicio con SNMP / RMON, en España. Se trata de realizar un laboratorio donde se va a realizar un ataque DoS controlado hacia un host específico. Se usan Redes neuronales, vecinos más cercanos a K y C5.0 para identificar los paquetes de ataques DoS a un host específico mediante el análisis de los datos SNMP/RMON. En conclusión, las redes neuronales se desempeñaron mucho más eficientemente que vecino cercano y algoritmo C5. Eso no quiere decir vecino cerca y algoritmo C5 no lo hicieron, por el contrario, también se desempeñaron muy bien.

Abdelrahman Manna and Mouhamad Alkasassbeh (2019), realizó la investigación de Detección de anomalías de red mediante aprendizaje automático y conjunto de datos SNMP-MIB con grupo IP, en Jordan. SNMP-MIB usa el aprendizaje automático para clasificar datos y de esta forma obtener resultados, pero cuando se usa en un gran conjunto de datos, no es eficiente y a la vez consume demasiado tiempo y recursos. Por eso se utilizó un árbol de decisión y también clasificadores de bosque aleatorio, para entrenar un modelo que pueda detectar las anomalías dentro de la red para predecir los ataques que puedan afectar a un grupo de IP. Se identificó que el SNMP da resultados positivos en la detección de ataques DDoS. En conclusión, este modelo preparado lo pueden utilizar en dispositivos que puedan detectar anomalías, como en sistemas de detección de intrusos.

Sulaiman Alhaidar, Ali Alharbi, Mansour Alshaikhsaleh, Mohamed Zohdy and Debatosh Debnath (2019), realizó la investigación de Detección de anomalías de tráfico de red basada en el algoritmo de Viterbi utilizando datos MMP de SNMP, en Mning. Durante los últimos años la Denegación de Servicios (DoS), ha sido solo uno de los ataques más críticos para la seguridad informática, siendo grave para muchos servicios en la internet. Siendo así, se han visto en la necesidad de usar nuevas técnicas de aprendizaje automático junto con criptografía y algunos parámetros de seguridad de firewall, para así poder mejorar el rendimiento de los Sistema de Detección de Intrusos (IDS). La presente investigación nos muestra cómo utilizar Hidden Modelo de Markov (HMM), basado en el algoritmo de Viterbi, que nos sirve para detectar anomalías en los datos SNMP MIB. De esta forma se compara estos 2 algoritmos de clasificación: Adaboost M1 y Naive Bayes. Los resultados que se han obtenido nos enseñan que HMM basado en el algoritmo de Viterbi son efectivos, logrando muy buenos resultados en la detección de ataques DDoS. El algoritmo de Viterbi es eficiente en la detección de ataques DDoS y otras anomalías en la red.

Erkin Kirdan, Daniel Raumer, Paul Emmerich y Georg Carle (2018), realizó la investigación de Creación de un controlador de tráfico para la mitigación de DDoS sobre hardware básico, en Alemania. La detección de ataques DDoS mediante el análisis de subredes usa un controlador de tráfico como la forma de solucionar la mitigación de los ataques de Denegación de Servicios Distribuido (DDoS), donde se puede prohibir el paso de los usuarios proveniente de una subred ilegítima, al mismo tiempo que los usuarios de una subred legítima, pueden seguir usando los servicios de internet con normalidad. Los ciclos de CPU por paquete aumentan de 432 a 492. Por lo tanto, con este controlador de tráfico escalable, se puede mitigar los ataques de Denegación de Servicio Distribuido (DDoS), gracias a su alto rendimiento.

1.3. Teorías relacionadas al tema.

Ataques DDoS

Los ataques de Denegación de Servicio (DoS), es solo un tipo de ataque a la red, donde un host o PC, envía una gran cantidad de datos a otro host o aplicación que este no puede procesar, ocasionando una disminución en la velocidad de respuesta o falla en el servicio o servidor.

Los ataques de Denegación de Servicio ahora son una amenaza constante para toda la información en la sociedad. “Según ha publicado recientemente la Agencia Europea de Seguridad de las Redes y de la Información (ENISA), entre los años 2013 y 2014 se observó su incremento en un 70 %” (T. Peng, 2007) . Los ataques de Denegación de Servicio Distribuido (DDoS), es parecido a un ataque DoS, pero este ataque viene que distintos equipos a la vez de manera coordinada.

Para esto el atacante primero debe tener varios hosts infectados en la red, a los cuales se les van a llamar botnet, que actúan como zombies controlados por los sistemas manipuladores.

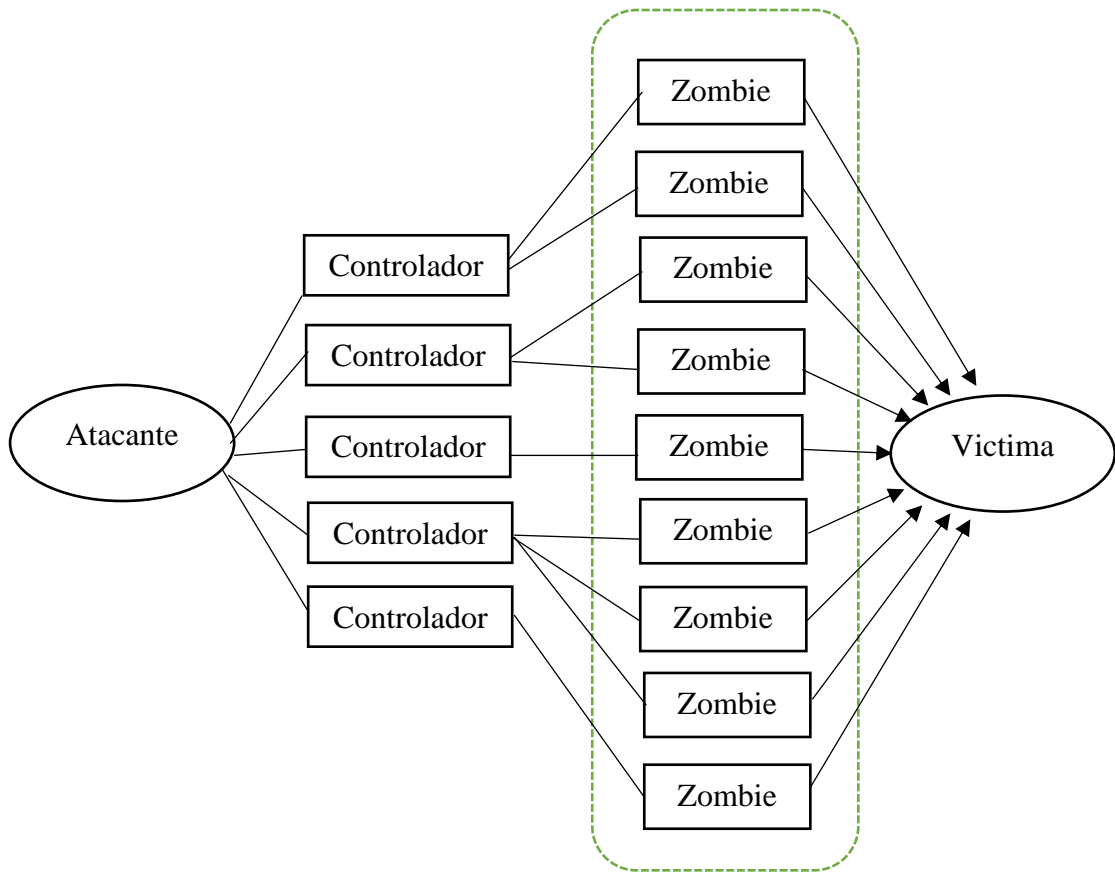


Figura 4 Arquitectura de un Ataque DDoS

Fuente: Elaboración Propia

En la actualidad existen 2 tipos de ataques conocidos que son inundación y vulnerabilidad. Es importante una clasificación adecuada para entender el tipo de ataque de DDoS al que estamos enfrentando.

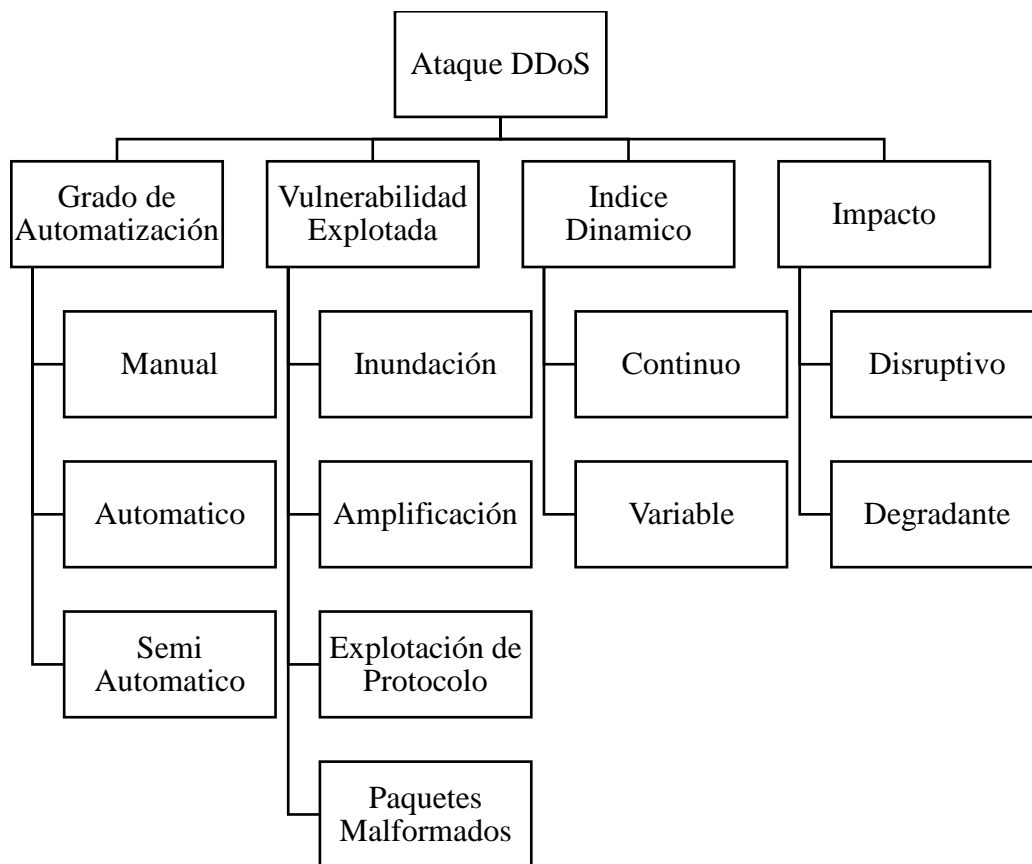


Figura 5 Clasificación de Ataque DDoS

Fuente: Elaboración Propia

a. Grado de Automatización

- Manual: Implica analizar el host que se quiere atacar para encontrar agujeros de bucle, y poder penetrarlos para instalar códigos de ataques.
- Automático: Es cuando un solo ataque inicia todo el proceso de DDoS.
- Semi Automático: Es cuando el atacante debe insertar los scripts de forma manual en los controladores, y organizar el ataque, para luego desplegarlo de forma automática controlando a los zombies o botnets.

b. Vulnerabilidad Explotada:

- Inundación: Implica congestionar el ancho de banda de la víctima, enviando una gran cantidad de tráfico.
- Amplificación: Aquí se utiliza las características de la dirección IP de difusión, donde se ordena a los enrutadores o routers que transmitan paquetes de datos fuera de la red a cada IP dentro del rango de difusión.
- Explotación de Protocolo: es cuando se aprovecha el atributo o error específico de un protocolo específico en el sistema de la víctima, de modo que se pueda consumir todos los recursos posibles.
- Paquetes Malformados: Básicamente es cuando se altera un paquete colocando la misma dirección IP de origen en la IP de destino, haciendo que el sistema de la víctima se confunda y luego falle.

c. Índice Dinámico

- Continuo: Son ataques que realizan con fuerza, sin detenerse, ocasionando un rápido impacto.
- Variable: Son ataques que se realizan de forma con una velocidad creciente, fluctuante, haciendo más difícil la detección del ataque DDoS.

d. Impacto:

- Disruptivo: Es cuando se corta la comunicación entre 2 dispositivos, resultando un DoS completo.
- Degradante: Solo consume cierta cantidad de recursos de la víctima, causando retraso en la detección y un gran daño en el sistema atacado.

Token Bucket

Es un algoritmo que permite para controlar el tráfico hacia nuestra red, lo que reduce el consumo de recursos para nuestros enrutadores. Esto se logra haciendo una cola que permite el paso de los paquetes que tengan un token disponible, impuesto en la tasa de transferencia limite aceptada.

Se configura cierta cantidad de token por cada segundo, para permitir el paso de los paquetes, y en caso no hay tokens disponibles, se almacenan los paquetes en buffer hasta que nuevos tokens estén disponibles.

$$C + \rho S = MS$$

Esta ecuación representa la cantidad de ancho de banda que puedo consumir durante una ráfaga de paquetes.

$$F(t) = \frac{C}{N(t)}$$

Con esta ecuación podemos determinar la cantidad de buffer de tokens para los paquetes que ingresan.

1.4. Formulación del Problema.

¿Cómo mitigar en forma eficiente la denegación de servicios distribuidos en las redes de datos de municipalidades peruanas?

1.5. Justificación e importancia del estudio.

Los ataques de DDoS tienen como objetivo inhabilitar un servidor, un servicio o una infraestructura. Durante un ataque DDoS, se envían simultáneamente múltiples solicitudes desde distintos puntos de la red, comprometiendo la estabilidad, y, en ocasiones, la disponibilidad del servicio.

Al inhabilitar la disponibilidad de un servicio, afecta económicamente y comercialmente a las Municipalidades, lo que se traduce en pérdidas de dinero y de confianza hacía el usuario, por tal motivo se busca la forma de mitigar los ataques DDoS, utilizando el algoritmo token bucket.

Actualmente los ataques DDoS van en aumento cada año y los hackers lo usan como herramienta principal para perjudicar a un entidad pública o privada, lo que produce pérdidas cuantitativas como se mencionó anteriormente, por tal

motivo se considera importante el estudio de la mitigación de este tipo de ataque DDoS.

1.6. Hipótesis.

Mediante la implementación del algoritmo Token Bucket se mitigará en forma eficiente la denegación de servicios distribuidos en las redes de datos de municipalidades peruanas.

1.7. Objetivos.

1.7.1. Objetivo general.

Evaluar la eficiencia del algoritmo Token Bucket para mitigar la Denegación de Servicios Distribuida en Municipalidades Peruanas.

1.7.2. Objetivos específicos.

- Analizar el caso de estudio seleccionado antes de la implementación del algoritmo Token Bucket.
- Preparar los escenarios de prueba en el caso de estudio.
- Capturar el tráfico de una red
- Analizar el tráfico capturado con el algoritmo Token Bucket para determinar un ataque DDoS.
- Ejecutar los casos de prueba con otro algoritmo para determinar la eficiencia.

II. MATERIAL Y MÉTODO

2.1. Tipo y Diseño de Investigación.

El tipo de investigación será cuantitativa ya que a través de las técnicas estadísticas, matemáticas o computacionales será posible obtener los resultados de nuestra investigación.

El tipo de diseño de investigación será cuasiexperimental porque la elección de nuestra muestra dependerá del propósito de nuestra investigación.

2.2. Población y muestra.

La Población son todos las Municipalidades Distritales que actualmente hay en el Perú, que son 1655 hasta la fecha del 2020, según el Registro Nacional de Municipalidades 2017 del INEI.

Y la muestra viene a ser mi caso de estudio que es la Municipalidad de Santiago de Surco, el cual es tiene constantes ataques DDoS.

2.3. Variables, Operacionalización.

Tabla 1 Variables y Operacionalización

Variables		Dimensión	Indicador	Ítem	Técnica e instrumentos de recolección de datos
Algoritmo Bucket	Token	Consumo de recursos	Grado de consumo de CPU	$Cc = \sum_j^n \frac{cc_j}{n}$	Registro Electrónico
			Grado de consumo de memoria	$Cm = \sum_j^n \frac{cm_j}{n}$	
			Promedio de tiempo de respuesta	$Tr = \sum_j^n \frac{tf_j - tf_i}{n}$	
Mitigación de Ataques DDoS	de	Rendimiento	Exactitud	$E = \frac{VP + VN}{VP + VN + FP + FN}$	Registro Electrónico
			Precisión	$P = \frac{VP}{VP + FP}$	
			Recall	$R = \frac{VP}{VP + FN}$	
			F-SCORE	$F1 = 2 \cdot \frac{\text{presicion} \cdot \text{exhaustividad}}{\text{presicion} + \text{exhaustividad}}$	

2.4. Técnicas e instrumentos de recolección de datos, validez y confiabilidad.

Para el presente proyecto vamos a usar los instrumentos mecánicos o electrónicos y los de Observación, como instrumentos de recolección de datos.

Instrumento Electrónico: En este proyecto vamos a usar los log de los equipos que se van a utilizar para esta investigación, y el Wireshark para capturar los paquetes de datos entrantes y salientes, para poder desarrollar la investigación.

Observación: Se dice que vamos a recolectar por Observación porque tenemos que leer y observar el comportamiento de los datos recolectados por el instrumento electrónico.

2.5. Procedimiento de análisis de datos.

Para medir las variables identificadas en este proyecto, vamos a utilizar los indicadores con sus respectivas fórmulas matemáticas:

a. Matriz de Confusión: Con la matriz de Confusión podemos calcular nuestros VP, FP, PN, VN.

Tabla 1 Matriz de Confusión

		Predicción	
		Positivos	Negativos
Observación	Positivos	Verdaderos Positivos (VP)	Falsos Negativos (FN)
	Negativos	Falsos Positivos (FP)	Verdaderos Negativos (VN)

b. Precisión: Sirve para medir la precisión del algoritmo utilizado.

$$Precisión = \frac{VP}{VP + FP}$$

c. **Exhaustividad:** Nos informa la cantidad de muestras correctas diagnosticadas.

$$Exhaustividad = \frac{VP}{VP + FN}$$

d. **F-SCORE – Valor:** Compara el rendimiento de la precisión y la exhaustividad.

$$F1 = 2 \cdot \frac{precision \cdot exhaustividad}{precision + exhaustividad}$$

e. **Exactitud:** Mide el porcentaje de casos que se han acertado.

$$Exactitud = \frac{VP + VN}{VP + VN + FP + FN}$$

2.6. Criterios éticos.

Para la elaboración de este proyecto, es fundamental tener en cuenta los siguientes criterios éticos:

- **Derecho de Autor:** Se hizo referencia de las citas y fuentes de investigación durante el proceso de elaboración del presente proyecto de Investigación.
- **Búsqueda del Bien:** La razón principal de la elaboración del presente proyecto de investigación, es la búsqueda del bien común, de tal forma que contribuya al beneficio de la sociedad y la carrera.
- **Confidencialidad:** Durante el desarrollo del presente proyecto de investigación, se guardará la absoluta confidencialidad en caso sea necesario.

2.7. Criterios de Rigor Científico.

Validez: En la presente investigación la validez es de tipo lógico, debido a que se utilizan modelos matemáticos y estándares.

Matriz de confusión:

		Predicción	
		Positivos	Negativos
Observación	Positivos	Verdaderos Positivos (VP)	Falsos Negativos (FN)
	Negativos	Falsos Positivos (FP)	Verdaderos Negativos (VN)

Error cuadrático promedio

$$ECM = \frac{1}{n} \sum_{i=1}^n (\hat{Y}_i - Y_i)^2$$

Confiabilidad: La manera estadística para determinar la confiabilidad es a través de técnicas e instrumentos de recolección de datos mecánicos o electrónicos para determinar métricas de rendimiento en clasificación.

Consistencia: En este proyecto de investigación se asegura la consistencia, repitiendo los resultados en el mismo caso de estudio e igual contexto.

III. RESULTADOS.

3.1. Resultados en Tablas y Figuras.

Los resultados que se muestran a continuación, pertenecen al tiempo promedio de respuesta efectuado al ejecutar cada algoritmo, y se ha obtenido con los indicadores del grado de consumo de CPU, y grado de consumo de memoria, para promediar el tiempo de respuesta.

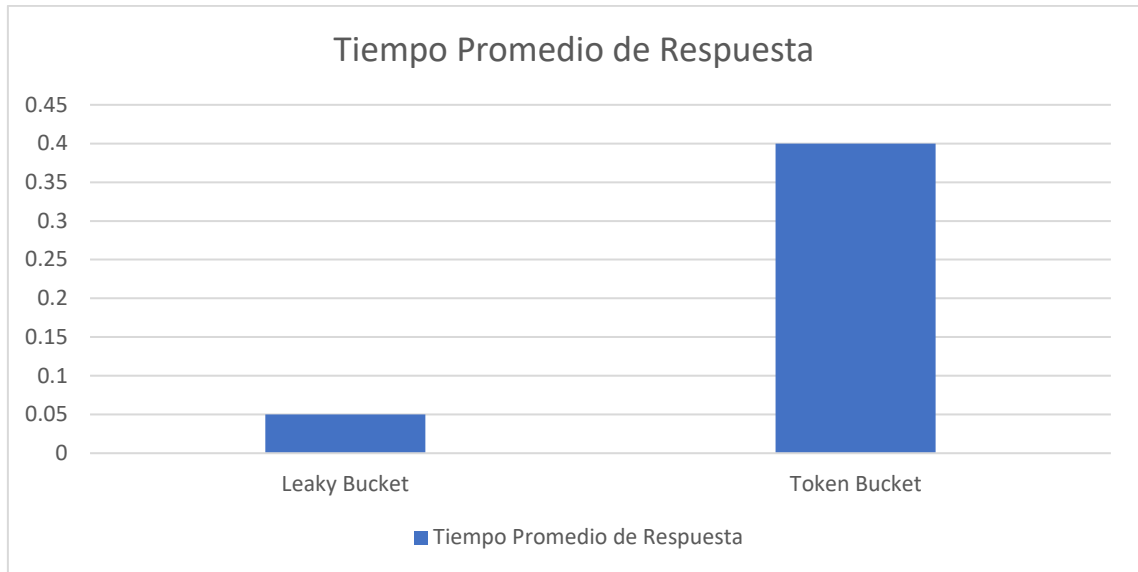


Figura 6 Tiempo Promedio de respuesta por algoritmo

Fuente: Elaboración Propia

En la Figura 6, se observa el tiempo promedio de respuesta para el algoritmo de Leaky Bucket y Token Bucket respectivamente. Con el algoritmo Leaky Bucket, se puede observar que el tiempo promedio de respuesta es de 0.05 segundos, debido a que no se ejecuta en tiempo real, por lo cual consume menos recursos. Y con el algoritmo Token Bucket podemos observar que el tiempo promedio de respuesta es de 0.4 segundos, por lo que este algoritmo se realiza en tiempo real y necesita más recursos para poder ejecutarse, lo cual no significa que un algoritmo sea mejor que el otro.

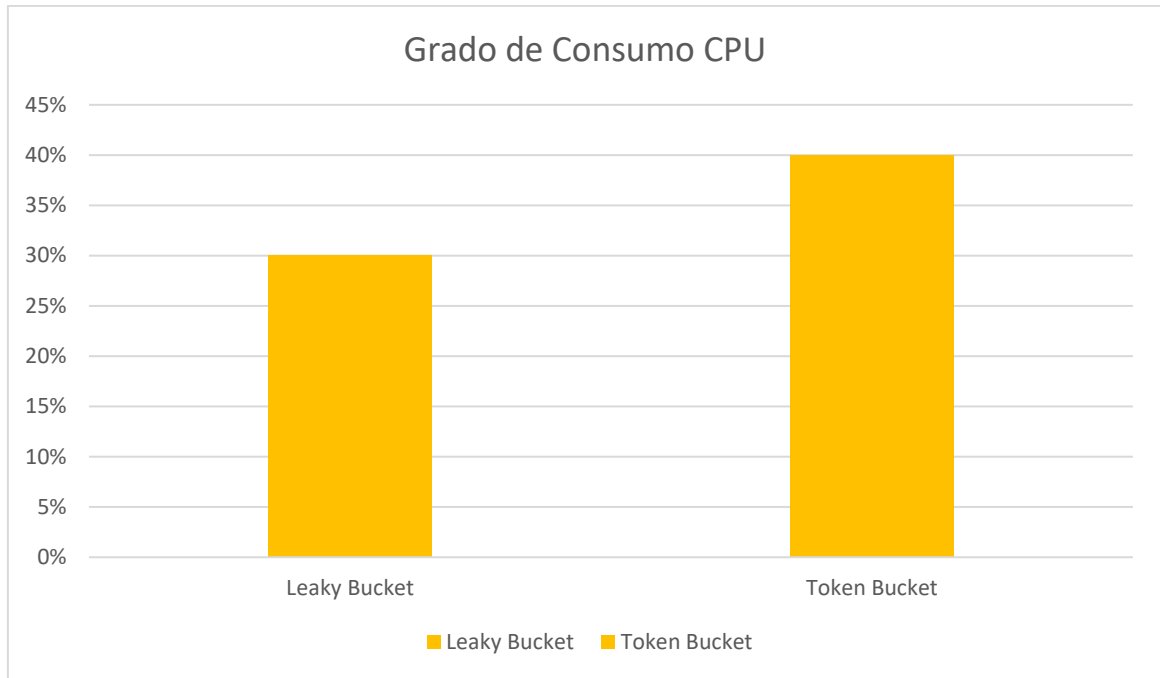


Figura 7 Grado de Consumo CPU

Fuente: Elaboración Propia

En la Figura 7, se muestran los resultados obtenidos con respecto al grado de consumo del CPU, cuando se ejecutan los algoritmos, dando así un 30% de consumo del CPU cuando se usa el algoritmo Leaky Bucket, debido a lo antes mencionado que no se ejecuta en tiempo real. En el caso del algoritmo Token Bucket, se obtiene un consumo del CPU en 40% debido a que este algoritmo se ejecuta en tiempo real.

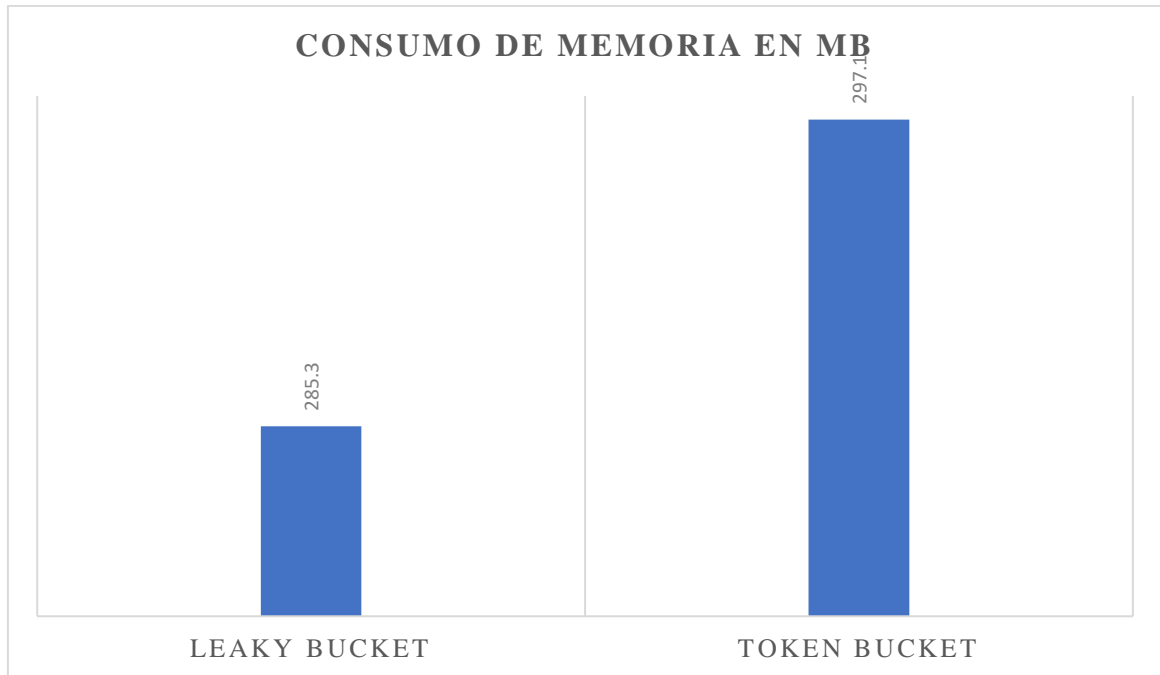


Figura 8 Consumo de Memoria en MB

Fuente: Elaboración Propia

En la figura 8, se visualiza el consumo de la memoria RAM expresado en Megabyte, por cada algoritmo que se ejecuta para analizar el tráfico de los ataques DDoS hacia la red. Al ejecutar el algoritmo Leaky Bucket se puede obtener un consumo de memoria de 285.3 MB y en el algoritmo Token Bucket se puede obtener un consumo de memoria de 297.1, lo cual nos da una diferencia de 11.8 MB.

Los resultados para la evaluar la eficiencia del algoritmo Token Bucket, se obtuvieron en con los indicadores de exactitud, precisión, recall y F-SCORE. Para esto se ejecutó los algoritmos mencionados, para dar como resultado la matriz de confusión y los porcentajes, dependiendo de cada indicador, según se muestra a continuación.

Resultados obtenidos de la Matriz de Confusión por algoritmo de clasificación:

Tabla 2 Matriz de Confusión del Algoritmo Leaky Bucket

		Clasificación	
		No Descartados	Descartados
Reales	Positivo	249	47
	Negativo	15	183

El algoritmo Leaky Bucket obtuvo 264 paquetes de datos como positivos y 230 paquetes de datos como negativos, haciendo un total de 494 registros.

De acuerdo a la tabla de la Matriz de Confusión del Algoritmo Leaky Bucket, de los paquetes positivos solo se logró predecir 249 paquetes positivos, y 15 paquetes negativos cuando en realidad eran paquetes positivos.

Por otro lado, de los paquetes negativos detectados solo se logró predecir 183 paquetes negativos de manera correcta, y 47 paquetes de datos erróneos, debido a que el algoritmo lo predijo como negativo, cuando en realidad era positivo.

Tabla 3 Matriz de Confusión del Algoritmo Token Bucket

		Clasificación	
		No descartados	Descartados
Reales	Positivo	254	42
	Negativo	8	190

El algoritmo Token Bucket obtuvo 262 paquetes de datos como positivos y 232 paquetes de datos como negativos, haciendo un total de 494 registros.

De acuerdo a la tabla de la Matriz de Confusión del Algoritmo Token Bucket, de los paquetes positivos solo se logró predecir 256 paquetes positivos, y 8 paquetes negativos cuando en realidad eran paquetes positivos.

Por otro lado, de los paquetes negativos detectados solo se logró predecir 190 paquetes negativos de manera correcta, y 42 paquetes de datos erróneos, debido a que el algoritmo lo predijo como negativo, cuando en realidad era positivo.

A continuación, se presenta los resultados del entrenamiento conjunto de datos, según los algoritmos clasificadores.

Tabla 4 Resultados de rendimiento según Algoritmo Clasificador

Algoritmo Clasificador	Exactitud	Precisión	Exhaustividad	F-SCORE
Leaky Bucket	87.4%	94.3%	84.1%	88.9%
Token Bucket	89.8%	96.9%	85.8%	91%

Según los datos que se visualizan en la tabla de rendimiento, se puede observar que el algoritmo de Token Bucket tiene mejores indicadores de rendimiento que el algoritmo Leaky Bucket para la detección de Ataques DDoS. Esto debido a que el algoritmo Token Bucket puedes predecir con mayor eficiencia los ataques DDoS.

En los datos obtenidos en la tabla de rendimiento, la precisión mide el porcentaje de paquetes de datos que se predijeron de manera correcta, y la exhaustividad mide el porcentaje de paquetes de falsos que se predijeron de forma correcta. Para evaluar la eficiencia del algoritmo Token Bucket se debe analizar la precisión y la exhaustividad, esto por lo general se ve como un conflicto, debido a que cuando se mejora la exhaustividad, produce que se reduzca la precisión. Por tal motivo, existe el F-SCORE, que calcula la media entre la precisión y la exhaustividad, y cuando F-SCORE alcanza su mejor valor en 1, significa una perfecta precisión y exhaustividad.

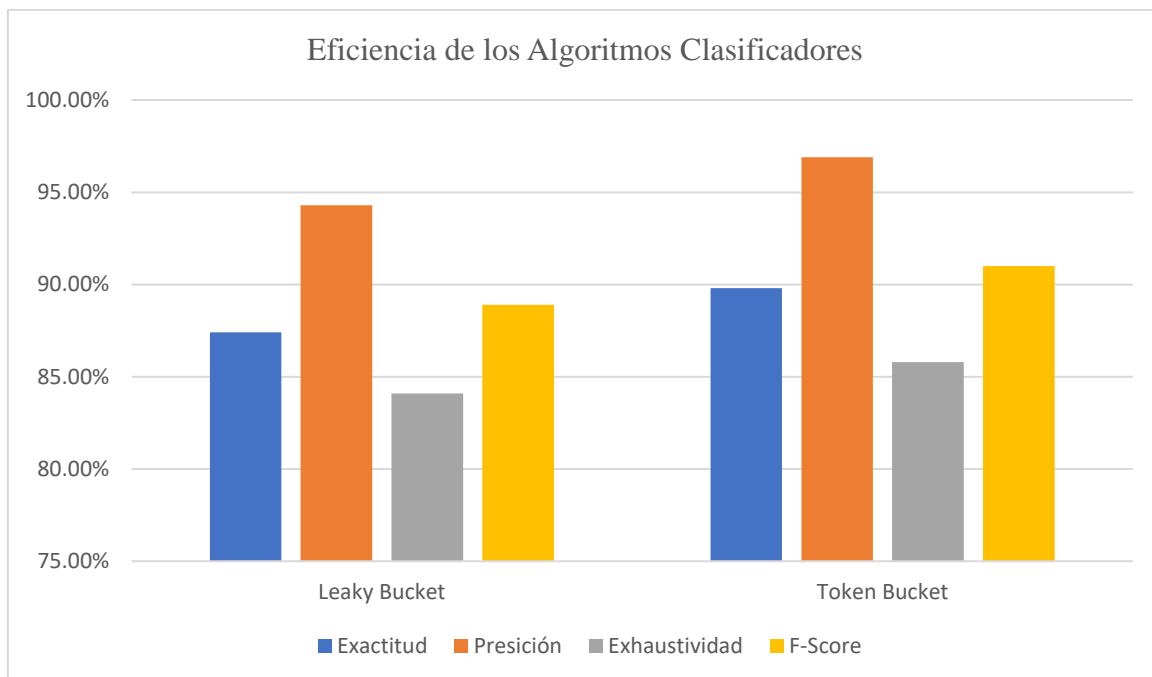


Figura 9 Tabla 6 Eficiencia de los Algoritmos Clasificadores

Fuente: Elaboración Propia

Los datos muestran que la eficiencia del algoritmo Token Bucket es superior a la del algoritmo Leaky Bucket. Se encontró una diferencia de 2.1% entre el F-Score del algoritmo Token Bucket y Leaky Bucket. En la exhaustividad se encontró una diferencia de 1.7%. En la precisión se encontró una diferencia de 2.6% y en la exactitud se encontró una diferencia de 2.5%.

3.2. Discusión de resultados.

Con los 494 registros obtenidos para realizar esta investigación, se obtuvo que el algoritmo Token Bucket tiene una exactitud de 89.8%, una precisión de 96.9, una exhaustividad de 85.8% y un F-Score de 91%, frente a los resultados obtenidos del algoritmo Leaky Bucket que tienen una exactitud de 87.4%, una precisión de 94.3%, una exhaustividad de 84.1% y un F-Score de 88.9%.

En base a los resultados que se ha obtenido, podemos decir que el algoritmo Token Bucket es más eficiente que el algoritmo Leaky Bucket para nuestro caso de estudio en la Municipalidad de Santiago de Surco.

Aqeel Sahi, David Lai, Yan li y Mohammed Diykh en su investigación de un sistema eficiente de detección y prevención de ataques de inundación DDoS TCP en un entorno de nube, el algoritmo Token Bucket con otro sistema de clasificación de prevención que pasan los paquetes de datos para comprobar si están en una lista negra, obteniendo así una precisión del 98%, frente a la precisión del 96% que se ha obtenido usando solo el algoritmo token Bucket,

Por tal motivo podemos decir que el algoritmo Token Bucket solo no es suficiente para defenderse de los Ataques DDoS, por lo cual es necesario combinarlo con otro algoritmos o sistemas que permitan obtener un mejor resultado.

3.3. Aporte práctico.

Para el presente trabajo de investigación se seleccionó la Municipalidad de Santiago de Surco, como caso de estudio debido a que en la actualidad las municipalidades distritales de Lima son afectadas por los ataques de denegación de servicios distribuido por lo menos una vez al mes.

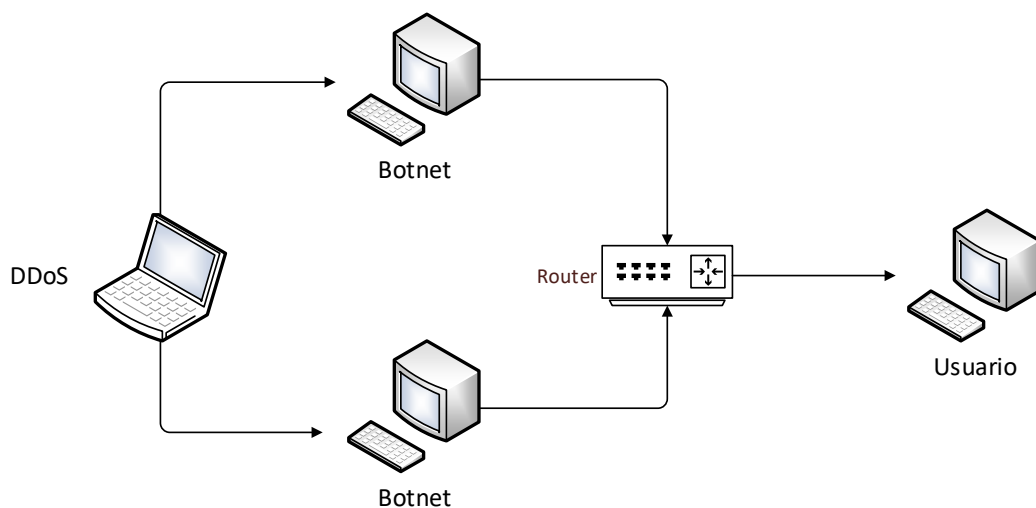
La Municipalidad de Santiago de Surco se seleccionó, ya que se cuenta con acceso a la información de dicha entidad y el número de ataques de ataques DDoS mensuales que recibe es superior a otros municipios, y es la única municipalidad en Lima que cuenta con su propia red de Fibra Óptica en todo el distrito.

Municipalidades	Acceso a la Información	Número de Ataques Mensual	Tipo de RED
Santiago de Surco	SI	3	Fibra
Surquillo	NO	1	Coaxial
Lince	NO	1	Coaxial
San Juan de Miraflores	NO	1	Coaxial

Tabla 5 Selección de Municipalidades

Se preparó un escenario de prueba local para simular la estructura de red un ataque DDoS hacia el caso de estudio en la Municipalidad de Santiago de Surco, donde se tendrá una laptop, la cual se usó por motivos de movilidad para simular un ataque DDoS desde cualquier ambiente. La laptop va a controlar 2 PCs que van a actuar como Botnet en la simulación de los ataques DDoS, debido a que es común el uso de los PC de usuarios para enviar estos ataques.

- 1 El escenario de prueba va a tener un switch router que va a simular el switch principal de la Municipalidad de Santiago de Surco, en el cual se va a conectar con un cable de red, una PC de usuario para ejecutar y analizar la eficiencia del algoritmo Token Bucket.



2

Figura 10 Escenario de Prueba

Fuente: Elaboración Propia

El algoritmo de token bucket maneja una tasa flexible de tráfico, regulando los paquetes entrantes sin descartarlos, y almacenándolos en búfer expansible hasta que encuentre tokens disponibles, y se puede aplicar en tiempo real.

El algoritmo Leaky Bucket, no se puede aplicar en tiempo real, por lo que solo se puede hacer el análisis luego de que se guardó todo el tráfico realizado. También se debe tener en cuenta que una vez lleno el búfer, este procede a descartar todos los paquetes adicionales.

Debido a la flexibilidad del búfer y capacidad de aplicarlo en tiempo real, se eligió el algoritmo Tocket Bucket para mitigar los ataques DDoS, debido a que tiene más ventajas que el algoritmo Leaky Bucket.

Tabla 6 Comparación de Algoritmos

Descripción	Leaky Bucket	Token Bucket
Parámetros Fundamentales	<ul style="list-style-type: none"> Tasa de Datos Numero de paquetes 	<ul style="list-style-type: none"> Tasa de generación de tokens Tamaño del Búfer
Tipo de trafico	Tasa Fija	Tasa Flexible
Comportamiento cuando se llena el búfer	Descarta Paquetes	Descarta tokens pero no paquetes
Aplicable en tiempo real	NO	SI

En el escenario de prueba propuesto primero hemos utilizado el Software Wireshark para capturar el tráfico que se va a analizar, obteniendo así un total de 494 paquetes, que van a hacer analizados con el algoritmo Token Bucket.

Nro	IP Origen	IP Destino	Protocolo	Información
1	192.168.8.111	172.217.192.94	UDP	57190 → 443 Len=122
2	172.217.192.94	192.168.8.111	UDP	443 → 57190 Len=183
3	172.217.192.94	192.168.8.111	UDP	443 → 57190 Len=25
4	192.168.8.111	172.217.192.94	UDP	57190 → 443 Len=33
5	192.168.8.111	3.230.129.93	TLSv1.2	Application Data

6	192.168.8.111	3.230.129.93	TCP	50696 → 443 [ACK] Seq=82 Ack=1 Win=512 Len=1400 [TCP segment of a reassembled PDU]
7	192.168.8.111	3.230.129.93	TCP	50696 → 443 [ACK] Seq=1482 Ack=1 Win=512 Len=1400 [TCP segment of a reassembled PDU]
8	192.168.8.111	3.230.129.93	TLSv1.2	Application Data
9	192.168.8.111	3.230.129.93	TLSv1.2	Application Data
10	3.230.129.93	192.168.8.111	TCP	443 → 50696 [ACK] Seq=1 Ack=1482 Win=68 Len=0
11	3.230.129.93	192.168.8.111	TCP	443 → 50696 [ACK] Seq=1 Ack=2965 Win=68 Len=0
12	3.230.129.93	192.168.8.111	TLSv1.2	Application Data
13	192.168.8.111	3.230.129.93	TCP	50696 → 443 [ACK] Seq=3977 Ack=272 Win=511 Len=0
14	192.168.8.111	69.171.250.60	TLSv1.2	Application Data
15	69.171.250.60	192.168.8.111	TCP	443 → 50173 [ACK] Seq=1 Ack=32 Win=841 Len=0
16	69.171.250.60	192.168.8.111	TLSv1.2	Application Data
17	192.168.8.111	69.171.250.60	TCP	50173 → 443 [ACK] Seq=32 Ack=39 Win=511 Len=0
18	172.217.192.189	192.168.8.111	UDP	443 → 57822 Len=44
19	192.168.8.111	172.217.192.189	UDP	57822 → 443 Len=33
20	92.223.66.31	192.168.8.111	TCP	80 → 50096 [ACK] Seq=1 Ack=1 Win=501 Len=0
21	192.168.8.111	92.223.66.31	TCP	[TCP ACKed unseen segment] 50096 → 80 [ACK] Seq=1 Ack=2 Win=512 Len=0
22	192.168.8.111	157.240.14.15	TLSv1.2	Application Data
23	157.240.14.15	192.168.8.111	TLSv1.2	Application Data
24	192.168.8.111	157.240.14.15	TCP	50583 → 443 [ACK] Seq=54 Ack=31 Win=514 Len=0
25	192.168.8.111	3.230.129.93	TCP	50696 → 443 [FIN, ACK] Seq=3977 Ack=272 Win=511 Len=0
26	3.230.129.93	192.168.8.111	TCP	443 → 50696 [FIN, ACK] Seq=272 Ack=3978 Win=68 Len=0

27	192.168.8.111	3.230.129.93	TCP	50696 → 443 [ACK] Seq=3978 Ack=273 Win=511 Len=0
28	192.168.8.111	45.57.90.1	TCP	50713 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
29	192.168.8.111	3.230.129.93	TCP	50714 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
30	192.168.8.111	107.167.110.216	TCP	50715 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
31	45.57.90.1	192.168.8.111	TCP	443 → 50713 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1400 WS=512 SACK_PERM=1
32	192.168.8.111	45.57.90.1	TCP	50713 → 443 [ACK] Seq=1 Ack=1 Win=131584 Len=0
33	192.168.8.111	45.57.90.1	TLSv1.3	Client Hello
34	192.168.8.111	192.168.8.1	DNS	Standard query 0xe649 A occ-0-3596- 185.1.nflxso.net
35	3.230.129.93	192.168.8.111	TCP	443 → 50714 [SYN, ACK] Seq=0 Ack=1 Win=35844 Len=0 MSS=1400 SACK_PERM=1 WS=512
36	192.168.8.111	3.230.129.93	TCP	50714 → 443 [ACK] Seq=1 Ack=1 Win=131584 Len=0
37	192.168.8.111	192.168.8.1	DNS	Standard query 0xe649 A occ-0-3596- 185.1.nflxso.net
38	192.168.8.111	3.230.129.93	TLSv1.3	Client Hello
39	192.168.8.1	192.168.8.111	DNS	Standard query response 0xe649 A occ-0- 3596-185.1.nflxso.net A 186.160.209.6
40	192.168.8.1	192.168.8.111	DNS	Standard query response 0xe649 A occ-0- 3596-185.1.nflxso.net A 186.160.209.6
41	192.168.8.111	54.207.37.84	TLSv1.2	Application Data
42	107.167.110.216	192.168.8.111	TCP	443 → 50715 [SYN, ACK] Seq=0 Ack=1 Win=42340 Len=0 MSS=1400
43	192.168.8.111	107.167.110.216	TCP	50715 → 443 [ACK] Seq=1 Ack=1 Win=64240 Len=0
44	192.168.8.111	107.167.110.216	TLSv1.2	Client Hello

45	3.230.129.93	192.168.8.111	TCP	443 → 50714 [ACK] Seq=1 Ack=532 Win=35328 Len=0
46	3.230.129.93	192.168.8.111	TLSv1.3	Server Hello, Change Cipher Spec, Application Data
47	54.207.37.84	192.168.8.111	TLSv1.2	Application Data
48	192.168.8.111	186.160.209.6	TCP	50716 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
49	192.168.8.111	186.160.209.6	TCP	50717 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1

Se presenta algunos paquetes que se han capturado con el Software Wireshark en la red propuesta, donde se puede observar las diversas consultas que se tiene al servidor.

Primero se analiza de forma manual todos los paquetes que están ingresando al servidor antes de ejecutar el algoritmo token bucket. Luego ejecutamos el algoritmo token bucket e identificamos el orden de ingreso y que paquetes están ingresando, también debemos validar que durante el ataque DDoS, el servicio web del servidor esté en funcionamiento.

Código del Algoritmo Token Bucket

```
from time import time

class TokenBucket(object):
    """An implementation of the token bucket algorithm.

    >>> bucket = TokenBucket(80, 0.5)
    >>> print bucket.consume(10)
    True
    >>> print bucket.consume(90)
    False
    """
    def __init__(self, tokens, fill_rate):
        """tokens is the total tokens in the bucket. fill_rate is the
        rate in tokens/second that the bucket will be refilled."""
        self.capacity = int(tokens)
        self._tokens = int(tokens)
        self.fill_rate = int(fill_rate)
        self.timestamp = time()

    def consume(self, tokens):
        """Consume tokens from the bucket. Returns True if there were
        sufficient tokens otherwise False."""
        if tokens <= self.tokens:
            self._tokens -= tokens
        else:
            return False
        return True
```

```

def get_tokens(self):
    now = time()
    if self._tokens < self.capacity:
        delta = int(self.fill_rate * (now - self.timestamp))
        self._tokens = int(min(self.capacity, self._tokens + delta))
    self.timestamp = now
    return self._tokens
tokens = property(get_tokens)

#####LLAMADA##

from time import sleep
import random

capacidad = 80
rate = 10
cola_total = 0
bucket = TokenBucket(capacidad, rate)

def generar_paquetes():
    paquetes_generados = 0
    global cola_total
    paquetes_gens = int(random.randint(40, 120))
    paquetes_generados = paquetes_gens
    print("Paquetes Entrantes: ",paquetes_generados)
    if(cola_total > 0):
        print("Paquetes procesados: ", bucket.tokens)
        cola_total = cola_total - bucket.tokens + paquetes_generados
        bucket.consume(bucket.tokens)

```

```

else:
    if(paquetes_generados <= bucket.tokens):
        print("Paquetes procesados.: ", paquetes_generados)
        bucket.consume(paquetes_generados)
    else:
        print("Paquetes procesados..: ", bucket.tokens)
        cola_total = cola_total - bucket.tokens + paquetes_generados
        bucket.consume(bucket.tokens)
def generar_token():
    sleep(1)
def resultados():
    print ("Tokens en buffer ==", bucket.tokens)
    print ("Cola de paquetes ===", cola_total)
print("##### DATOS INICIALES #####")
print ("Capacidad de Buffer =", capacidad)
print ("Rate =", rate)
print ("Tokens en Buffer Inicial =", bucket.tokens)
print("##### GENERAR Y PROCESAR PAQUETES #####")
generar_paquetes()
resultados()
print("##### GENERAR TOKEN #####")
generar_token()
print ("Tokens Generados =", rate)
resultados()
print("##### GENERAR Y PROCESAR PAQUETES #####")
generar_paquetes()
resultados()
print("##### GENERAR TOKEN #####")
generar_token()
print ("Tokens Generados =", rate)
resultados()

```

Fuente: Elaboración Propia

Luego de ejecutar el algoritmo Token Bucket, volvemos a capturar el tráfico con la herramienta Wireshark para validar los paquetes que han sido descartados.

IV. CONCLUSIONES Y RECOMENDACIONES

4.1. Conclusiones.

Se concluye que la Municipalidad de Santiago de Surco, es la mejor opción como caso de estudio debido a las facilidades de acceso a la información y los equipos de comunicaciones para la implementación del escenario de prueba.

En base a los estudios realizados en los artículos de investigación, se valida que el escenario de prueba que se utilizó para el presente proyecto es la más práctica para poder medir la eficiencia del Algoritmo Token Bucket en el caso de estudio.

Los datos capturados en la red por medio del Software Wireshark, permitieron poder medir la eficiencia del Algoritmo en la Municipalidad de Santiago de Surco.

Una vez analizado el tráfico con el algoritmo Token Bucket, haciendo la simulación de un ataque DDoS, se puede observar que el algoritmo le permite dar continuidad al servicio web que ejecuta el servidor, pero debido a la cantidad de paquetes que tiene que procesar por el ataque DDoS, la respuesta del servidor es considerablemente alto a lo que comúnmente se demora.

Así mismo, se hizo la misma prueba con el algoritmo Leaky Bucket, donde se puede observar que este algoritmo descarta los paquetes, y por ende algunas de las consultas que se hacen al servidor web, nunca obtienen respuesta, por tal motivo, este algoritmo, no se considera eficiente para un ataque DDoS.

4.2. Recomendaciones.

Se recomienda analizar el tráfico de red de forma manual antes usar el algoritmo token bucket para saber los paquetes que están ingresando, y también analizar los paquetes que pasan el filtro del algoritmo token bucket para determinar que paquetes son los que están siendo aceptados.

También se recomienda hacer uso de otro algoritmo de Clasificación que pueda acompañar al Algoritmo Token Bucket para ayudar a aumentar la eficiencia del mismo.

REFERENCIAS.

- Altarade, M. (2020). *The Definitive Guide to NoSQL Databases*. Obtenido de Developers: <https://www.toptal.com/database/the-definitive-guide-to-nosql-databases>
- R. Jansen, F. T. (2014). The Sniper Attack: Anonymously Deanonimizing and Disabling the Tor Network. *Proc. of the 18th Symposium on Network and Distributed System Security (NDSS)*.
- T. Peng, C. L. (2007). Survey of network-based defense mechanisms countering the DoS and DDoS problems. *ACM Computing Surveys*, 1-42.
- Caicedo, E., & López, J. (2017). *Una aproximación práctica a la redes neuronales artificiales*. Santiago de Cali, Colombia: Programa Editorial Universidad del Valle.
- Carmona, E. (2016). *Máquinas de Vectores Soporte (SVM)*. Madrid, España.
- Chapman, C. (5 de Marzo de 2020). *The Daily Swig*. Obtenido de The Daily Swig: <https://portswigger.net/daily-swig/high-severity-regex-bugs-discovered-in-parse-server>
- Curphey, M., & Groves, D. (2017). *OWASP*. Obtenido de OWASP: <https://wiki.owasp.org/images/5/5e/OWASP-Top-10-2017-es.pdf>
- Cuzzocrea, A., & Shahriar, H. (2017). Data masking techniques for NoSQL database security: A systematic review. *IEEE Xplore*, 2-7. doi:10.1109/BigData.2017.8258486

- ElevenPaths. (2014). *Como funciona las MongoDB Injection*. Obtenido de <https://empresas.blogthinkbig.com/como-funcionan-las-mongodb-injection/>
- Fernández, R., Seijo, R., Suárez, P., & Martínez, R. (2016). Statistical Model for Prediction of Diabetic Foot Disease in Type 2 Diabetic Patients. *Medisur*, 10. Obtenido de <http://www.medisur.sld.cu/index.php/medisur/article/view/3151/1985>
- Guamán, F. (2014). *Vulnerabilidades de Bases de Datos NoSQL*. Madrid: UPM.
- Hernández, A. (2013). La recolección y análisis de datos cuantitativos . *UNY*, 3-16.
- Islam, R., Islam, S., Ahmed, Z., Iqbal, A., & Shahriyar, R. (2019). Automatic Detection of NoSQL Injection Using Supervised Learning. *IEEE Xplore*. doi:10.1109/COMPSAC.2019.00113
- Islam, R., Islam, S., Ahmed, Z., Iqbal, A., & Shahriyar, R. (2019). Automatic Detection of NoSQL Injection Using Supervised Learning. *2019 IEEE 43rd Annual Computer Software and Applications Conference (COMPSAC)*, 760-769. doi:10.1109/COMPSAC.2019.00113
- Joyanes, L. (2016). Big Data, Análisis de grandes volúmenes de datos en organizaciones. *Alfaomega*. Obtenido de <https://books.google.com.pe/books?id=1GywDAAAQBAJ&pg=PT215&dq=tipos%20de%20almacenamiento%20de%20datos%20NoSQL&hl=es-419&sa=X&ved=2ahUKEwjG-q6zpvvqAhVPI7kGHXiMCx4Q6AEwA3oECAAQAg#v=onepage&q=tipos%20de%20almacenamiento%20de%20datos%20NoSQL&f=false>
- Kang, M., & Jameson, N. (2018). *Machine Learning*. Maryland, Estados Unidos: Sons Ltd.
- Lan, H., & Pan, Y. (2019). A Crowdsourcing Quality Prediction Model Based on. *IEEE/ACIS*, 315-319. doi:10.1109/ICIS46139.2019.8940306
- Little, J. (2020). *MathWorks*. Obtenido de <https://www.mathworks.com/help/phased/examples/detector-performance-analysis-using-roc-curves.html>
- Macqueen, J. (2008). SOME METHODS FOR CLASSIFICATION AND ANALYSIS OF MULTIVARIATE OBSERVATIONS. *projecteuclid*, 281-297. Obtenido de <https://projecteuclid.org/euclid.bsm/1200512992>
- Mauny, I. (24 de Enero de 2019). *APISECURITY*. Obtenido de *APISECURITY*: <https://apisecurity.io/issue-15-fortnite-hack-tls-mitm-attacks-sql-injections-for-nosql/>
- Mulling, C. (10 de Octubre de 2010). *Database Trends and Applications*. Obtenido de <https://www.dbta.com/Columns/DBA-Corner/Defining-Database-Performance-70236.aspx>
- El Peruano. (6 de 6 de 2019). *El Peruano*. Obtenido de El Peruano: <https://elperuano.pe/noticia-ciber-seguridad-data-centers-80284.aspx>

- Ron, A., Bronshtein, E., & Shulman-Peleg, A. (2015). No SQL, No Injection? Examining NoSQL Security. *ResearchGate*. Obtenido de https://www.researchgate.net/publication/278332363_No_SQL_No_Injection_Examining_NoSQL_Security
- Segal, M. (2004). Machine Learning Benchmarks and Random Forest Regression. *eScholarship*, 2-15.
- Troyano, J., Díaz, V., & Enriquez, F. (10 de Junio de 2002). *researchgate*. Obtenido de [researchgate: https://www.researchgate.net/publication/267565005_Aplicacion_de_Modelos_de_Markov_y_Maquinas_SVM_al_Reconocimiento_de_Entidades](https://www.researchgate.net/publication/267565005_Aplicacion_de_Modelos_de_Markov_y_Maquinas_SVM_al_Reconocimiento_de_Entidades)
- Tutte, W. (2001). Graph Theory. *Combridge Mathematical Library*, 30. Obtenido de https://books.google.com.pe/books?id=uTGhooU37h4C&pg=PA30&redir_esc=y#v=onepage&q&f=false
- Wang, X., Lin, X., & Dang, X. (2020). *Supervised learning in spiking neural networks: A review of algorithms*. People's Republic of China.
- Ye, Q., Lin, B., & Li, Y.-J. (2005). *SENTIMENT CLASSIFICATION FOR CHINESE REVIEWS: A COMPARISON BETWEEN SVM AND SEMANTIC APPROACHES*. China: Proceedings of the Fourth International Conference on Machine Learning and Cybernetics.
- Zhang, K. (2019). A Machine Learning based Approach to Identify. *IEEE/ACM International Conference on Automated Software Engineering (ASE)*, 1286-1287. Obtenido de <https://ieeexplore.ieee.org/abstract/document/8952467>

ANEXOS.

Anexo 1. Resolución de aprobación del proyecto de investigación

FACULTAD DE INGENIERÍA, ARQUITECTURA Y URBANISMO
RESOLUCIÓN N°1317-2020/FIAU-USS

Pimentel, 17 de julio de 2020

VISTO:

El Acta de reunión N°1606-2020, de fecha 16 de junio de 2020 del Comité de investigación de la Escuela profesional de INGENIERIA DE SISTEMAS, para la ejecución : "EVALUACIÓN DE LA EFICIENCIA DEL ALGORITMO TOKEN BUCKET PARA MITIGAR LA DENEGACIÓN DE SERVICIOS DISTRIBUIDA EN MUNICIPALIDADES PERUANAS. CASO DE ESTUDIO MUNICIPALIDAD DE SANTIAGO DE SURCO", presentado por el(los) tesista(s) GONZALES GUEVARA ROMMEL ANDRES, del Programa de estudios INGENIERIA DE SISTEMAS, y;

CONSIDERANDO:

Que, de conformidad con la Ley Universitaria N° 30220 en su artículo 48º que a letra dice: "La investigación constituye una función esencial y obligatoria de la universidad, que la fomenta y realiza, respondiendo a través de la producción de conocimiento y desarrollo de tecnologías a las necesidades de la sociedad, con especial énfasis en la realidad nacional. Los docentes, estudiantes y graduados participan en la actividad investigadora en su propia institución o en redes de investigación nacional o internacional, creadas por las instituciones universitarias públicas o privadas.";

Que, de conformidad con el Reglamento de grados y títulos en su artículo 21º señala: "Los temas de trabajo de investigación, trabajo académico y tesis son aprobados por el Comité de Investigación y derivados a la facultad o Escuela de Posgrado, según corresponda, para la emisión de la resolución respectiva. El período de vigencia de los mismos será de días años, a partir de su aprobación. En caso un tema perdiera vigencia, el Comité de Investigación evaluará la ampliación de la misma.

Que, de conformidad con el Reglamento de grados y títulos en su artículo 24º señala: La tesis es un estudio que debe denotar rigurosidad metodológica, originalidad, relevancia social, utilidad teórica y/o práctica en el ámbito de la escuela profesional. Para el grado de doctor se requiere una tesis de máxima rigurosidad académica y de carácter original. Es individual para la obtención de un grado; es individual o en pares para obtener un título profesional. Asimismo, en su artículo 25º señala: "El tema debe responder a alguna de las líneas de investigación Institucionales de la USS S.A.C."

Que, en el Acta de reunión N°1606-2020 de fecha 16 de junio de 2020, del Comité de investigación de la Escuela profesional de INGENIERIA DE SISTEMAS, se indica entre los acuerdos la aprobación del Proyecto de tesis denominado "EVALUACIÓN DE LA EFICIENCIA DEL ALGORITMO TOKEN BUCKET PARA MITIGAR LA DENEGACIÓN DE SERVICIOS DISTRIBUIDA EN MUNICIPALIDADES PERUANAS. CASO DE ESTUDIO MUNICIPALIDAD DE SANTIAGO DE SURCO" de la línea de investigación de INFRAESTRUCTURA, TECNOLOGÍA Y MEDIO AMBIENTE, a cargo de GONZALES GUEVARA ROMMEL ANDRES en condición de estudiante, del Programa de estudios INGENIERIA DE SISTEMAS.

Estando a lo expuesto, y en uso de las atribuciones conferidas y de conformidad con las normas y reglamentos vigentes;

SE RESUELVE:

ARTÍCULO 1º: APROBAR, el Proyecto de denominado "EVALUACIÓN DE LA EFICIENCIA DEL ALGORITMO TOKEN BUCKET PARA MITIGAR LA DENEGACIÓN DE SERVICIOS DISTRIBUIDA EN MUNICIPALIDADES PERUANAS. CASO DE ESTUDIO MUNICIPALIDAD DE SANTIAGO DE SURCO", perteneciente a la línea de investigación de INFRAESTRUCTURA, TECNOLOGÍA Y MEDIO AMBIENTE, a cargo de GONZALES GUEVARA ROMMEL ANDRES, del Programa de estudios INGENIERIA DE SISTEMAS.

ARTÍCULO 2º: ESTABLECER, que la inscripción del Título de Proyecto de tesis se realice a partir de emitida la presente resolución y tendrá una vigencia de dos (02) años.

ARTÍCULO 3º: DEJAR SIN EFECTO, toda Resolución emitida por la Facultad que se oponga a la presente Resolución.

REGÍSTRESE, COMUNÍQUESE Y ARCHÍVESE



Dr. María Mercedes Pimentel
Rector - Facultad de Ingeniería,
Arquitectura y Urbanismo
UNIVERSIDAD SURCO DE LIMA S.A.C.



M.A. María Mercedes López
Directora Académica / Facultad de Ingeniería,
Arquitectura y Urbanismo
UNIVERSIDAD SURCO DE LIMA S.A.C.

Cc: Interesado, Archivo

Anexo 2. Carta de aceptación de la institución para la recolección de datos.

Municipalidad de Santiago de Surco
Gerencia de Tecnologías de la Información.

"Año de la Universalización de la Salud"

Santiago de Surco, 17 de Diciembre del 2020

Carta N° 003 – 2020 – GTI – MSS

Ing. Heber Iván Mejía Cabrera
Director (e) de la Universidad SEÑOR DE SIPAN S.A.C.
Ciudad.-

Santiago de Surco.-

Asunto: Trabajo de Investigación

Tengo el agrado de dirigirme a Usted para saludarle y en atención al asunto de la referencia, hacemos de su conocimiento que en atención a lo solicitado por su Institución, el Sr. GONZALES GUEVARA ROMMEL ANDRES identificado con DNI N° 47006712, cuenta con nuestra autorización para el recojo de información relevante que será usado para la realización de su trabajo de Investigación "EVALUACION DE LA EFICIENCIA DEL ALGORITMO TOKEN BUCKET PARA MITIGAR LA DENEGACION DE SERVICIOS DISTRIBUIDA EN MUNICIPALIDADES PERUANAS".

Es todo cuanto informo para los trámites administrativos correspondientes.

Atentamente,

Municipalidad de Santiago de Surco

.....
Carlos Espinoza Alegria
Gerente de Tecnologías de la Información

CEM nr

Anexo 3. FORMATO PARA EL REGISTRO DE MATRIZ DE CONFUSIÓN.

		Resultados Clasificación	
		Paquete Positivo	Paquete Positivo
Resultados Reales	Paquete Positivo		
	Paquete Positivo		

Ítem	Valor
Verdadero positivo (VP)	
Falso Positivo (FP)	
Verdadero Negativo (VN)	
Falso Negativo (FN)	

Ítem	Valor
Exactitud	
Precisión	
Exhaustividad	
F-Score	
AUC	

Verdadero Positivo (VP)		Falso Positivo (FP)	
Ítem	Valor	Ítem	Valor
Realidad		Realidad	
Predicción del modelo AA		Predicción del modelo AA	
Numero de resultados		Numero de resultados	
Falso Negativo (FN)		Verdadero Negativo (FP)	
Ítem	Valor	Ítem	Valor
Realidad		Realidad	
Predicción del modelo AA		Predicción del modelo AA	
Numero de resultados		Numero de resultados	

Anexo 4. FORMATO PARA INFORME DE CONSUMO DE CPU.

Consumo de CPU	
Ítem	Valor
Uso	
Velocidad	
Procesos	
Subprocesos	
Tiempo	

Anexo 5. FORMATO DE INFORME DE CONSUMO DE MEMORIA.

Consumo de Memoria	
Ítem	Valor
Uso	
Disponibilidad	
Confirmada	
En caché	
Tiempo	

Anexo 6. FORMATO PARA INFORME DE PROMEDIO DE TIEMPO DE RESPUESTA.

Promedio de Tiempo de respuesta	
Ítem	Valor
Velocidad	
Tiempo	
CPU	
Memoria	
Disco	