UNIVERSIDAD DE BURGOS
ESCUELA POLITÉCNICA SUPERIOR
Grado en Ingeniería Informática

**TFG del Grado en Ingeniería Informática**

# GII-24.19 Contramedidas IoT mediante Reinforcement Learning

Presentado por César Rodríguez Villagrá
en Universidad de Burgos — 11 de junio de 2025
Tutores: Rubén Ruiz González y Nuño Basurto Hornillos

UNIVERSIDAD DE BURGOS
ESCUELA POLITÉCNICA SUPERIOR
Grado en Ingeniería Informática

D. Rubén Ruiz González, profesor del Departamento de Digitalización, y Área de Ingeniería de Sistemas y Automática.
D. Nuño Basurto Hornillos, profesor del Departamento de Digitalización, y Área de Ciencia de la Computación e Inteligencia Artificial.

Exponen:

Que el alumno D. César Rodríguez Villagrá, con DNI 71970000X, ha realizado el Trabajo Final de Grado en Ingeniería Informática titulado "Contramedidas IoT mediante reinforcement learning".

Y que dicho trabajo ha sido realizado por el alumno bajo la dirección de quienes suscriben, en virtud de lo cual se autoriza su presentación y defensa.

En Burgos, 11 de junio de 2025

| Vº. Bº. del Tutor: | Vº. Bº. del co-tutor: |
| --- | --- |
| Rubén Ruiz González | D. Nuño Basurto Hornillos |

## Resumen

La ciberseguridad en los dispositivos del Internet de las Cosas (IoT) es un tema de creciente importancia debido a la masificación de estos dispositivos en nuestra vida diaria. Cuanto más se extiende el uso de IoT, más vulnerabilidades y amenazas surgen, lo que hace necesario desarrollar sistemas de seguridad adecuados para mitigar los ataques que puedan recibir, en los que los más comunes son los ataques de denegación de servicio (DoS) y denegación de servicio distribuido (DDoS). Estos ataques pueden afectar gravemente la disponibilidad y funcionalidad de los dispositivos IoT, provocando tiempos de respuesta lentos o incluso el colapso de los sistemas. Para abordar este problema, el proyecto propone un sistema de ciberseguridad que utiliza técnicas de aprendizaje por refuerzo para mitigar estos ataques en tiempo real. El aprendizaje por refuerzo es un enfoque de aprendizaje automático que permite adaptarse a entornos dinámicos y aprender de la experiencia, pudiendo incluso adaptarse a nuevas amenazas que puedan surgir. El agente aprende a minimizar los paquetes descartados mientras intenta maximizar el tráfico que transmite. Los resultados obtenidos mediante simulación muestran que el sistema es capaz de reducir significativamente el impacto de los ataques DoS y DDoS en los dispositivos IoT, mejorando así la seguridad y la disponibilidad de estos sistemas en comparación con enfoques tradicionales.



## Descriptores

Aprendizaje por refuerzo, Internet de las cosas, Ciberseguridad, Contramedidas, Aprendizaje automático, Web, Seguridad, Redes informáticas, Despliegue en la nube, Mitigación de ataques.

## Abstract

Cybersecurity in Internet of Things (IoT) devices has become increasingly critical due to the rapid proliferation of these devices across diverse applications. The expanding use of IoT introduces new vulnerabilities and threats, notably Denial of Service (DoS) and Distributed Denial of Service (DDoS) attacks, which can severely compromise the availability and functionality of IoT systems.

This project presents a cybersecurity framework based on reinforcement learning to mitigate DoS and DDoS attacks in real time. Reinforcement learning enables the system to adapt to dynamic environments and learn from experience, facilitating the detection and response to both known and emerging threats. The proposed agent is trained to minimize packet loss while maximizing legitimate network throughput. Experimental results, obtained through simulation, demonstrate that the system can significantly reduce the impact of DoS and DDoS attacks on IoT devices, resulting in improved security and system availability compared to traditional approaches.



## Keywords

Reinforcement Learning, Internet of Things (IoT), Cybersecurity, Countermeasures, Machine Learning, Web Technologies, Security, Computer Networks, Cloud Deployment, Attack Mitigation.

# Índice general

# Índice de figuras

# Índice de tablas

# 1. Introducción

En la actualidad, el uso masivo de dispositivos conectados a Internet ha permitido a los cibercriminales llevar a cabo ataques informáticos de forma más sencilla y efectiva. Estos ataques pueden tener consecuencias graves, como el robo de datos, la interrupción de servicios, el secuestro de sistemas, etc. Por lo que es fundamental tener medidas de seguridad adecuadas para proteger todo sistema informático.

Debido al crecimiento y desarrollo de nuevas tecnologías, ha provocado un aumento en la frecuencia, automatización y complejidad de los ciberataques [9]. Según el informe de IBM Security, el coste medio global de una brecha de datos alcanzó los 4,9 millones de dólares. Por otro lado, el coste ahorrado en organizaciones que usan seguridad con inteligencia artificial y una automatización extensa en la seguridad es de 2,2 millones de dólares en promedio [1]. Esto ha generado una necesidad de invertir en la investigación y el desarrollo de nuevas tecnologías para mejorar la seguridad, garantizando el correcto funcionamiento de los sistemas a la par que seguros. También ha generado que todos los profesionales que trabajan con sistemas informáticos estén concienciados de la importancia de la seguridad informática y de la necesidad de seguir e implementar buenas prácticas de seguridad en todos los procesos.

En los últimos años, los dispositivos de Internet de las Cosas (IoT) se han vuelto cada vez más populares, tanto en el ámbito empresarial como en el doméstico. Estos dispositivos, que van desde cámaras de seguridad hasta termostatos inteligentes, ofrecen una gran comodidad y eficiencia en la recolección y gestión de datos. Sin embargo, también son dispositivos muy

[1] IBM Security. (2024). *Cost of a Data Breach Report 2024*. https://www.ibm.com/reports/data-breach

vulnerables a ciberataques, ya que muchos de ellos carecen de medidas de seguridad adecuadas, por lo que es aún más importante implementar buenas prácticas de seguridad en estos dispositivos.

Otro tema que está en auge en la actualidad es el uso de la Inteligencia Artificial (IA) en múltiples áreas, incluyendo la ciberseguridad. La IA puede ayudar a detectar patrones, predecir amenazas e incluso mitigar ataques en tiempo real.

Se ha extendido el uso del aprendizaje por refuerzo para todo tipo de temas [11], incluyendo la robótica, el control autónomo de vehículos y, más recientemente, la ciberseguridad. Este tipo de aprendizaje automático permite a los sistemas realizar acciones basadas en lo que ha aprendido y en una retroalimentación del entorno, convirtiéndolo en un enfoque adecuado para un entorno tan complejo y dinámico como es la ciberseguridad, un entorno que cambia constantemente y con ataques muy diversos. En lugar de aprender solo de datos históricos, permite adaptarse en tiempo real a nuevas situaciones, optimizando su política de decisión, y ante amenazas desconocidas, permite adaptarse y aprender de ellas. Esta capacidad de adaptación posiciona al aprendizaje por refuerzo como una herramienta clave para la ciberseguridad.

Por lo que en este trabajo se explorará cómo la IA, mediante el aprendizaje por refuerzo, puede mejorar la seguridad de los sistemas informáticos, especialmente en el contexto de los dispositivos IoT.

## 1.1. Estructura del trabajo

La documentación se ha dividido en 2 partes principales:

- **Memoria.** En esta parte se presenta y describe el trabajo realizado, incluyendo la motivación, objetivos, metodología y resultados obtenidos.
- **Anexos.** Esta parte se centra más en los aspectos técnicos del trabajo del desarrollo del software y elementos importantes en la gestión del proyecto.

Todo el trabajo realizado está disponible en el repositorio de GitHub del proyecto que es: https://github.com/CesarRodrigu/GII-24.19-contramedidas-IoT-mediante-reinforcement-learning.

También se ha creado un sitio web para el proyecto. El sitio web es: https://cesarrv.com.

Para el control de calidad del software se ha utilizado SonarQube, una herramienta de análisis de código estático que permite detectar problemas de calidad en el código. El informe de SonarQube se puede consultar en el siguiente enlace: https://sonarcloud.io/project/overview?id=CesarRodrigu_GII-24.19-contramedidas-IoT-mediante-reinforcement-learning.

# 2. Objetivos del proyecto

El objetivo general del proyecto es el desarrollo de un agente de aprendizaje por refuerzo capaz de mitigar ataques de denegación de servicio basados en la ocupación de la cola de un router de la red, en un entorno de dispositivos de internet de las cosas. Para ello, se ha desarrollado un entorno de simulación que permite la generación de tráfico benigno y malicioso, así como el entrenamiento del agente. Además, se ha desarrollado una aplicación web que permite la gestión de los modelos entrenados por cada usuario, mostrar la documentación asociada y visualizar los resultados obtenidos por el agente, junto con la visualización de la función de recompensa y el estado del entorno.

Los principales objetivos del proyecto son:

- Implementar un entorno de simulación para el entrenamiento del agente.
- Desarrollar un agente de Inteligencia Artificial, mediante aprendizaje por refuerzo capaz de detectar y mitigar ataques a una red informática, especialmente en un entorno IoT.
- Desarrollar y desplegar una aplicación web que permita la visualización de los resultados obtenidos por el agente.

## 2.1. Objetivos específicos

Los objetivos más específicos del proyecto son:

## Generación de los datos de entrenamiento

El objetivo de esta parte del proyecto es la generación artificial de generadores de paquetes para simular tráfico benigno y tráfico atacante en un entorno de dispositivos de internet de las cosas, así como la creación de un entorno de simulación para el entrenamiento del agente de aprendizaje por refuerzo. Los requisitos específicos de este apartado son:

- Desarrollar generadores de tráfico benigno representativo de dispositivos IoT basado en datos realistas.
- Implementar generadores de tráfico malicioso que simulen ataques de denegación de servicio basado en datos realistas.
- Realizar un estudio de la función de recompensa para su optimización, en el que se muestre la evolución de la recompensa según la acción y el estado del entorno.
- Usar los datos generados en un entorno simulación que permita el entrenamiento y la evaluación del agente.
- Realizar una evaluación gráfica de los resultados obtenidos por el agente.

## Desarrollo de la aplicación web

El propósito de este apartado es el desarrollo de una aplicación web para la gestión y visualización de los modelos y resultados. Los requisitos específicos son:

- Implementar una interfaz para la visualización de algunos los resultados obtenidos por el agente.
- Permitir la gestión y almacenamiento persistente de los modelos entrenados por los usuarios.
- Incluir la documentación asociada a cada modelo, accesible desde la aplicación.
- Facilitar la visualización de la función de recompensa y del estado del entorno de simulación.
- Permitir la gestión de los modelos almacenados por cada usuario.

- Permitir la gestión de los usuarios por parte del administrador.
- Proporcionar ayuda a los usuarios en la aplicación web para facilitar su uso.

## Otros objetivos

Además de los objetivos ya descritos, es importante destacar otros objetivos que se han tenido en cuenta durante el desarrollo del proyecto:

- **Seguridad:** Desarrollar el sistema siguiendo buenas prácticas de seguridad informática para proteger los datos y la integridad del sistema.
- **Reducción de costes:** Diseñar el sistema de forma que se minimicen los recursos requeridos y el costo asociado, especialmente para el despliegue de la aplicación web.
- **Usabilidad:** Diseñar una interfaz web intuitiva, accesible y simple para los usuarios, asegurando una experiencia de usuario satisfactoria.
- **Escalabilidad:** Permitir que el sistema pueda ser fácilmente ampliado o adaptado a nuevas necesidades o requisitos futuros.
- **Mantenibilidad:** Estructurar el código del proyecto de forma modular y documentada para facilitar su mantenimiento, extensión o reutilización en futuros desarrollos, siguiendo patrones de diseño y buenas prácticas de programación.

# 3. Conceptos teóricos

En este capítulo se explican los conceptos teóricos relacionados con el proyecto, necesarios para la comprensión del mismo.

## 3.1. Aprendizaje por refuerzo

El aprendizaje por refuerzo o *Reinforcement Learning* (RL) es una rama del aprendizaje automático cuyo objetivo es aprender cómo los agentes deben tomar decisiones en un entorno, para maximizar la recompensa asociada a sus acciones y estado del entorno. A diferencia del aprendizaje supervisado, donde se dispone de un conjunto de datos etiquetados, en el aprendizaje por refuerzo el agente aprende a través de la interacción con el entorno, recibiendo recompensas o penalizaciones en función de las acciones que realiza. Su aprendizaje se basa en la prueba y error, donde el agente aprende a través de la experiencia acumulada en el entorno. Los principales elementos del aprendizaje por refuerzo son:

- **Agente:** Es el ente que se encarga de la toma de decisiones y aprende a través de la interacción con el entorno a lo largo del tiempo. También se le puede llamar modelo.
- **Entorno:** Es el contexto en el que el agente se encuentra, incluyendo todos los elementos con los que se relaciona.
- **Estado:** Es una instantánea en un momento dado, que puede incluir información sobre el estado actual del agente y del entorno.
- **Acción:** Es una decisión tomada por el agente que afecta al estado del entorno.

- **Política:** Es una estrategia que define cómo el agente selecciona acciones en función del estado actual del entorno.
- **Recompensa:** Es una señal que indica al agente la calidad de la acción tomada en un estado determinado. El objetivo del agente es maximizar la recompensa acumulada a lo largo del tiempo.

El aprendizaje por refuerzo se utiliza en una amplia variedad de aplicaciones, como juegos, robótica, optimización de procesos, entre otros.

Se usa en situaciones donde:

- Las acciones afectan el estado futuro del entorno
- No se dispone de un conjunto de datos etiquetados
- Se tiene retroalimentación escasa o tardía de la acción que se ha tomado
- El agente debe aprender a través de la experiencia acumulada.

Es especialmente útil en entornos muy variables o dinámicos, donde se tienen que tomar acciones secuencialmente y se necesita una adaptación a los cambios en el entorno. En la Figura 3.1 se muestra cómo funciona el aprendizaje por refuerzo.

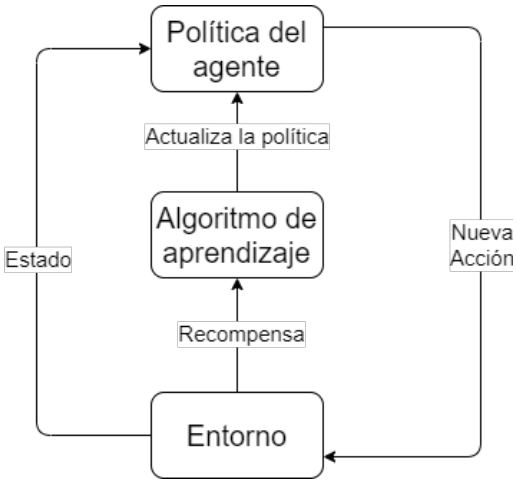


Figura 3.1: Diagrama explicativo del funcionamiento del aprendizaje por refuerzo

## 3.2. Redes de computadoras

Una red de computadoras es un conjunto de dispositivos interconectados que pueden comunicarse entre sí. Permiten el intercambio de mensajes y recursos entre los elementos de la red. Actualmente, las redes de computadoras están muy extendidas y son fundamentales para el funcionamiento de Internet y de muchas aplicaciones y servicios en línea. Las redes se pueden clasificar en diferentes tipos según su alcance, los más comunes son:

- **Redes de área local (LAN):** Conectan computadoras en un área geográfica limitada, como una oficina o un edificio.
- **Redes de área amplia (WAN):** Conectan computadoras en áreas geográficas más amplias, como ciudades o países.

Los elementos principales de una red de computadoras son:

- **Dispositivos finales:** Son los dispositivos que se conectan a la red, los cuales se quieren comunicar con otros dispositivos finales. Los dispositivos finales pueden ser computadoras, teléfonos móviles, impresoras, cámaras, sensores, etc.
- **Dispositivos de red:** Son los dispositivos que permiten la comunicación entre los dispositivos finales, como routers, switches, etc.
- **Medios de transmisión:** Son los canales a través de los cuales se transmiten la información entre dispositivos, como cables Ethernet, fibra óptica, Wi-Fi, etc.

La comunicación entre dispositivos se realiza a través de paquetes de datos que viajan por los medios de transmisión. Los paquetes incluyen información sobre el origen, destino, protocolo y contenido del mensaje, entre otros. Es importante tener en cuenta el tamaño de los paquetes, siendo necesario en algunas ocasiones dividir los mensajes en varios paquetes para su transmisión. Los protocolos de red son un pilar fundamental en la comunicación de computadoras en redes, ya que son un conjunto de reglas y convenciones que definen cómo se transmiten los datos entre estos. Algunos de los protocolos más comunes son:

- **HTTP (HyperText Transfer Protocol):** Es el protocolo que se usa par la transferencia de datos en internet. Permite la comunicación entre

navegadores web y servidores. Sin embargo, HTTP tiene problemas de seguridad, ya que los datos no se transmiten cifrados.

- **TLS (Transport Layer Security):** Es un protocolo criptográfico que proporciona comunicaciones seguras sobre una red [6], su predecesesor es la capa de puertos seguros o SSL. TLS cifra los datos antes de enviarlos y verifica que no hayan sido alterados durante la transmisión. Es ampliamente utilizado en protocolos como HTTPS, correo electrónico, y otros servicios en línea.
- **HTTPS (HTTP Secure):** Es la versión de transmisión de datos segura de HTTP [6]. Utiliza una capa adicional de seguridad llamada TLS (Transport Layer Security) para cifrar la información durante la transmisión. Esto garantiza la seguridad de los datos, protegiendo contra ataques como la interceptación o modificación de mensajes.
- **TCP/IP (Transmission Control Protocol / Internet Protocol):** Es el conjunto de protocolos más utilizado en Internet. TCP [15] se encarga de dividir los mensajes en paquetes, enviarlos y asegurarse de que lleguen completos y en orden. El protocolo TCP está hecho para ser utilizado con el IP, que se encarga de direccionar y enrutar los paquetes hacia su destino a través de la red.
- **IPv4 (Internet Protocol version 4) e IPv6 (Internet Protocol version 6):** Es la cuarta versión del protocolo Internet Protocol(IP) y la más utilizada. Utiliza direcciones de 32 bits. Debido al crecimiento de dispositivos conectados, se ha ido agotando el espacio de direcciones IPv4. IPv6 Es la versión más reciente del protocolo IP. Utiliza direcciones de 128 bits, lo que soluciona el problema de las direcciones limitadas IPv4. IPv6 también mejora aspectos como la eficiencia y configuración entre otros.
- **DNS (Domain Name System):** Es el sistema que traduce los nombres de dominio en direcciones IP, facilitando al usuario el acceso a los servidores que proporcionan la información solicitada. Principalmente permite que los usuarios accedan a sitios web usando nombres fáciles de recordar(dominios) en lugar de números IP difíciles de manejar.

En cuanto a los dispositivos con funciones de seguridad de red, los principales son:

- **Firewalls:** Son dispositivos hardware o software que controlan el tráfico de red, permitiendo o bloqueando el paso de paquetes según

reglas predefinidas [6]. Se utilizan para proteger la red de accesos no autorizados y ataques informáticos.

- **Proxies:** Como Cloudflare, son servidores que actúan como intermediarios entre los dispositivos finales y los servidores de destino [6]. Se utilizan para mejorar el rendimiento, la seguridad y la privacidad de la red.
- **CDN (Content Delivery Network):** Son redes de servidores distribuidos en distintas ubicaciones que almacenan copias del contenido estático (como imágenes, scripts etc) para entregarlo de forma más rápida y eficiente al usuario. Además de mejorar el rendimiento, muchas CDN también contribuyen a la seguridad al absorber tráfico malicioso. Cloudflare es un ejemplo de CDN que ofrece servicios de seguridad y rendimiento para aplicaciones web.

## 3.3. Interfaces de Programación de Aplicaciones

Una API (*Application Programming Interface*) es un conjunto de definiciones y protocolos que permiten que dos aplicaciones se comuniquen entre sí. Las APIs [4] actúan como una interfaz que facilita el acceso a funcionalidades o datos de un sistema sin necesidad de conocer su implementación interna. Una API REST (*Representational State Transfer*) es un tipo de API que sigue los principios de la arquitectura REST, siguiendo estos principios:

- **Cliente-servidor:** La API REST separa el cliente del servidor, permitiendo que ambos se desarrollen de forma independiente, pero manteniendo la comunicación entre ellos.
- **Sin estado:** Donde en el servidor no se almacenen datos del cliente entre solicitudes.
- **Cacheable:** Las respuestas de la API REST deben poderse guardar en caché, para mejorar el rendimiento y reducir la carga en el servidor.
- **Interfaz uniforme:** La API REST debe tener una interfaz consistente y fácil de entender, para facilitar su uso y mantenimiento.

Estas llamadas a la API REST se realizan a través de peticiones HTTP, utilizando los métodos más comunes del protocolo, como GET para obtener

datos del servidor, POST para enviar o crear datos, PUT para actualizar recursos existentes y DELETE para eliminar recursos. Existen otros métodos menos utilizados, pero estos son los más frecuentes.

## 3.4. Internet de las cosas

Los dispositivos de Internet de las cosas, o por sus siglas en inglés *Internet of Things* (IoT) [7], son dispositivos físicos que están conectados a una red y se encargan de enviar, recibir y procesar datos del mundo real.

Actualmente estos dispositivos están muy extendidos y son fundamentales para el funcionamiento de muchas aplicaciones y servicios en línea, como la domótica, la monitorización de la salud, la gestión de la energía, cámaras de seguridad, etc. Estos dispositivos se caracterizan por la escasa capacidad de procesamiento y almacenamiento, siendo también muy susceptibles a ataques informáticos debido a su baja seguridad. Un ataque muy común en estos dispositivos es el ataque de denegación de servicio, que busca saturar los recursos del dispositivo o de la red a la que está conectado, impidiendo su correcto funcionamiento, o incluso utilizando estos dispositivos para formar botnets [8].

## 3.5. Ataques informáticos

Los ataques informáticos son acciones que tienen como objetivo comprometer la seguridad de los sistemas, redes o datos. Estas acciones, cuyo objetivo es afectar a uno o varios de los tres pilares fundamentales de la ciberseguridad: la confidencialidad, integridad y disponibilidad (conocidos como el triángulo CIA) [2, 3].

Actualmente, los ataques informáticos son una amenaza constante para la seguridad de las redes y sistemas. Estos ataques pueden tener diferentes objetivos, como robar información, interrumpir servicios, entre otros.

- **Ataques de denegación de servicio (DoS y DDoS):** Buscan saturar los recursos de un sistema o red, impidiendo el acceso de los usuarios legítimos a los servicios. Denied of Service (DoS) se refiere a un ataque desde una sola fuente, mientras que Distributed Denied of Service (DDoS) implica la versión distribuida, atacando desde múltiples fuentes. Para ataques DDoS se suelen usar redes de bots o *botnets* [8], que son redes de dispositivos comprometidos que se controlan de forma

remota para llevar a cabo el ataque. Debido a la gran cantidad de dispositivos de internet de las cosas que se usan actualmente, y su baja seguridad, son un objetivo común para los atacantes que buscan crear redes de bots.

- **Malware:** Es un programa informático malicioso diseñado para dañar, robar o modificar la información o sistemas informáticos [6]. Incluye virus, gusanos, troyanos, ransomware, entre otros.

- **Exploits de vulnerabilidades:** Son técnicas o herramientas que aprovechan de fallos de seguridad [6] en el software o hardware para comprometer el sistema.

- **Ataques CSRF (Cross-Site Request Forgery):** CSRF es un tipo de ataque que fuerza a un usuario autenticado en el sistema para que realice acciones no deseadas. El atacante envía una solicitud al navegador del usuario, que se ejecuta sin su conocimiento, aprovechando la sesión activa del usuario.

- **Ataques XSS (Cross-Site Scripting):** XSS es un tipo de ataque que permite a un atacante incluir scripts maliciosos en una página web, que se ejecutan en el navegador de los usuarios que visitan esa página. Estos scripts pueden robar información del usuario, o realizar acciones no autorizadas en nombre del usuario.

Para que los ataques informáticos no tengan éxito, es importante implementar medidas de seguridad adecuadas y mitigar los ataques que se detecten en la medida de lo posible. Algunas de las mitigaciones más comunes son:

- **Descartar paquetes:** Se descartan los paquetes que se consideran basados en reglas establecidas [10, 20], por ejemplo, paquetes que no cumplen con el formato esperado, que provienen de ciertas direcciones IP, ubicaciones geográficas, protocolos etc.

- **Rate limiting:** Consiste en limitar la cantidad de peticiones que puede realizar un cliente en un intervalo de tiempo determinado. Esto permite evitar que el servidor se pueda saturar por un gran número de peticiones desde un mismo cliente, ya sea un cliente legítimo o un bot malicioso que realice escaneos masivos, ataques de denegación de servicio o ataques de fuerza bruta.

- **Desafíos interactivos (tipo CAPTCHA):** Es común encontrar en aplicaciones web desafíos [17] para verificar si el usuario que intenta acceder al recurso es humano o no. Estos desafíos suelen ser imágenes, preguntas o tareas que requieren la intervención del usuario para conseguir pasarlos.

# 4. Técnicas y herramientas

Las técnicas y herramientas utilizadas en el desarrollo de un proyecto son fundamentales para cumplir su objetivo de manera eficiente. En este apartado se describen las principales tecnologías, lenguajes de programación, bibliotecas y herramientas de desarrollo empleadas en el proyecto, así como su justificación y el papel que desempeñan en el mismo.

Los principales lenguajes de programación utilizados en el proyecto son Python y Java.

Python se ha utilizado principalmente para el desarrollo del agente de aprendizaje por refuerzo, debido a su amplia gama de bibliotecas y frameworks especializados en este campo, como Stable-Baselines3 [13] y Gymnasium. También se ha usado para el desarrollo de la API con Flask-RESTful, para facilitar la posible interacción de la web con el agente, y que permite una implementación rápida y eficiente en recursos.

Java se ha utilizado para el desarrollo de la aplicación web, aprovechando su robustez y escalabilidad, junto con el framework Spring Boot 3 para facilitar la creación de aplicaciones web. Para la gestión de las dependencias se ha utilizado Maven, que permite una gestión eficiente de las dependencias y la construcción del proyecto. Para las pruebas unitarias se ha utilizado JUnit, y para los informes de pruebas Jacoco. Spring Boot tiene una gran comunidad y soporte, lo que facilita la resolución de problemas y la implementación de nuevas funcionalidades, junto con la posibilidad de integrar fácilmente otras tecnologías y herramientas, como es el caso de persistencia de datos con JPA, Hibernate y MySQL, la integración de un sistema de autenticación y autorización robusto con Spring Security, o la integración de un modelo de lenguaje a través de una API.

Los principales entornos de desarrollo utilizados han sido:

- **IntelliJ IDEA:** Para el desarrollo de la aplicación web en Java, que ofrece una amplia gama de herramientas y características para facilitar el desarrollo, refactorización, depuración, entre otras. S ha utilizado con la extensión de SonarQube.
- **Visual Studio Code:** Para el desarrollo del agente de aprendizaje por refuerzo en Python y la API REST, que es un editor ligero y altamente configurable. Se ha utilizado con extensiones para Python, Jupyter Notebook, SonarQube y Docker.

Para la visualización de datos en la aplicación web se ha utilizado Plotly.js, que permite crear gráficos interactivos y visualizaciones avanzadas de datos, mejorando la experiencia del usuario. Para el desarrollo del frontend se ha utilizado HTML5, CSS y JavaScript, junto con Bootstrap 5 para facilitar el diseño responsivo y moderno de la interfaz de usuario.

Como control de versiones se ha usado Git, que permite llevar un seguimiento de los cambios realizados en el código fuente del proyecto, facilitando la gestión de versiones, en el que el cliente usado es GitHub, permitiendo el almacenamiento de las versiones en la nube, junto con la posibilidad de usar GitHub Actions, e integrarlo con GitHub Projects para la gestión de tareas y seguimiento del proyecto.

Para la gestión de la documentación se ha utilizado LaTeX, que permite crear documentos técnicos de alta calidad, y para la transformación a pdf se ha utilizado una GitHub Action que usa una imagen de Docker para transformar los archivos fuente a pdf, y para la gestión de referencias bibliográficas se ha utilizado Mendeley, que facilita la gestión y citación de referencias en el proyecto.

Para la creación de diagramas y gráficos se ha utilizado Draw.io, que permite crear todos los diagramas de manera sencilla y rápida, con integración tanto en la web como en Visual Studio Code.

Para la calidad de código y la detección de errores se ha utilizado SonarQube, que permite analizar el código fuente del proyecto y detectar posibles errores, vulnerabilidades y problemas de calidad. Se ha integrado con GitHub Actions para realizar análisis automáticos del código en cada commit y pull request. También se ha utilizado AutoPep8 para el formateo del código Python, que permite mantener un estilo de codificación consistente y mejorar la legibilidad del código.

Para la gestión de los registros DNS, compra del dominio y seguridad de la aplicación web se ha utilizado Cloudflare. Para la compra del dominio se

ha elegido Cloudflare debido a que ofrece un servicio de registro de dominios a precios competitivos y con una interfaz fácil de usar, además de que proporciona una gran variedad de servicios, reduciendo la necesidad de utilizar múltiples proveedores para diferentes servicios. Cloudflare proporciona en su capa gratuita amplios servicios de seguridad y rendimiento para aplicaciones web, incluyendo protección contra ataques DDoS, optimización del rendimiento y gestión de certificados SSL. Cloudflare [12, 1] es una plataforma muy popular y confiable, utilizada por muchas empresas, particulares y organizaciones en todo el mundo, con una alta efectividad.

En el despliegue de la aplicación se ha elegido un servidor AWS EC2, que permite un control total sobre la infraestructura y la configuración del servidor. Es una de las principales plataformas de computación en la nube más populares, confiables y seguras, con una amplia gama de servicios y características que permiten escalar y gestionar aplicaciones de manera eficiente. La capa gratuita permite la ejecución de aplicaciones pequeñas pero con funcionalidades limitadas. La capa gratuita, proporciona una instancia con 1GB de RAM, lo cual no es suficiente para ejecutar este proyecto, por lo que se ha optado por una instancia t3.medium, que proporciona 4GB de RAM a un precio asequible.

En la siguiente tabla se muestran marcadas con X las herramientas y tecnologías utilizadas en cada parte del proyecto.

| Herramientas | Modelo de aprendizaje por refuerzo | Web | API REST | BD | Memoria |
|---|---|---|---|---|---|
| Python | X | | X | | |
| Jupyter Notebook | X | | | | |
| Stable-Baselines3 | X | | | | |
| Gymnasium | X | | | | |
| Matplotlib | X | | | | |
| Java | | X | | | |
| Spring Boot 3 | | X | | | |

*continúa en la página siguiente*

*continúa desde la página anterior*

| Herramientas | Modelo de aprendizaje por refuerzo | Web | API REST | BD | Memoria |
|---|---|---|---|---|---|
| Maven | | X | | | |
| JUnit | | X | | | |
| Selenium | | X | | | |
| HTML5 | | X | | | |
| CSS3 | | X | | | |
| Plotly.js | | X | | | |
| JavaScript | | X | | | |
| Bootstrap 5 | | X | | | |
| Flask-RESTful | | | X | | |
| Pytest | | | X | | |
| REST | | X | X | | |
| JSON | X | X | X | | |
| Docker | X | X | X | X | |
| Docker Compose | X | X | X | X | |
| MySQL | | | | X | |
| SQL | X | | | X | |
| Git | X | X | X | X | X |
| GitHub Projects | X | X | X | X | X |
| GitHub Actions | X | X | X | | X |
| GitHub Dependabot | X | X | X | | X |
| GitHub Secret Scanning | X | X | X | | X |
| GitHub Code Scanning | X | X | X | | X |
| GitHub Copilot | X | X | X | | X |
| SonarQube | X | X | X | | X |
| Draw.io | | | | | X |
| LaTeX | | | | | X |
| IntelliJ IDEA | | X | | X | |
| VS Code | X | | X | | X |

*continúa en la página siguiente*

*continúa desde la página anterior*

| Herramientas | Modelo de aprendizaje por refuerzo | Web | API REST | BD | Memoria |
|---|---|---|---|---|---|
| Mendeley | | | | | X |
| AWS EC2 | | X | X | X | |
| Cloudflare | | X | | | |
| API Groq | | X | | | |

Tabla 4.1: Herramientas y tecnologías utilizadas en cada parte del proyecto

# 5. Aspectos relevantes del desarrollo del proyecto

El proyecto se ha dividido en tres partes, cada una de las cuales tiene un objetivo específico:

- **Generación del entorno de simulación y la generación de los datos de entrenamiento:** Cuyo objetivo es la generación artificial de paquetes para simular tráfico atacante y tráfico de benigno en un entorno de dispositivos de internet de las cosas. También tiene como objetivo la creación de un entorno de simulación para el entrenamiento del agente de aprendizaje por refuerzo.
- **Desarrollo del entorno y el agente de aprendizaje por refuerzo:** Incluyendo el ajuste de parámetros, mejora en la función de recompensa, entrenamiento del agente y su posterior evaluación una vez entrenado.
- **Desarrollo de la aplicación web:** Su principal objetivo es la visualización de los resultados obtenidos por el agente, y que los usuarios puedan visualizar de manera sencilla los resultados.

## 5.1. Generación del entorno de simulación y la generación de los datos de entrenamiento

Para el entorno de simulación se ha utilizado la librería de Python `Gymnasium`, que facilita la creación del entorno de simulación para el agente

de aprendizaje por refuerzo. La arquitectura del entorno se encuentra en la **Figura 5.1**. Esta arquitectura simula una generación de paquetes que son enviados a un dispositivo que pueda controlar el flujo de paquetes, que en este caso es un *router* simulado, que la gestión que hace de los paquetes entrantes es introducirlos en una simple cola. Junto al *router* se implementa el agente, que se encarga de decidir si descarta los paquetes, o permite el paso a su destino original.

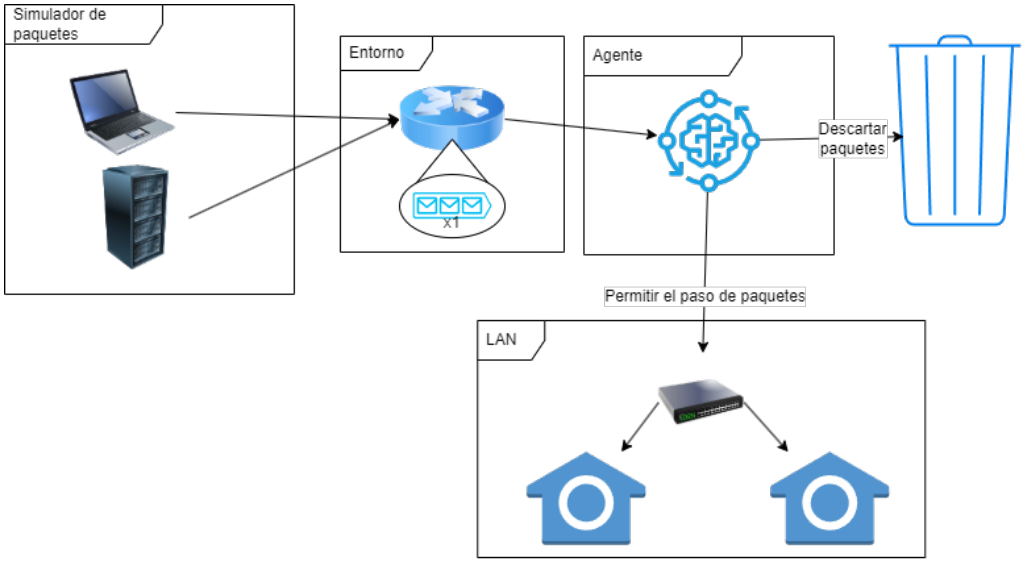


Figura 5.1: Arquitectura del entorno de simulación

La generación de los datos de entrenamiento se ha realizado creando generadores de paquetes, tanto de tráfico benigno como de tráfico atacante, que simulan el tráfico que se puede generar en un entorno de dispositivos de internet de las cosas. Los atributos de los paquetes se generan siguiendo una distribución por probabilidades, para que se asemeje lo más posible al tráfico real que se puede generar en un entorno de dispositivos de internet de las cosas, tanto cuando reciben un ataque de denegación de servicio, como cuando no.

En el entorno de entrenamiento, los estados de ataque y benigno cambian siguiendo una máquina de estados, en la que hay una cierta probabilidad de que el estado cambie. Entre cada cambio de estado, los paquetes transmitidos tienen las mismas características, es decir, cada ataque tiene un patrón de tráfico específico que cambia cuando cambia de estado. Esto permite que los

ataques sean variables para que el agente pueda aprender a detectar ataques de denegación de servicio con diferentes características.

## 5.2. Desarrollo del entorno y el agente

### Algoritmo de entrenamiento del agente

En el entrenamiento del agente se ha usado el algoritmo de aprendizaje por refuerzo Proximal Policy Optimization (PPO), que es un algoritmo de optimización de políticas que se basa en la idea de maximizar la función de recompensa esperada, y que ha demostrado ser efectivo [14, 19] en una amplia variedad de entornos y tareas.

El algoritmo PPO se encuentra implementado en la librería de Python `Stable Baselines3`, que es una librería de aprendizaje por refuerzo que proporciona implementaciones de algoritmos de aprendizaje por refuerzo basados en PyTorch, y que es ampliamente utilizada [13] en la comunidad de aprendizaje por refuerzo. Este tiene una compatibilidad total con la librería `Gymnasium`, que es la librería que se ha usado para crear el entorno de simulación, y con distintos tipos de espacios, que son cómo se han definido los estados, las acciones y las recompensas del entorno de simulación. En el caso de PPO, tiene compatibilidad con todos los espacios de `Gymnasium`, incluyendo espacios *Box*, *Discrete*, *MultiDiscrete* y *MultiBinary*, junto con la capacidad de multiprocesamiento, lo que permite entrenar el agente en múltiples entornos de simulación al mismo tiempo, mejorando así la eficiencia del entrenamiento y reduciendo el tiempo necesario para entrenar el agente. Busca un equilibrio entre la simplicidad y el rendimiento del mismo, proporcionando una implementación más sencilla que otros algoritmos de optimización de políticas, como el algoritmo Trust Region Policy Optimization (TRPO), también permite una gran flexibilidad en la configuración del entorno, y durante el entrenamiento del agente suele mantener una gran estabilidad, lo que lo hace adecuado para este tipo de entornos y tareas.

Las políticas disponibles del algoritmo PPO son:

- **MlpPolicy:** Alias de `ActorCriticPolicy`.
- **CnnPolicy:** Alias de `ActorCriticCnnPolicy`.
- **MultiInputPolicy:** Alias de `MultiInputActorCriticPolicy`.

La política que se ha usado en el entrenamiento del agente es la `MultiInputPolicy` ya que es la única de las 3 que permite usar múltiples entradas, lo que es necesario para el entorno de simulación.

En este caso, el espacio de acciones es discreto, ya que el agente solo puede tomar una acción a la vez, o permitir el tráfico o denegarlo en un instante de tiempo. En cuanto al espacio de observación, que son las variables que el agente puede observar del entorno, se ha definido como un espacio de tipo Dict de `Gymnasium`, que permite definir un espacio de observación con múltiples entradas. En este caso, se puede observar solo la ocupación de la cola de paquetes, en un intervalo entre 0 y 1, donde 0 significa que la cola está vacía y 1 significa que la cola está llena, y el número de paquetes descartados en el último instante de tiempo, que es un número entero mayor o igual que 0.

Para calcular los descartados, el agente cuenta todos los paquetes que no han podido entrar en la cola de paquetes, ya sea porque la cola está llena o porque el agente ha decidido denegar el tráfico en ese instante de tiempo, permitiendo gestionar el tráfico de manera más eficiente y evitar la saturación de la cola de paquetes.

### Función de recompensa

La función de recompensa utilizada es la siguiente:

$$reward = \begin{cases} -c \cdot \text{des}^2 + c_3 \cdot (\text{o\_an} - \text{o\_ac}) \cdot \text{o\_ac}, & \text{si des} > 0 \text{ y act} = \text{PERM} \\ -c_2 \cdot \text{des} + c_3 \cdot (\text{o\_an} - \text{o\_ac}) \cdot \text{o\_ac}, & \text{si des} > 0 \text{ y act} \neq \text{PERM} \\ (1 - \text{o\_ac}) \cdot c_4 + c_5, & \text{si des} <= 0 \text{ y act} = \text{PERM} \\ (1 - \text{o\_ac}) \cdot c_4, & \text{si des} <= 0 \text{ y act} \neq \text{PERM} \end{cases}$$

Las variables que se usan en la función de recompensa son:

- **des:** Número de paquetes descartados en el instante actual de tiempo. Es un número entero mayor o igual que 0.
- **o_an:** Ocupación de la cola de paquetes en el instante anterior, es decir, la ocupación de la cola de paquetes antes de tomar la acción actual. Es un número flotante entre 0 y 1.
- **o_ac:** Ocupación de la cola de paquetes en el instante actual. Es un número flotante entre 0 y 1.

- **act:** Acción tomada por el agente, que puede ser PERMITIR (PERM en la función) o DENEGAR el flujo de paquetes.

Donde los parámetros a configurar son:

- **c:** Parámetro de penalización por permitir el tráfico cuando hay paquetes descartados.
- **c2:** Parámetro de penalización por denegar tráfico cuando hay paquetes descartados.
- **c3:** Parámetro que multiplica a la mejora de la ocupación de la cola con respecto al instante anterior, es decir, la diferencia de la ocupación entre el instante actual y el anterior.
- **c4:** Parámetro de recompensa cuando no hay paquetes descartados.
- **c5:** Parámetro de recompensa que se suma a la recompensa cuando no hay paquetes descartados y se permite el tráfico.

Para la optimización de la función de recompensa se ha visualizado la función de recompensa como dos hiperplanos, uno con la acción permitir y otro con la acción denegar. Este gráfico interactivo se encuentra en la web en la sección de visualización de parámetros, representado en la **Figura 5.2**.

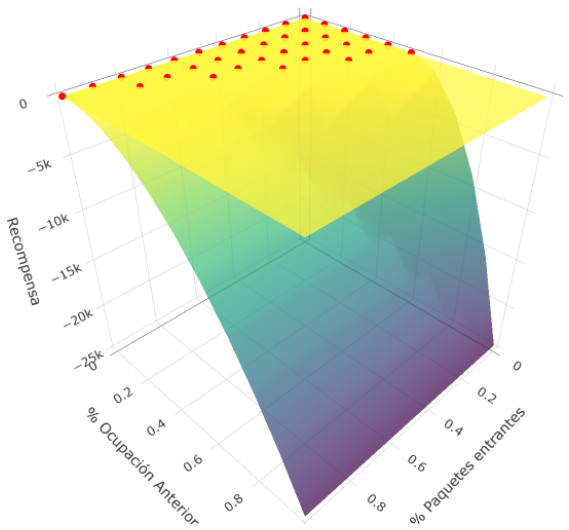


Figura 5.2: Función de recompensa del agente

El parámetro de precisión indica la cantidad de puntos que dibuja por eje, en el que valores mayores mostrarán resultados más precisos, pero se aumenta el tiempo de carga de los planos. Por otro lado, el parámetro de tolerancia indica las unidades de diferencia que tiene que haber en un punto entre los dos planos para que dibuje un punto rojo para indicar una "intersección" entre los planos.

Donde el plano amarillo representa la acción de denegar, en él tiene una pendiente poco pronunciada en comparación con el plano de permitir tráfico. Los puntos rojos indican las regiones del plano donde la recompensa es muy parecida entre los dos planos.

### Entrenamiento del agente

Para la evaluación del entrenamiento del agente se han usado las métricas de recompensa media, que indica cómo ha ido evolucionando la recompensa media a lo largo del tiempo, y *explained variance*, una métrica importante para comprobar si el agente está aprendiendo o no, que indica la calidad de las predicciones, en las que valores negativos indican que el agente obtendría mejores resultados si tomara siempre la misma acción, en que el mejor resultado posible es 1 [5], indicando que las predicciones del agente son perfectas. Las estadísticas obtenidas durante el entrenamiento del agente son las siguientes:

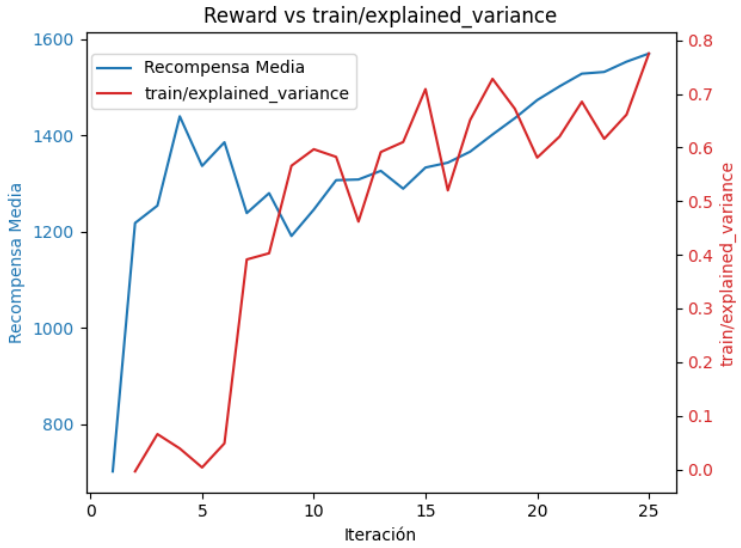


Figura 5.3: Estadísticas del entrenamiento del agente

Como se puede observar en la imagen anterior, la recompensa media ha ido aumentando a lo largo del tiempo, mejorando con el tiempo. En cambio, la *explained variance* ha sido más variable, aunque con un valor relativamente bueno.

### Evaluación del agente

Para la evaluación del agente ya entrenado, se genera un entorno de simulación, en el que se guardan los estados de los generadores de paquetes, para posteriormente, poderlo comparar con las acciones tomadas por el agente y así poder evaluar su rendimiento. Durante las predicciones del agente en el entorno, también se han guardado otros datos relevantes, como la ocupación de la cola en cada instante o la acción tomada.

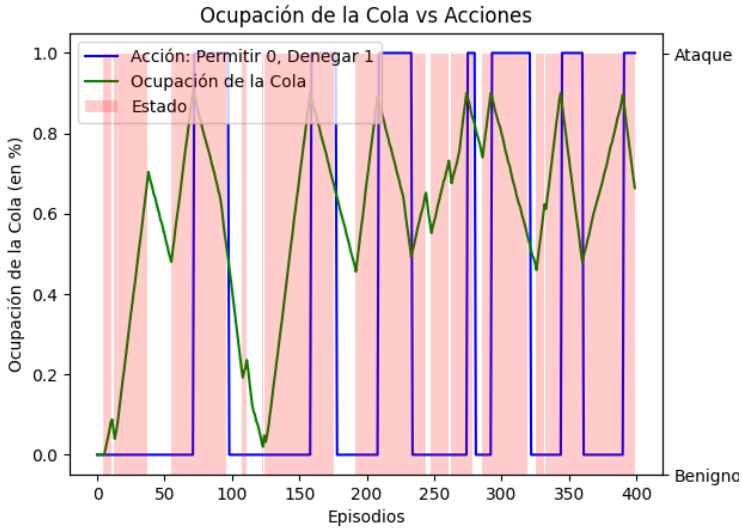


Figura 5.4: Estadísticas de la evaluación del agente

Como se puede ver en la imagen anterior, el fondo rojo indica que está en un estado de ataque, mandando paquetes de denegación de servicio hacia el agente, y el fondo blanco, lo contrario, es decir, generando tráfico benigno. En verde se puede ver el porcentaje de ocupación de la cola de paquetes, y en azul las acciones tomadas, donde 0 indica que el agente ha permitido el tráfico y 1 que lo ha denegado. Se puede apreciar que el agente ha aprendido a denegar el tráfico para que no sature la cola de paquetes, cumpliendo con el objetivo del proyecto, que es evitar la saturación de la cola de paquetes y

denegar el tráfico de denegación de servicio, permitiendo así que el tráfico benigno pueda ser procesado.

## 5.3. Desarrollo de la aplicación web

### Despliegue de la aplicación web

Para el despliegue de la aplicación web se ha usado un servidor virtual en **Amazon Web Services (AWS) EC2**, con una imagen de AMI de Amazon Linux 2023, y con un tamaño de instancia t3.medium, que dispone de los recursos necesarios para el despliegue y la ejecución de la aplicación. Para facilitar el despliegue y la gestión de la aplicación, se ha usado **Docker** y **Docker Compose**, que permiten crear contenedores separados para cada parte de la aplicación, con cada uno sus dependencias, facilitando así su despliegue y gestión. La arquitectura de la aplicación web se ha diseñado para que sea escalable y modular, permitiendo añadir nuevas funcionalidades en el futuro sin afectar al resto de la aplicación.

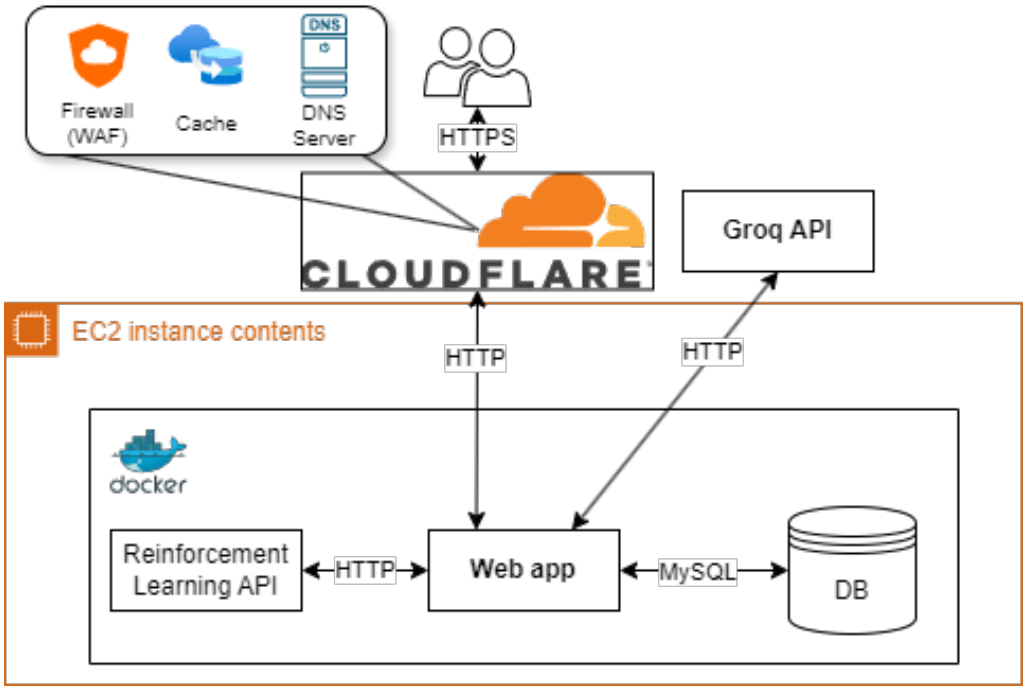


Figura 5.5: Arquitectura de la aplicación web

Los contenedores que se han creado son:

- **Web:** Contiene la aplicación web desarrollada en Java con Spring Boot 3, que se encarga de servir las páginas web al usuario, y realizar toda la gestión de los modelos, junto con la visualización de información sobre el proyecto. El contenedor final generado utiliza una construcción en multi-etapas, que en la primera etapa compila el proyecto en un archivo JAR, y en la segunda etapa se ejecuta el archivo JAR generado, permitiendo así reducir el tamaño del contenedor final, pudiéndose ahorrar dependencias solo necesarias para la etapa de compilación.
- **API:** Contiene la API REST desarrollada en Python con Flask, que se encarga de servir los datos necesarios desde Python para la aplicación web.
- **Base de datos:** Contiene la base de datos MySQL, que almacena los datos necesarios para la aplicación web y la API REST. Que usa un volumen de Docker para persistir los datos, de manera que si el contenedor se elimina, los datos no se pierden.

Para la comunicación entre los contenedores se han usado dos redes de Docker, una para la comunicación entre la aplicación web y la API REST, y otra para la comunicación entre la aplicación web y la base de datos. Esto permite que cada contenedor se pueda comunicar solo y exclusivamente con los contenedores que necesita, mejorando la seguridad y el rendimiento de la aplicación.

Para el despliegue de los contenedores de Docker, se ha usado Docker Compose, que permite definir y ejecutar aplicaciones multi-contenedor a partir de la configuración de un archivo `docker-compose.yml`. Este archivo define los servicios, redes y volúmenes necesarios para la aplicación, facilitando así su despliegue y gestión. En la configuración de Docker Compose se necesita definir las variables de entorno en el archivo `.env`.

## Seguridad del proyecto

Uno de los pilares más importantes en el desarrollo de software es la seguridad, y en este proyecto se ha tenido en cuenta desde el principio, implementando medidas de seguridad en todos los niveles.

### Seguridad en el código

Para garantizar la seguridad del código, se han seguido las siguientes prácticas:

- **Control de versiones:** Se ha utilizado Git como sistema de control de versiones, lo que permite llevar un registro de todos los cambios realizados en el código, facilitando la identificación de posibles errores y vulnerabilidades.

- **Revisión de dependencias:** Se ha realizado una revisión automática de las dependencias utilizadas en el proyecto, para asegurarse de que no contienen vulnerabilidades conocidas. Utilizando Dependabot, que es una herramienta de GitHub que permite detectar vulnerabilidades en las dependencias del proyecto y sugerir actualizaciones para solucionarlas.

- **Revisión de código:** Se ha realizado una revisión de código mediante pull requests, donde se revisa el código automáticamente con GitHub Copilot, SonarQube y CodeQL analysis, para detectar posibles errores y vulnerabilidades en el código. También se han utilizado herramientas de análisis estático de código de GitHub como Code Scanning y Secret Scanning, que permiten detectar posibles errores y vulnerabilidades en el código, así como secretos expuestos en el código, como contraseñas, claves API, etc, pudiendo llegar a bloquear el *commit* si se detecta algún secreto expuesto.

- **Gestión de Secretos:** En la gestión de secretos se ha utilizado un archivo `.env` que contiene las variables de entorno necesarias para la aplicación, como las credenciales de la base de datos, las claves API, etc. Este archivo no se debería incluir en el repositorio de Git para evitar que los secretos queden expuestos, debiendo ser añadido al archivo `.gitignore` para evitar que se suba al repositorio. Sin embargo, para que se pueda utilizar y probar el proyecto por parte del tribunal se ha incluido el archivo `.env`, que contienen las variables de entorno mínimas del funcionamiento del proyecto, pero no contiene los "secretos reales", como credenciales SSH para subir contenido al servidor y a GitHub y claves API de Cloudflare y de Groq. Si se desea usar la funcionalidad total de la parte web se debe de acceder a la **url** de la aplicación web, que es `https://www.cesarrv.com`.


Figura 5.6: Secretos usados en las acciones de GitHub

Los secretos usados en las acciones de GitHub son los mostrados en la **Figura 5.6**, siendo los necesarios en el funcionamiento del proyecto, como las credenciales de AWS, la clave API de Groq, y el token de Sonar y las claves SSH que permiten subir contenido al servidor y a GitHub.

### Seguridad en la infraestructura

En cuanto a la seguridad de la aplicación web, se han implementado las siguientes medidas:

- **Autenticación y autorización:** Se ha implementado un sistema de autenticación con Spring Boot Security, en el que permite gestionar los usuarios y sus roles, así como proteger las rutas de la aplicación web.
- **Protección contra ataques CSRF:** Se ha implementado protección contra ataques CSRF implementada por Spring Boot Security.
- **Cifrado de datos:** Se ha guardado el *hash* de las contraseñas de los usuarios en la base de datos, utilizando el algoritmo **BCrypt**, que es un algoritmo de cifrado de contraseñas seguro y ampliamente utilizado.

Esto permite proteger las contraseñas de los usuarios en caso de que la base de datos sea comprometida.

### Uso de Cloudflare

En la aplicación web se ha usado Cloudflare como servicio de protección, estadísticas, rendimiento y personalización de reglas, con el objetivo de mitigar ataques, mejorar el rendimiento, mejorar la seguridad general de la aplicación y optimización de costes. También se ha aprovechado la compra del dominio de `www.cesarrv.com` a través de Cloudflare, pudiendo así gestionar el dominio y los DNS desde la misma plataforma, facilitando la configuración de estos. En cuanto a las solicitudes que recibe el dominio, aún redirigiendo al servidor apagado, recibe alrededor de 1000 solicitudes al día, de las cuales ninguna es legítima, ya que el dominio no está activo y el servidor donde se aloja la web no está activo. Esto indica que el dominio ha sido indexado por los motores de búsqueda y está siendo escaneado por bots maliciosos.

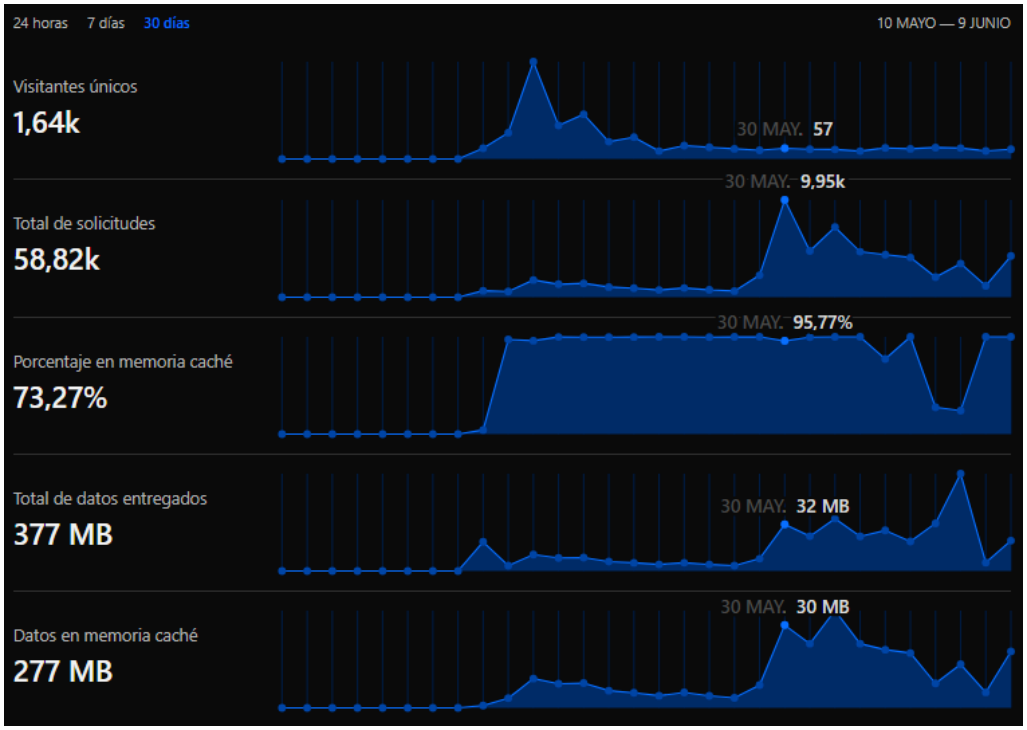


Figura 5.7: Estadísticas de Cloudflare del dominio cesarrv.com

Como se puede observar en la imagen anterior, el dominio ha recibido un gran número de solicitudes. Otros datos a reseñar es que la mayoría de

las solicitudes entregadas han estado guardadas en caché, lo que indica que Cloudflare ha podido servir las solicitudes sin necesidad de enviar la petición al servidor, mejorando así el rendimiento y reduciendo la carga y costos asociados del servidor.

Los principales escaneos detectados son:

- **Escaneos de git:** De las peticiones recibidas uno de los patrones más comunes es el de escanear el repositorio de git, buscando archivos sensibles como `.git/config`, `.git/HEAD`, `.git/index`, etc. Estos archivos pueden contener información sensible sobre la configuración del repositorio, como las ramas, los usuarios, las contraseñas, etc.
- **Escaneos de WordPress:** Otro patrón común detectado es el de escanear la web en busca de vulnerabilidades relacionadas con WordPress, como la búsqueda de archivos como `wp-config.php`, `wp-login.php`, `/wp-includes/`, etc. Estos archivos y directorios pueden contener información sensible sobre la configuración de WordPress, como las contraseñas, los usuarios, etc.

### Analíticas de tráfico web con Cloudflare

Cloudflare no solo proporciona funcionalidades de seguridad y rendimiento, sino que también permite obtener estadísticas detalladas sobre el tráfico web recibido en el dominio registrado. Estas estadísticas son muy útiles para entender el tráfico que recibe el dominio, identificar patrones de tráfico y detectar posibles ataques. Cloudflare proporciona estadíticas detalladas como, el número de solicitudes recibidas, el número de solicitudes bloqueadas, el número de solicitudes que han pasado las reglas de seguridad, etc. También proporciona estadísticas sobre el tráfico por país, por tipo de navegador, por tipo de dispositivo, etc.

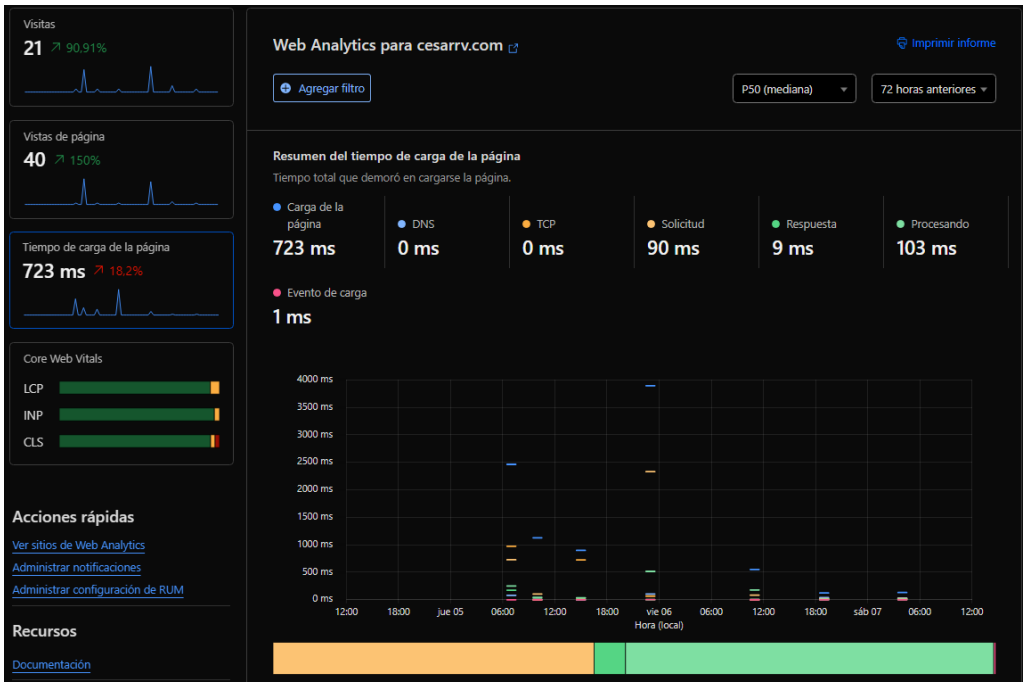


Figura 5.8: Estadísticas web de carga de Cloudflare del dominio cesarrv.com

La **Figura 5.8** muestra la mediana de los tiempos entre que un usuario hace una solicitud y el servidor responde. Como se puede observar, cuenta con tiempos reducidos en el que el mayor tiempo es la carga de la página, normal ya que tiene que cargar todo el contenido. Un aspecto relevante de la imagen es el apartado de *Core Web Vitals*, que son métricas de rendimiento web que miden la experiencia del usuario al interactuar con una página web y en este caso la mayor parte de las métricas están en verde, lo que indica que la página web cumple con los estándares de rendimiento recomendados.

### Configuración y reglas aplicadas en Cloudflare

Para la mitigación de los ataques descritos anteriormente, se han aplicado las siguientes reglas en Cloudflare:

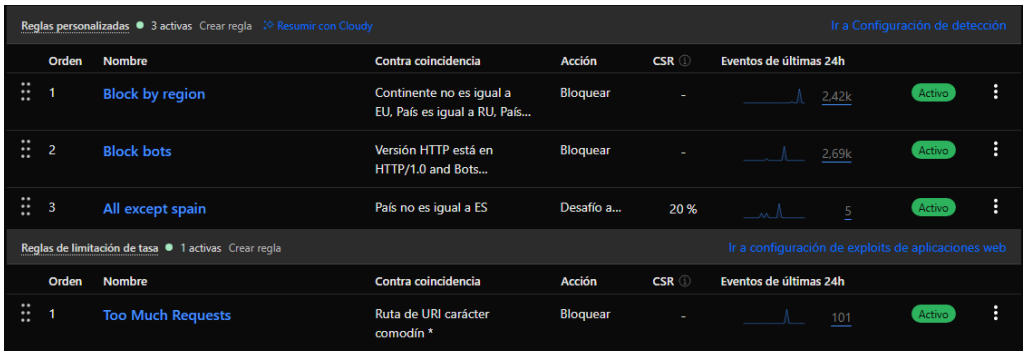


Figura 5.9: Reglas de seguridad aplicadas en Cloudflare

En el que la función principal de cada una es:

- **Block by region:** Bloquea las solcicitudes que provienen de regiones geográficas específicas, en este caso, se han bloqueado las solicitudes que provienen de fuera de Europa, y países espacíficos dentro de Europa que se han detectado que son origen de ataques, como Rusia, Ucrania, Irlanda, etc.
- **Block bots:** Bloquea las solicitudes que siguen patrones comunes por bots maliciosos, centrándose en los encabezados de las peticiones y el tipo de petición que hacen.
- **All except Spain:** De todas las solicitudes que han pasado las anteriores reglas, si el país de la solicitud no es España, administra un desafío gestionado por Cloudflare [17], que requiere que el usuario resuelva un CAPTCHA para poder acceder al sitio web. Esto ayuda a filtrar las solicitudes legítimas de las maliciosas.
- **Too Much Requests:** Limita las solicitudes que puede hacer un mismo usuario en un periodo de tiempo determinado, en este caso, se ha configurado para que un usuario no pueda hacer más de 40 solicitudes cada 10 segundos, lo que ayuda a prevenir ataques de denegación de servicio (DoS) y ataques de escaneo por fuerza bruta.

La parte de seguridad frente a ataques, Cloudflare se ha configurado con el objetivo de evitar y/o mitigar distintos tipos comunes de ataques:

- **Ataques de denegación de servicio (DoS y DDoS):** Cloudflare ayuda a mitigar ataques de denegación de servicio (DoS y DDoS)

mediante la filtración de solicitudes con fines maliciosos, la limitación de solicitudes y la protección contra bots maliciosos. Además si se detecta un ataque se puede activar el modo *I'm Under Attack* [18], que activa una protección adicional contra ataques DDoS, requiriendo que los usuarios resuelvan un desafío antes de acceder al sitio web.

- **Ataques XSS:** Cloudflare agrega ciertos encabezados de seguridad a las respuestas HTTP que proporcionan protección contra este tipo de ataques.

- **Ataques por URL:** Cloudflare hace un normalizado de las URL entrantes para que transforme los caracteres
en , juntar varias barras en una sola y aplicar la normalización de URL de la RFC 3986 [16]. Todos estos pasos ayudan a prevenir ataques basados en la manipulación de URL, como los ataques de inyección de código o los ataques de re-dirección.

# 6. Trabajos relacionados

Los trabajos más relacionados con el proyecto encontrados son:

- **TFG GII 22.56 Aprendizaje por refuerzo (Reinforcement Learning) en redes ópticas pasivas del grado en Ingeniería Informática de la Universidad de Burgos:** Un trabajo final de grado hecho por David Perez Moreno, que explora el uso de técnicas de aprendizaje por refuerzo para optimizar el rendimiento de las redes ópticas pasivas. El trabajo se centra en la implementación de un agente de aprendizaje por refuerzo que aprende a gestionar los recursos de la red de manera eficiente.
- **Deep Reinforcement Learning based Smart Mitigation of DDoS Flooding in Software-Defined Networks [10]:** Un trabajo que consiste en la mitigación de ataques DDoS en redes definidas por software (SDN) utilizando técnicas de aprendizaje por refuerzo.

# 7. Conclusiones y Líneas de trabajo futuras

## 7.1. Conclusiones

El proyecto ha conseguido desarrollar un agente, que mediante el uso de técnicas de aprendizaje por refuerzo es capaz de mitigar ataques de denegación de servicio en un entorno de simulación basado en dispositivos del Internet de las Cosas. Debido a que los dispositivos IoT presentan características particulares, como pueden ser recursos limitados y heterogeneidad en sus comportamientos, ello conlleva a que la mitigación de los ataques en este tipo de entornos tenga mucho impacto y sea un desafío considerable.

El agente desarrollado con el algoritmo PPO ha demostrado la capacidad de aprendizaje de los patrones de los ataques para evitar la saturación del sistema, y por consecuencia la denegación de servicio a los usuarios. Gracias al uso de la librería **Gymnasium**, en el presente trabajo se ha realizado la creación de un entorno dinámico y personalizado, en el que se simula un *router*, que decide en cada instante si el tráfico se reenvía o se deniega. En este entorno, el agente ha sido capaz, a partir de solamente conocer el porcentaje de ocupación de la cola y los paquetes descartados en el último instante, decidir cuando permite o deniega el tráfico para mitigar los posibles ataques, minimizando los paquetes que se descartan. La función de recompensa ha sido clave tanto en la penalización por pérdida de paquetes, como la denegación de tráfico legítimo. El proceso de ajuste y visualización interactiva ha sido un apoyo que ha permitido entrenar un agente equilibrado.

Por otro lado, en la aplicación web desarrollada es importante en términos de visualización de datos y de documentación del proyecto, permitiendo a los usuarios tener un mayor conocimiento del trabajo realizado a lo largo de este trabajo de fin de grado.

En resumen, el proyecto no solo se ha logrado una solución eficaz para la mitigación de ataques de denegación de servicio mediante aprendizaje por refuerzo, sino que también se ha realizado una herramienta de apoyo, como es la interfaz web, donde se proporciona un apoyo visual y un sistema en el que visualizar las estadísticas de los modelos más fácilmente, permitiendo una mejor comprensión del trabajo realizado.

## 7.2. Líneas de trabajo futuras

Como posibles líneas futuras del proyecto se han planteado varias que, principalmente buscan mejorar el rendimiento del agente y la funcionalidad de la aplicación web.

### Mejoras en el agente de aprendizaje por refuerzo

Las principales líneas de trabajo son:

- **Mejora en el entorno de simulación:** Llevar la simulación a un entorno de simulación de tráfico, como GNS3, en el que se implemente el agente y se pueda ver y evaluar cual es su desempeño en un entorno más complejo.
- **Mejora en la recompensa:** Optimizar la función de recompensa añadiendo más variables, así el agente puede tener una mayor información relevante y mejorar su aprendizaje.
- **Mitigación de más ataques:** Añadir al agente la posibilidad de mitigación de más tipos de ataques como:
  - Escaneo de puertos (PortScan)
  - Escaneo de sistema operativo (OSScan)

### Mejoras en la aplicación web

Las principales líneas de trabajo son:

- **Mejora de la interfaz de usuario:** Realizar cambios en el diseño y la usabilidad de la aplicación para facilitar la interacción del usuario. En cuanto a la parte técnica, se plantea la sustitución de los estilos de Bootstrap 5 por un framework de frontend moderno como *React*, *Vue.js* o *Angular*, lo que permitiría una interfaz más dinámica, modular y fácil de mantener.
- **Ampliación de funcionalidades:** Incluir nuevas características en la aplicación web, como la posibilidad de ajustar los parámetros de entrenamiento del agente desde la aplicación web, permitiendo a los usuarios personalizar el comportamiento del agente según sus necesidades.

# Bibliografía


[1] Sandi Adhar and Usep Saprudin. Implementasi cloudflare zero trust dalam mendeteksi aktivitas cryptojcking pada jaringan komputer. *JTKSI (Jurnal Teknologi Komputer dan Sistem Informasi)*, 6, 2023.

[2] Saman Shojae Chaeikar, Mohammadreza Jafari, and Hamed Taherdoost. Definitions and criteria of cia security. *International Journal of Advanced Computer Science and Information Technology*, 1, 2012.

[3] Nicholas Edwards, Sara Bliss Kiser, and Janel Bell Haynes. Answering the cybersecurity issues: Confidentiality, integrity, and availability. *Journal of Strategic Innovation and Sustainability*, 15, 2020.

[4] Aulab Hackademy. Api y api restful qué son y las principales diferencias | aulab hackademy. https://aulab.es/noticia/86/api-y-api-restful-que-son-y-las-principales-diferencias.

[5] Jonathan Hui. Rl — tips on reinforcement learning | by jonathan hui | medium. https://jonathan-hui.medium.com/rl-tips-on-reinforcement-learning-fbd121111775, 2 2023.

[6] Sushil Jajodia, Pierangela Samarati, and Moti Yung. *Encyclopedia of Cryptography, Security and Privacy*. Springer Berlin Heidelberg, 3 edition, 2025.

[7] Mohammed Aziz Al Kabir, Wael Elmedany, and Mhd Saeed Sharif. Securing iot devices against emerging security threats: Challenges and mitigation techniques. *Journal of Cyber Security Technology*, 7, 2023.

[8] Constantinos Kolias, Georgios Kambourakis, Angelos Stavrou, and Jeffrey Voas. Ddos in the iot: Mirai and other botnets. *Computer*, 50, 2017.

[9] Yuchong Li and Qinghui Liu. A comprehensive review study of cyber-attacks and cyber security; emerging trends and recent developments. *Energy Reports*, 7, 2021.

[10] Yandong Liu, Mianxiong Dong, Kaoru Ota, Jianhua Li, and Jun Wu. Deep reinforcement learning based smart mitigation of ddos flooding in software-defined networks. In *IEEE International Workshop on Computer Aided Modeling and Design of Communication Links and Networks, CAMAD*, volume 2018-September, 2018.

[11] Yutaka Matsuo, Yann LeCun, Maneesh Sahani, Doina Precup, David Silver, Masashi Sugiyama, Eiji Uchibe, and Jun Morimoto. Deep learning, reinforcement learning, and world models. *Neural Networks*, 152, 2022.

[12] Muhammad Nadeem, Ali Arshad, Saman Riaz, Syeda Wajiha Zahra, Muhammad Rashid, Shahab S. Band, and Amir Mosavi. Preventing cloud network from spamming attacks using cloudflare and knn. *Computers, Materials and Continua*, 74, 2023.

[13] Antonin Raffin, Ashley Hill, Adam Gleave, Anssi Kanervisto, Maximilian Ernestus, and Noah Dormann. Stable-baselines3: Reliable reinforcement learning implementations. *Journal of Machine Learning Research*, 22, 2021.

[14] Sanjna Siboo, Anushka Bhattacharyya, Rashmi Naveen Raj, and S. H. Ashwin. An empirical study of ddpg and ppo-based reinforcement learning algorithms for autonomous driving. *IEEE Access*, 11, 2023.

[15] Cloudflare Team. ¿qué es un protocolo? | definición de protocolo de red | cloudflare. https://www.cloudflare.com/es-es/learning/network-layer/what-is-a-protocol/.

[16] Cloudflare Team. How url normalization works · cloudflare rules docs. https://developers.cloudflare.com/rules/normalization/how-it-works/, 11 2024.

[17] Cloudflare Team. How challenges work · cloudflare challenges docs. https://developers.cloudflare.com/cloudflare-challenges/concepts/how-challenges-work/, 5 2025.

[18] Cloudflare Team. Under attack mode · cloudflare fundamentals docs. https://developers.cloudflare.com/fundamentals/reference/under-attack-mode/, 5 2025.

[19] Chao Yu, Akash Velu, Eugene Vinitsky, Jiaxuan Gao, Yu Wang, Alexandre Bayen, and Yi Wu. The surprising effectiveness of ppo in cooperative multi-agent games. In *Advances in Neural Information Processing Systems*, volume 35, 2022.

[20] Noe M. Yungaicela-Naula, Cesar Vargas-Rosales, Jesús Arturo Pérez-Díaz, and Diego Fernando Carrera. A flexible sdn-based framework for slow-rate ddos attack mitigation by using deep reinforcement learning. *Journal of Network and Computer Applications*, 205, 2022.