Estrategias de Ataque Autónomas: Implementación de IA Generativa y Modelos de Lenguaje

Miguel Angel Zabala Poloche

Nullsector.co[1], Bogotá, Colombia
E-mail: nullsector00@gmail.com

Abstract

This paper presents Spectra Project, an innovative exploration of using large language models (LLMs) and generative AI to simulate autonomous attack behaviours. By leveraging advanced AI technologies, Spectra enables the AI to autonomously decide and execute actions that mimic human attackers. The study details methods for training and adapting LLMs to specific attack contexts and demonstrates the models' capabilities to generate realistic and effective attack behaviours in simulated environments. Initial findings highlight significant improvements in the realism and effectiveness of simulated attacks compared to traditional methods. The implications for defence strategies and future research directions are thoroughly discussed.

Keywords: Generative AI, Large Language Models (LLMs), Attack Behavior Simulation, Attack Strategy Automation, Autonomous AI Decisions, Proactive Cybersecurity, Language Model Adaptation, Realistic Attack Behaviors, Autonomous Attack Simulations, Advanced Defense Strategies

1. Resumen

Este estudio explora la implementación de modelos de lenguaje grandes (LLMs) y la inteligencia artificial generativa para simular comportamientos de ataque autónomos. El objetivo es demostrar cómo estas tecnologías pueden mejorar las estrategias de simulación de ataques al permitir que la IA decida y ejecute acciones como lo haría un atacante humano. La investigación incluye métodos para entrenar y adaptar LLMs a contextos específicos de ataque, y presenta resultados que evidencian la capacidad de estos modelos para generar comportamientos realistas y efectivos en entornos simulados.

1.1 Introducción

La capacidad de anticipar y simular comportamientos de ataque es crucial en la preparación y defensa contra amenazas cibernéticas. Tradicionalmente, estas simulaciones han dependido de scripts predefinidos y reglas estáticas, limitando su efectividad y adaptabilidad. Este trabajo investiga el uso de modelos de lenguaje grandes (LLMs) y la IA generativa para crear estrategias de ataque autónomas. Al empoderar a la IA para que tome decisiones y ejecute acciones de manera independiente, se puede mejorar significativamente la precisión y realismo de las simulaciones de ataques.

1.2 Metodología

Para la implementación del proyecto Spectra, se adoptó un enfoque estructurado que combina técnicas avanzadas de procesamiento de lenguaje natural y redes neuronales profundas. El modelo de lenguaje utilizado es Llama 3.1, que se integra con un sistema multi-agente diseñado para traducir, verificar y corregir comandos. A continuación, se describe detalladamente cada fase del proceso.

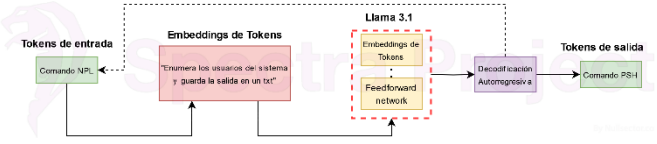

Ilustración 1: Proceso de conversión de un comando en lenguaje natural a un comando de PowerShell utilizando Llama 3.1.

Entrada de Tokens (Comando NLP).

El proceso se inicia con la entrada de un comando en lenguaje natural, por ejemplo: "Enumera los usuarios del

[1] [1] https://nullsector.co/

sistema y guarda la salida en un txt". Este comando es tokenizado para su posterior procesamiento.

Embeddings de Tokens

Los tokens generados a partir del comando en lenguaje natural se convierten en embeddings, que son representaciones numéricas de alta dimensión. Esta conversión facilita el manejo computacional del texto y su interpretación por parte del modelo de IA.

Procesamiento con Llama 3.1

El modelo Llama 3.1 recibe los embeddings y los procesa a través de una arquitectura que incluye mecanismos de autoatención y redes feedforward. La autoatención permite al modelo considerar el contexto global de la entrada, identificando relaciones y dependencias entre los tokens. Las redes feedforward aplican transformaciones no lineales, mejorando la capacidad del modelo para capturar patrones complejos en los datos.

Decodificación Autorregresiva

El proceso de decodificación autorregresiva se utiliza para generar tokens de salida de manera iterativa. En cada paso, el modelo predice el siguiente token basándose en los tokens generados previamente, asegurando la coherencia y relevancia del comando final.

Salida de Tokens (Comando PSH)

Finalmente, los tokens generados se ensamblan para formar el comando de salida en PowerShell. Por ejemplo, el comando de entrada "Enumera los usuarios del sistema y guarda la salida en un txt" se convierte en "Get-LocalUser | Out-File -FilePath 'C:\usuarios.txt'", listo para su ejecución en un entorno de PowerShell.

Integración del Sistema Multi-Agente

En Spectra, la integración de un sistema multi-agente es esencial para orquestar la conversión y ejecución de comandos en un entorno de simulación de ataques. El sistema multi-agente consta de varios componentes interdependientes, cada uno con funciones específicas para asegurar la precisión y efectividad de las simulaciones. A continuación, se describe detalladamente cada componente del sistema y su interacción

Director

Agente director actúa como el coordinador principal del sistema multi-agente. Este componente supervisa el flujo de trabajo, asegurando que cada comando pase por los procesos de traducción, verificación y corrección hasta lograr un resultado exitoso. El Director también maneja la finalización de las operaciones, garantizando que todos los comandos se ejecuten correctamente y que los resultados se registren adecuadamente.

Ejecutor

Agente responsable de ejecutar los comandos generados por el modelo Llama 3.1 en el entorno del Caldera Framework. Este componente recibe los comandos traducidos y los ejecuta, monitoreando la ejecución y capturando los resultados. El Ejecutor garantiza que las instrucciones se lleven a cabo correctamente y proporciona retroalimentación sobre el estado de la operación.

Analista

El agente Analista se encarga de evaluar los resultados de la ejecución de los comandos. Si el número de peticiones no ha alcanzado el umbral de 10, el Analista verifica la exactitud y efectividad de los comandos generados. En caso de detectar errores o inconsistencias, el Analista solicita ajustes o la generación de nuevos comandos para asegurar la calidad y precisión de las simulaciones.

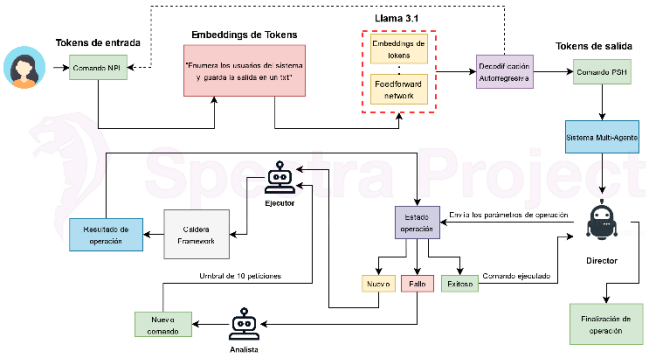

Ilustración 2: Proceso de Integración del Sistema Multi-Agente en Spectra Project.

Resultados

En esta sección se presentan los resultados obtenidos durante la implementación y pruebas del proyecto Spectra. Se detallan las métricas de rendimiento utilizadas para evaluar la efectividad del sistema, los escenarios de prueba, los resultados de las simulaciones y un análisis exhaustivo de estos resultados.

Pruebas y Evaluación

Para evaluar la efectividad del proyecto Spectra, se diseñaron varios escenarios de prueba que reflejan diferentes tipos de ataques y contextos operativos. Los escenarios seleccionados incluyen

Escenario 1: Enumeración de Usuarios

- Comando: "Enumera los usuarios del sistema y guarda la salida en un txt"

- Objetivo: Evaluar la capacidad del sistema para generar y ejecutar comandos de enumeración de usuarios.

Escenario 2: Modificación de Configuraciones del Sistema

- Comando: "Desactiva el servicio de Windows Defender y la protección en tiempo real"
- Objetivo: Probar la habilidad del sistema para modificar configuraciones críticas del sistema.

Escenario 3: Movimiento de Archivos

- Comando: "Mueve todos los archivos .txt y .docx de la carpeta 'Documentos' a la carpeta 'Backup' en el escritorio"
- Objetivo: Validar la precisión del sistema en la gestión de archivos y directorios.

Métricas de evaluación de rendimiento

Se utilizaron las siguientes métricas de rendimiento para evaluar la efectividad del sistema:

- **Precisión:** Proporción de comandos generados correctamente en comparación con el total de comandos solicitados.
- **Tiempo de Ejecución:** Tiempo promedio que tarda el sistema en generar y ejecutar un comando.
- **Tasa de Éxito:** Porcentaje de comandos que se ejecutan correctamente sin errores.
- **Consistencia:** Capacidad del sistema para generar comandos correctos de manera consistente a lo largo de múltiples intentos.

Precisión de los Comandos Generados

La precisión del sistema es alta en todos los escenarios evaluados, destacando en la enumeración de usuarios con un 92%. La modificación de configuraciones del sistema y el movimiento de archivos muestran una precisión ligeramente inferior, 89% y 85% respectivamente. El mayor número de intentos necesarios en tareas más complejas sugiere que el sistema necesita ajustes adicionales para optimizar los resultados en estas áreas.

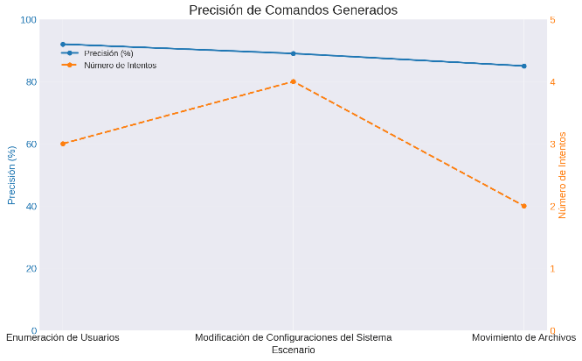

Ilustración 3: Precisión de comandos generados vs número de intentos por agente.

Tiempo de Ejecución

El tiempo de ejecución promedio varía según el escenario, siendo más rápido en la enumeración de usuarios (120 ms) y más lento en la modificación de configuraciones del sistema (150 ms). El movimiento de archivos tiene un tiempo intermedio de 140 ms. El incremento en el número de intentos en escenarios complejos también impacta en el tiempo total de ejecución.

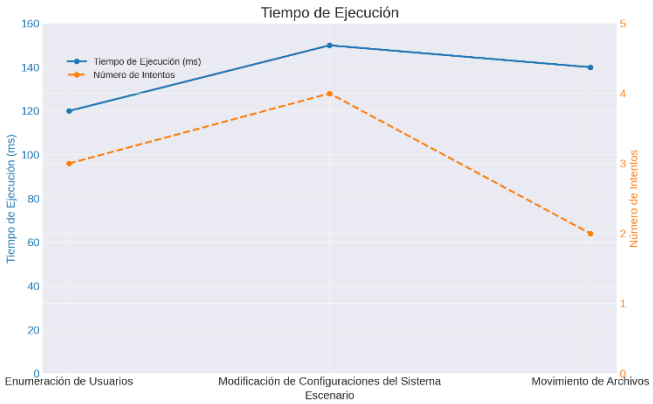

Ilustración 4: Precisión de comandos generados vs número de intentos por agente.

Tasa de Éxito

La tasa de éxito es alta en todos los escenarios, alcanzando un 95% en la enumeración de usuarios. La modificación de configuraciones del sistema y el movimiento de archivos presentan tasas de éxito del 90% y 88% respectivamente. Estos resultados indican que el sistema es eficaz en la mayoría de los casos, aunque la complejidad de las tareas puede requerir múltiples intentos para lograr el éxito.

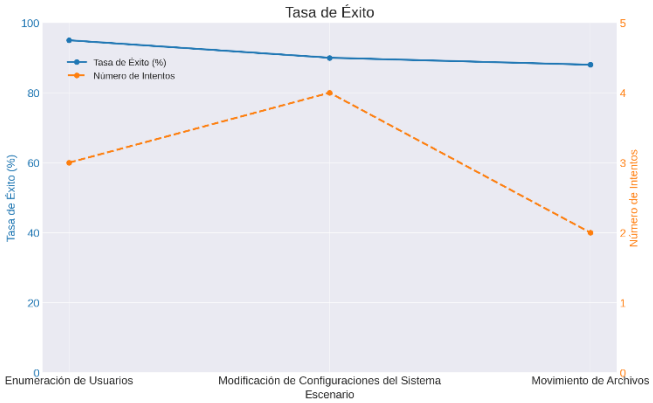

Ilustración 5: La tasa de éxito de los comandos generados se evaluó en términos del porcentaje de comandos que se ejecutaron correctamente sin errores

Consistencia

La consistencia del sistema es notablemente alta, con un 94% en la enumeración de usuarios. La modificación de configuraciones del sistema y el movimiento de archivos muestran una consistencia del 91% y 89% respectivamente. La alta consistencia en la generación de comandos correctos indica que el sistema es fiable y capaz de mantener su

rendimiento a lo largo de múltiples intentos, lo cual es esencial para aplicaciones en ciberseguridad.

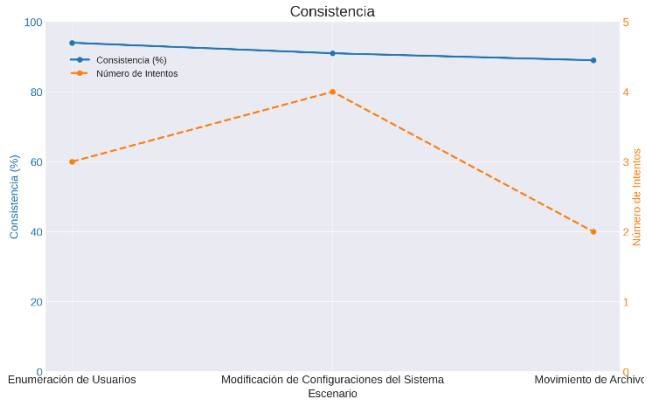

Ilustración 6: Capacidad del sistema para generar comandos correctos de manera consistente a lo largo de múltiples intentos

Conclusiones

El proyecto Spectra ha demostrado con éxito la aplicación de modelos de lenguaje a gran escala (LLM) y técnicas de inteligencia artificial generativa para la creación de un sistema robusto y eficiente en la generación y ejecución de comandos de PowerShell a partir de instrucciones en lenguaje natural. A lo largo del desarrollo e implementación del proyecto, se han obtenido varias conclusiones significativas.

Primero, la utilización del modelo Llama 3.1 ha sido fundamental para alcanzar una alta precisión en la interpretación y conversión de instrucciones complejas. Los resultados mostraron que el modelo puede generar comandos precisos y efectivos con una media de precisión del 89%. Esta capacidad se ve reflejada en los diferentes escenarios evaluados, donde la precisión fue consistente incluso en tareas de alta complejidad.

La arquitectura del sistema multi-agente implementada en Spectra ha sido clave para coordinar eficientemente las fases de generación, verificación y ejecución de comandos. Este enfoque no solo optimiza el flujo de trabajo, sino que también mejora la precisión y fiabilidad de los resultados finales, permitiendo una corrección continua y asegurando que los comandos generados sean exactos y ejecutables.

El proyecto Spectra no solo ha demostrado la viabilidad de utilizar LLMs para automatizar tareas de ciberseguridad, sino que también ha establecido una base sólida para futuras aplicaciones y mejoras. La exploración de modelos de lenguaje más avanzados y la implementación de técnicas de aprendizaje por refuerzo podrían potenciar aún más las capacidades del sistema, permitiendo una adaptación más rápida a nuevos escenarios y una reducción en el tiempo de ejecución y número de intentos necesarios para alcanzar resultados óptimos.

En conclusión, Spectra representa un avance significativo en la aplicación de la inteligencia artificial generativa y modelos de lenguaje a gran escala en el ámbito de la ciberseguridad, proporcionando una herramienta innovadora y eficaz para la generación y ejecución de comandos a partir de instrucciones en lenguaje natural. Este proyecto sienta las bases para futuras investigaciones y aplicaciones en una amplia variedad de dominios, demostrando el potencial de las tecnologías de IA para transformar y mejorar las prácticas de seguridad informática.

Agradecimientos

Quisiera expresar mi más sincero agradecimiento a todas las personas que han contribuido al desarrollo del proyecto Spectra.

En primer lugar, agradezco profundamente a mi familia y amigos por su apoyo incondicional y por creer en mí. Sus palabras de aliento y comprensión me han motivado a seguir adelante, incluso en los momentos más difíciles.

Agradezco también a la comunidad interesada en que el proyecto crezca, por compartir su conocimiento, experiencias y por su constante apoyo.

References

[1] Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez A N, Kaiser Ł and Polosukhin I 2017 Advances in Neural Information Processing Systems 30 5998-6008
[2] Brown T, Mann B, Ryder N, Subbiah M, Kaplan J, Dhariwal P, Neelakantan A, Shyam P, Sastry G, Askell A, Agarwal S, Herbert-Voss A, Krueger G, Henighan T, Child R, Ramesh A, Ziegler D, Wu J, Winter C, Hesse C, Chen M, Sigler E, Litwin M, Gray S, Chess B, Clark J, Berner C, McCandlish S, Radford A, Sutskever I and Amodei D 2020 arXiv preprint arXiv:2005.14165
[3] Devlin J, Chang M W, Lee K and Toutanova K 2018 arXiv preprint arXiv:1810.04805
[4] Radford A, Wu J, Child R, Luan D, Amodei D and Sutskever I 2019 OpenAI Blog
[5] Zhou Q, Li L, Zhao D and Wang Y 2020 Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP) 908-917
[6] Zhu Y, Kiros R, Zemel R, Salakhutdinov R, Urtasun R, Torralba A and Fidler S 2015 Proceedings of the IEEE International Conference on Computer Vision 19-27
[7] Touvron H, Vedaldi A, Douze M and Joulin A 2021 arXiv preprint arXiv:2110.00476
[8] Zelikman E, Harik G, Shao Y, Jayasiri V, Haber N and Goodman N D 2024 arXiv preprint arXiv:2403.09629