Universidad Corporativa | Programas Académicos

Ingeniería en Desarrollo de Software

Actividad 3 – Cross Site Scripting (XSS)

Auditoria Informática

Ingeniería en Desarrollo de Software

Tutor: Mtra. Jessica Hernández Romero

Alumno: Fernando Pedraza Garate

Fecha: 25 de octubre del 2024

Índice

Introducción

La gestión de sesiones y la pérdida de autenticación son términos comunes en el contexto de seguridad informática, particularmente cuando se trabaja con aplicaciones web o servicios que requieren autenticación de usuarios, donde la sesión representa un período en el que el sistema y el usuario están conectados de manera autenticada, y la gestión, se refiere al control y mantenimiento de la sesión del usuario, una vez que ha sido autenticado, se crea una sesión única para identificarlo y asociar sus acciones en el sistema, en donde los sistemas suelen almacenar una "cookie", o un token para identificar al usuario en cada solicitud sin tener que pedir credenciales repetidamente, estas sesiones suelen tener un tiempo de vida limitado, si un usuario está inactivo durante mucho tiempo o si se cumplen ciertas condiciones, la sesión puede expirar y necesitará autenticarse de nuevo, el sistema debe permitir al usuario cerrar la sesión de manera segura cuando ya no necesite estar autenticado.

La pérdida de autenticación ocurre cuando un usuario pierde el acceso a su sesión autenticada por algún motivo, lo que puede requerir que se autentique nuevamente, y si la sesión tiene un tiempo de vida predeterminado y el usuario ha estado inactivo demasiado tiempo, el sistema cerrará la sesión automáticamente, en algunos casos, un administrador del sistema puede revocar las sesiones activas, en aplicaciones como escritorios remotos o servicios basados en la nube, una interrupción en la conexión puede causar pérdida de autenticación, obligando al usuario a volver a iniciar sesión y si los tokens de autenticación se invalidan o son manipulados (por ejemplo, en ataques de seguridad), el usuario perderá su autenticación.

La serialización se refiera al proceso de convertir un objeto o datos en una estructura que puede ser almacenada o transmitida (por ejemplo, convertir un objeto en un archivo o cadena de texto), **la deserialización** es el proceso inverso, donde esa estructura de datos se convierte nuevamente en un objeto usable por la aplicación, es decir, convertir datos que están en un formato específico (como JSON, XML o binarios) de vuelta a un objeto que pueda ser manipulado por una aplicación, y **la deserialización insegura** hace referencia a una vulnerabilidad de seguridad que ocurre cuando una aplicación deserializa datos sin validarlos adecuadamente, permitiendo a un atacante inyectar o manipular datos maliciosos en el proceso.

Si el proceso de deserialización no se maneja correctamente, un atacante puede aprovecharlo para inyectar objetos maliciosos que pueden hacer que el sistema ejecute comandos no deseados o acceda a datos confidenciales. impactando en una ejecución remota de código (RCE), manipulando los datos para ejecutar código malicioso en el servidor, lo que puede resultar en una toma de control total del sistema afectado donde el atacante puede deserializar objetos que contienen información confidencial, como credenciales o información personal para realizar acciones no autorizadas, como modificar permisos o el comportamiento de una aplicación, ocasionando ataques de negación de servicio (DoS), datos maliciosos o desproporcionadamente grandes, intentando sobrecargar el servidor para provocar fallos en el sistema.

El Cross-Site Scripting (XSS) es un tipo de vulnerabilidad de seguridad en aplicaciones web que permite a un atacante inyectar scripts maliciosos en páginas web vistas por otros usuarios, los cuales se ejecutan en el navegador de la víctima y robar información confidencial, como cookies, sesiones de usuario o datos sensibles, sin que el usuario lo note.

El XSS Reflejado (Reflected XSS), ocurre cuando una aplicación web procesa datos de entrada del usuario (como un parámetro de URL) sin validarlos adecuadamente y los refleja en la página web, el XSS Almacenado (Stored XSS), es más peligroso, ya que afecta a todos los usuarios que acceden a la página comprometida, almacenado en el servidor (por ejemplo, en una base de datos) y se carga en la página web cada vez que un usuario la visita, y el XSS basado en DOM (DOM-based XSS) ocurre en el lado del cliente cuando el script malicioso manipula el Document Object Model (DOM) de la página web sin interactuar directamente con el servidor.

Descripción.

Una empresa de software está solicitando realizar varias pruebas de seguridad en páginas web que no cuentan con los candados de seguridad, y para la primera etapa, requiere se realice una prueba de la vulnerabilidad de la pérdida de autenticación y gestión de sesiones utilizando el programa WireShark con el objetivo de obtener las credenciales que se ingresaron y que estas se puedan mostrar

Posteriormente se deberá seleccionar un proyecto web que se haya realizado anteriormente y que cuente con las características de la función de inicio de sesión y de registro de usuarios, que cuente con conexión con una base de datos y subir el proyecto a un servidor web con la base de datos incluida, una vez que el proyecto esté en Internet, se realizará la instalación de WireShark, donde este programa permitirá realizar el ataque al sitio web, aprovechando la vulnerabilidad de falta de SSL o seguridad, y para la segunda etapa, se solicita realizar una prueba de deserialización insegura en una página específica mediante las cookies, para lograrlo se debe utilizar el programa Burp Suite Community Edition, con el objetivo de iniciar sesión como un usuario normal y luego pasar a modo administrador a través de las cookies, y con la ayuda de la plataforma PortSwigger, se realizará el ataque a la página proporcionada por ellos, en la que se iniciará sesión con las credenciales que se proporcionan, las cuales son para usuarios normales; no obstante, a través de las cookies, se entrara al modo administrador.

Este laboratorio utiliza un mecanismo de sesión basado en serialización y, por ende, es vulnerable a la escalada de privilegios, en el que hay que editar el objeto serializado en la cookie de sesión para aprovechar esta vulnerabilidad, obteniendo privilegios administrativos para finalmente, **eliminar la cuenta de Carlos como objetivo e iniciar sesión en la propia cuenta con las credenciales:**

Usuario: **wiener**

Contraseña: **Peter**

Para concretar se solicita realizar una prueba de vulnerabilidad de Cross Site Scripting (XSS) y se deben obtener las credenciales que se ingresen para iniciar sesión y posteriormente, desde Burp Suite, modificar la información para comprobar si se puede iniciar sesión o no.

Justificación.

La gestión de sesiones es crucial en el diseño de sistemas que involucran la autenticación de usuarios y el acceso a recursos privados o sensibles, algunas de las razones por las que la gestión adecuada de sesiones es fundamental, es asegurar que el sistema pueda identificar y controlar quién está interactuando con él de manera continua, protegiendo así los datos sensibles de los usuarios, y al mantener controlada la sesión se evitaran situaciones de acceso no autorizado que pueden ser aprovechados por terceros malintencionados que accedan al dispositivo del usuario o intercepten su sesión, ayudando a mitigar ataques comunes como el hijacking de sesión (secuestrar una sesión activa) o la falsificación de solicitudes entre sitios (CSRF), situaciones donde el atacante utiliza una sesión activa para realizar acciones en nombre del usuario sin su consentimiento.

Un sistema bien gestionado en cuanto a sesiones puede mejorar la experiencia del usuario al evitar interrupciones innecesarias, cuando la sesión expira, el sistema puede pedir que el usuario vuelva a iniciar sesión de manera segura y controlada, minimizando los inconvenientes, permitiendo que los usuarios puedan retomar actividades donde las dejaron, si se implementan funcionalidades como "sesión persistente".

Muchos sectores (como el financiero, médico o gubernamental) están sujetos a normativas de protección de datos y seguridad (como el Reglamento General de Protección de Datos (GDPR) o la Ley de Portabilidad y Responsabilidad de los Seguros de Salud (HIPAA) en EE.UU.), y las sesiones deben ser gestionadas de forma que se pueda garantizar que solo los usuarios autenticados tienen acceso a ciertos recursos o datos, siendo necesario saber quién accedió a qué, cuándo, y desde dónde, y esto solo se puede hacer correctamente si las sesiones son gestionadas y registradas, impactando en la eficiencia y el rendimiento del sistema, liberando recursos, lo que es importante en sistemas de gran escala, previniendo sobrecarga por mantener demasiadas sesiones simultáneas abiertas innecesariamente.

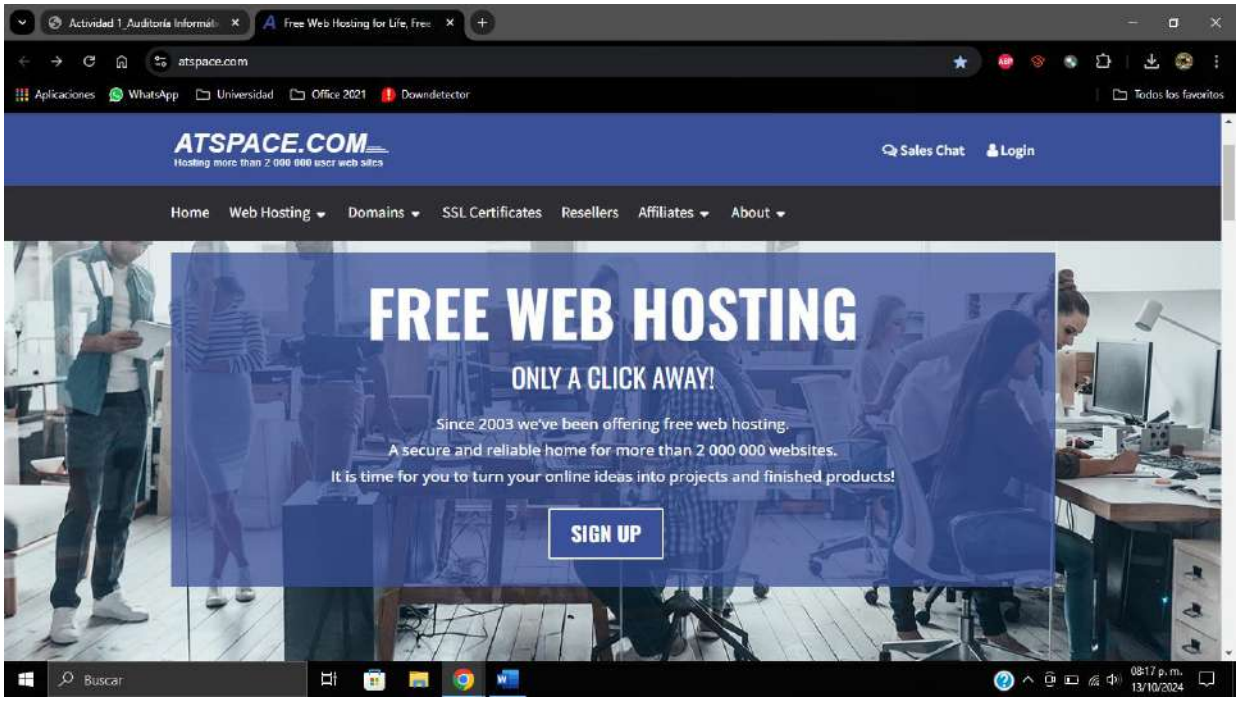
Las políticas de seguridad internas de las empresas o instituciones muchas veces requieren una gestión de sesiones robusta para cumplir con los estándares de la industria como una autenticación multifactor (MFA), y bloqueo automático si se detectan actividades inusuales o cambios en el entorno de seguridad del sistema protegiendo al usuario contra ataques o accesos no autorizados.

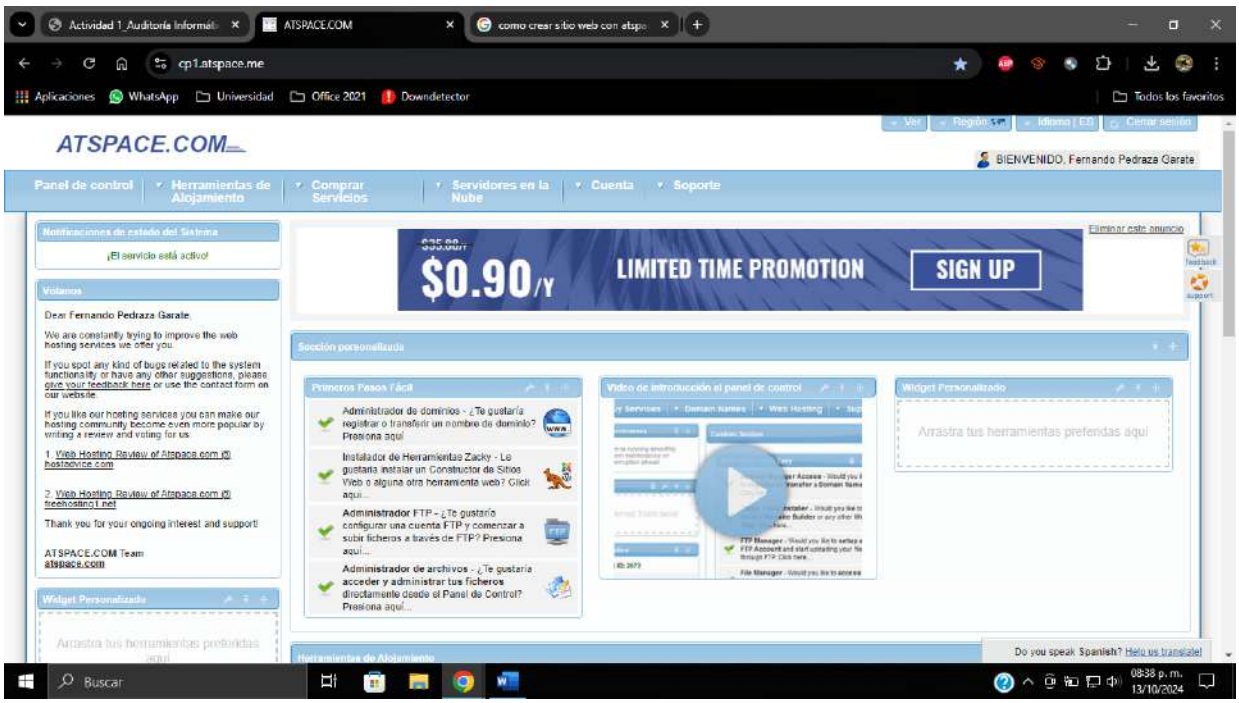
La deserialización insegura radica en que este tipo de vulnerabilidad permite a los atacantes explotar las funcionalidades internas de una aplicación para realizar acciones maliciosas a través de la inyección de datos manipulados en el proceso de deserialización, comprometiendo la integridad, confidencialidad y disponibilidad del sistema afectado, por esta razón, **es crucial implementar mecanismos robustos de validación y seguridad, así como evitar la deserialización de fuentes no confiables para mitigar estos riesgos.**

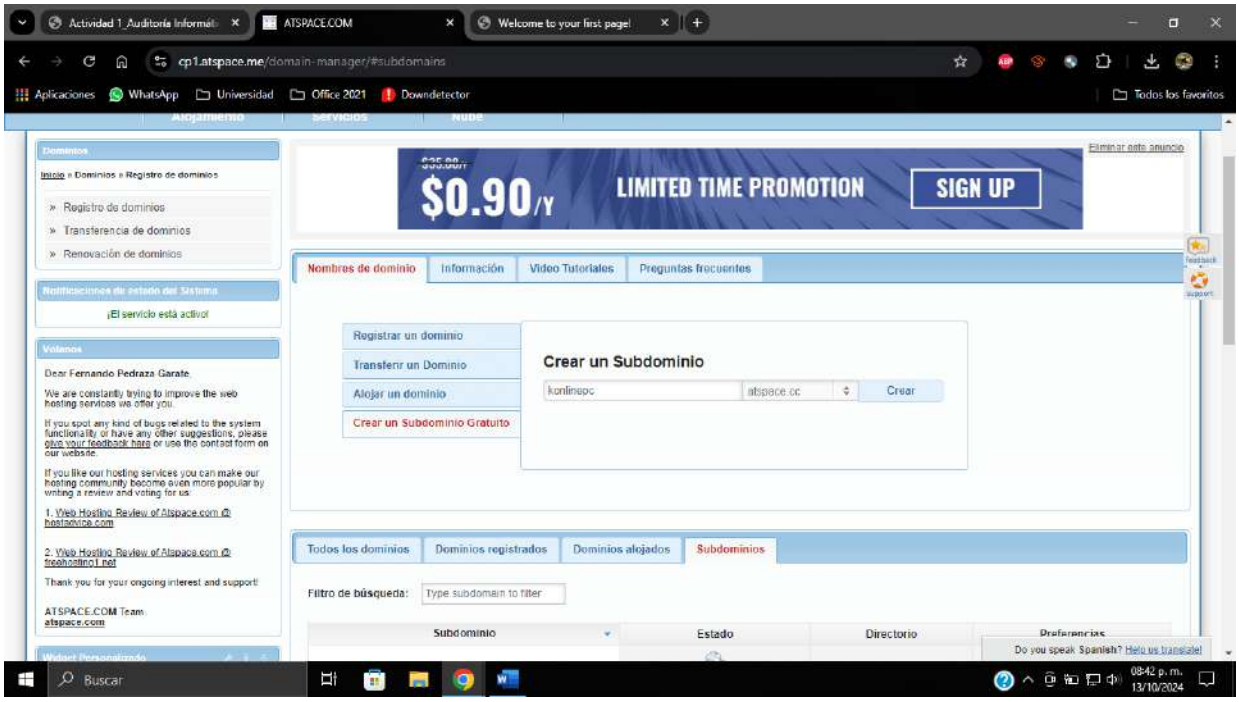
Existen casos documentados donde las empresas han sufrido graves consecuencias debido a la deserialización insegura, impactando en los servidores, comprometiendo por completo la seguridad del sistema, llevando al acceso no autorizado a datos confidenciales, con implicaciones legales y de privacidad, ocasionando que estos ataques pueden ser difíciles de detectar porque ocurren en la capa de deserialización, lo que justifica la necesidad de un manejo proactivo de esta vulnerabilidad, las organizaciones y entidades de seguridad, como OWASP (Open Web Application Security Project), reconocen la deserialización insegura como una de las principales amenazas en aplicaciones web, lo que justifica la necesidad de seguir prácticas seguras, como la de validar y filtrar los datos antes de deserializarlos, asegurando que provienen de fuentes confiables, o usar métodos más seguros para transferir y almacenar datos, implementando medidas de seguridad adicionales, como autenticación, autorización y uso de controles de acceso.

Ataque al sitio.

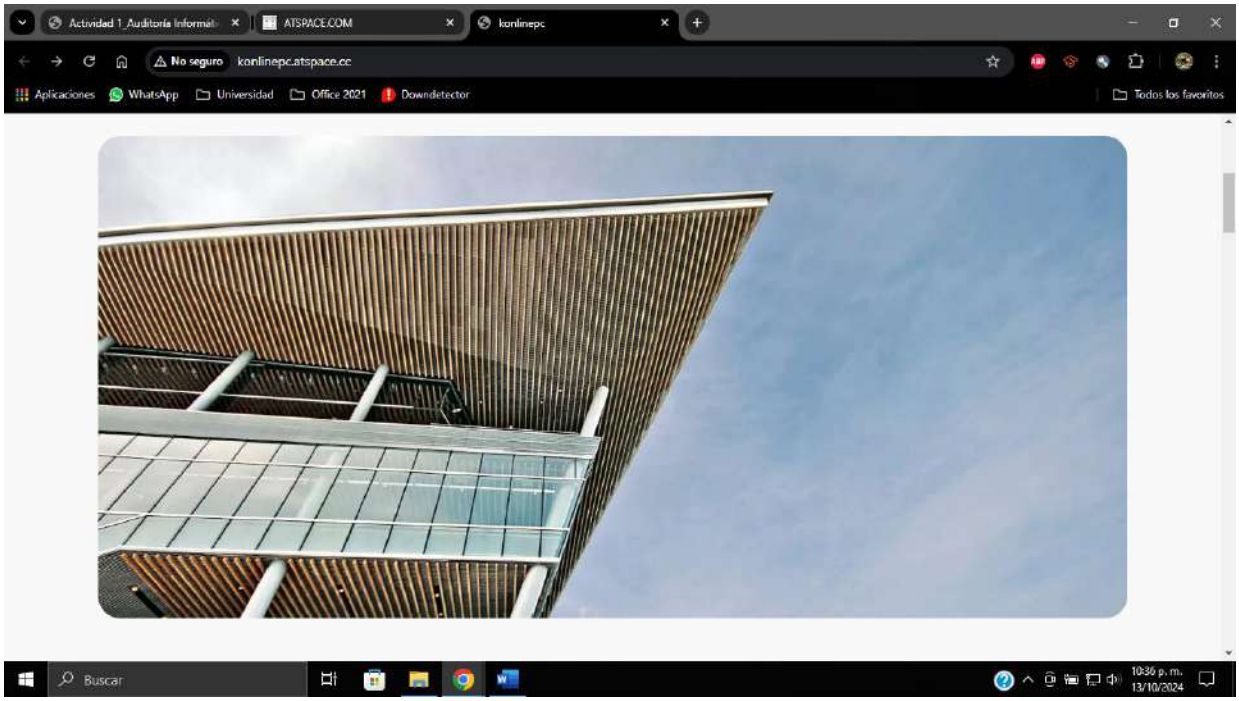
Etapa 1

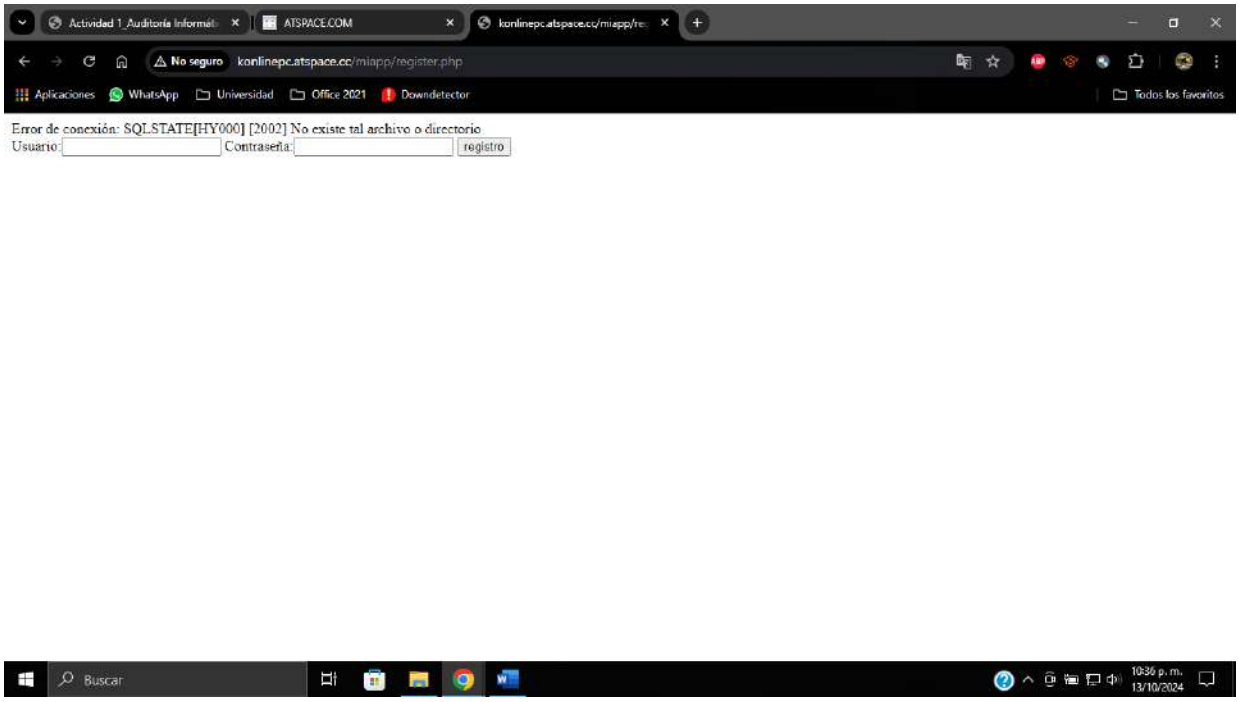




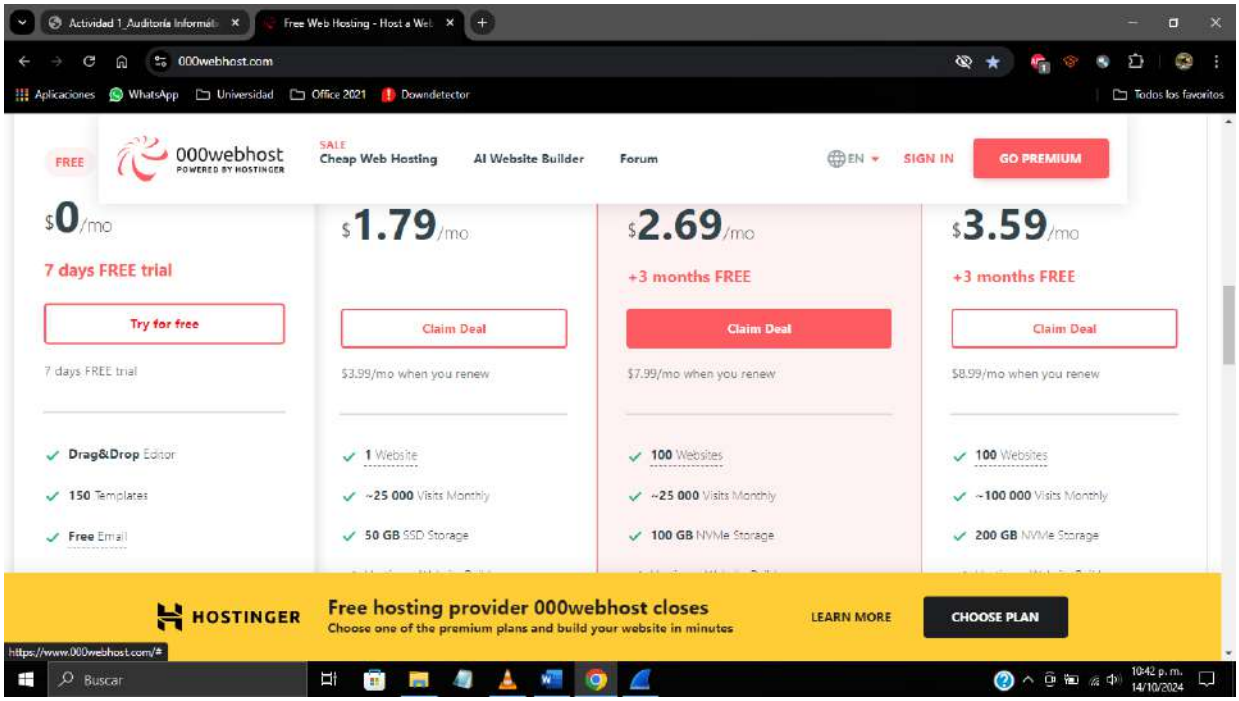


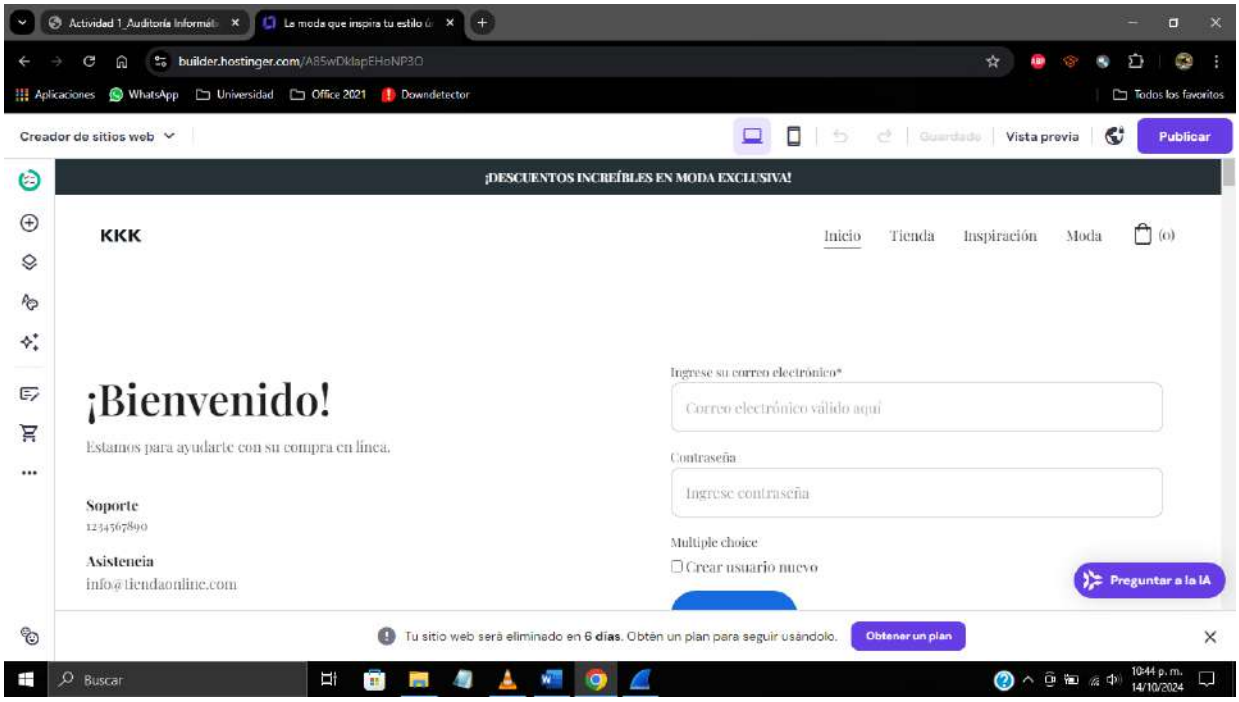
Se carga el proyecto donde se creó la página a monitorear con wireshark



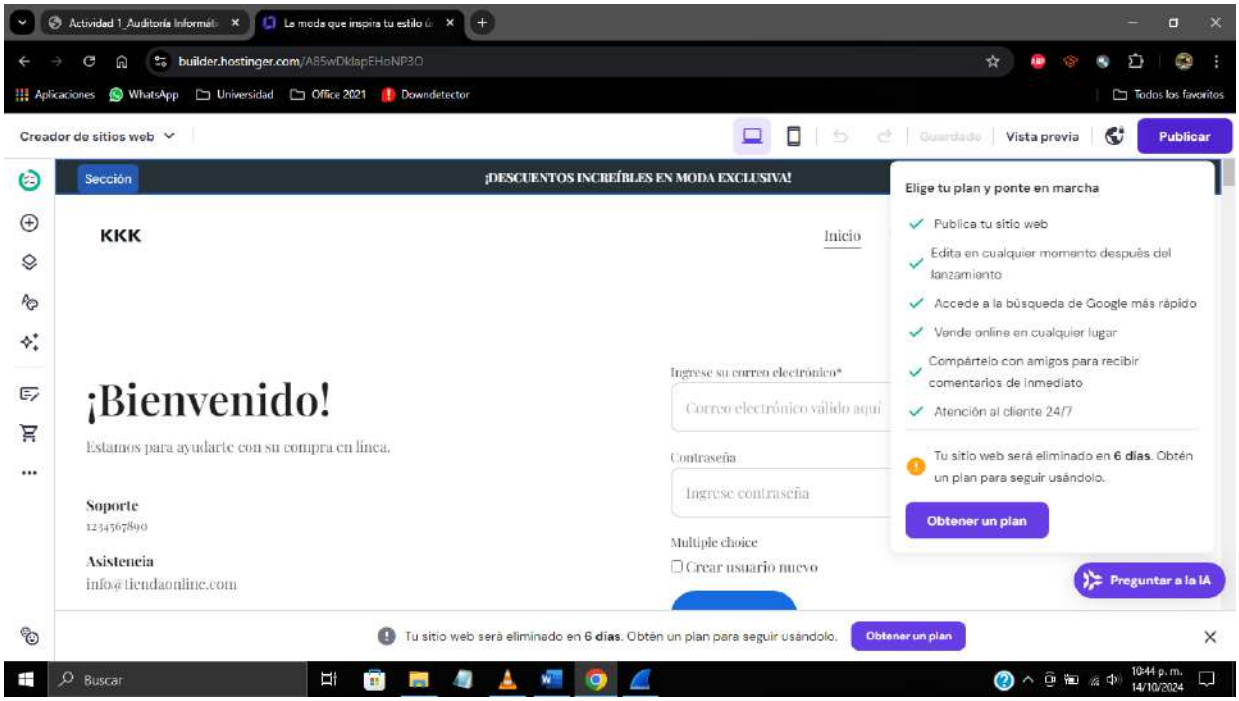


Sin poder cargar el dominio a internet en la versión gratuita

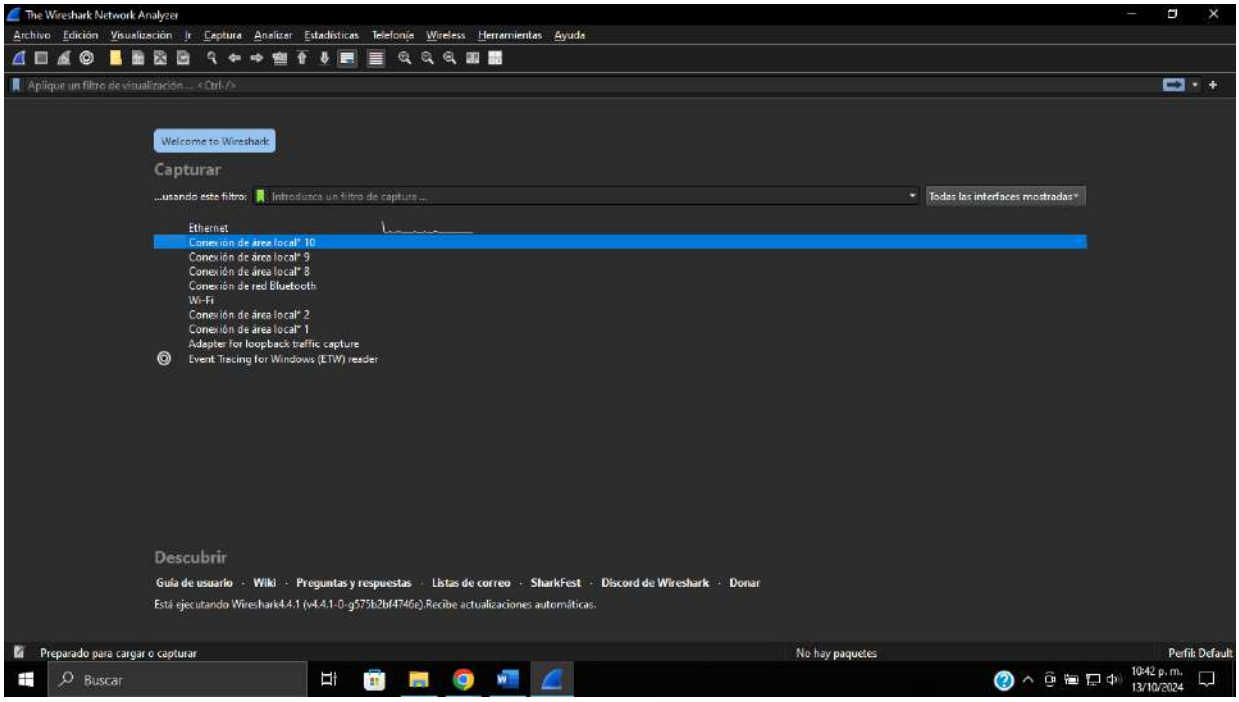


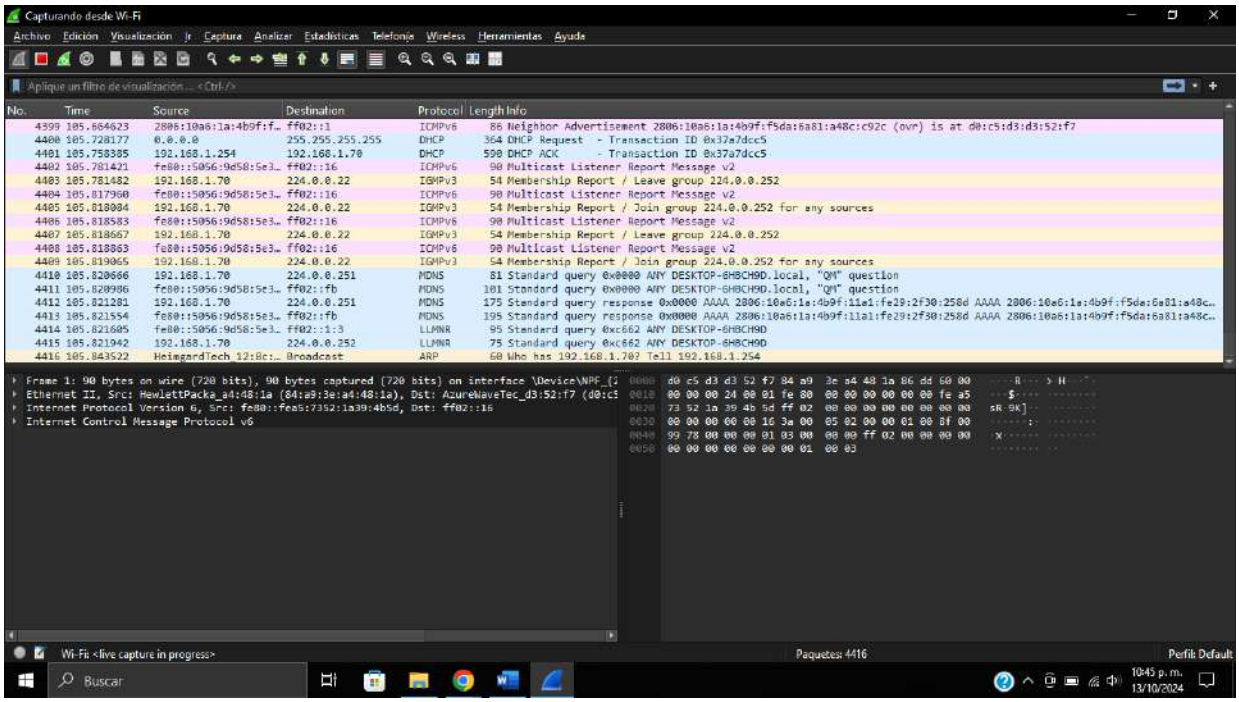


Intentando con Hostinger, obteniendo el mismo resultado



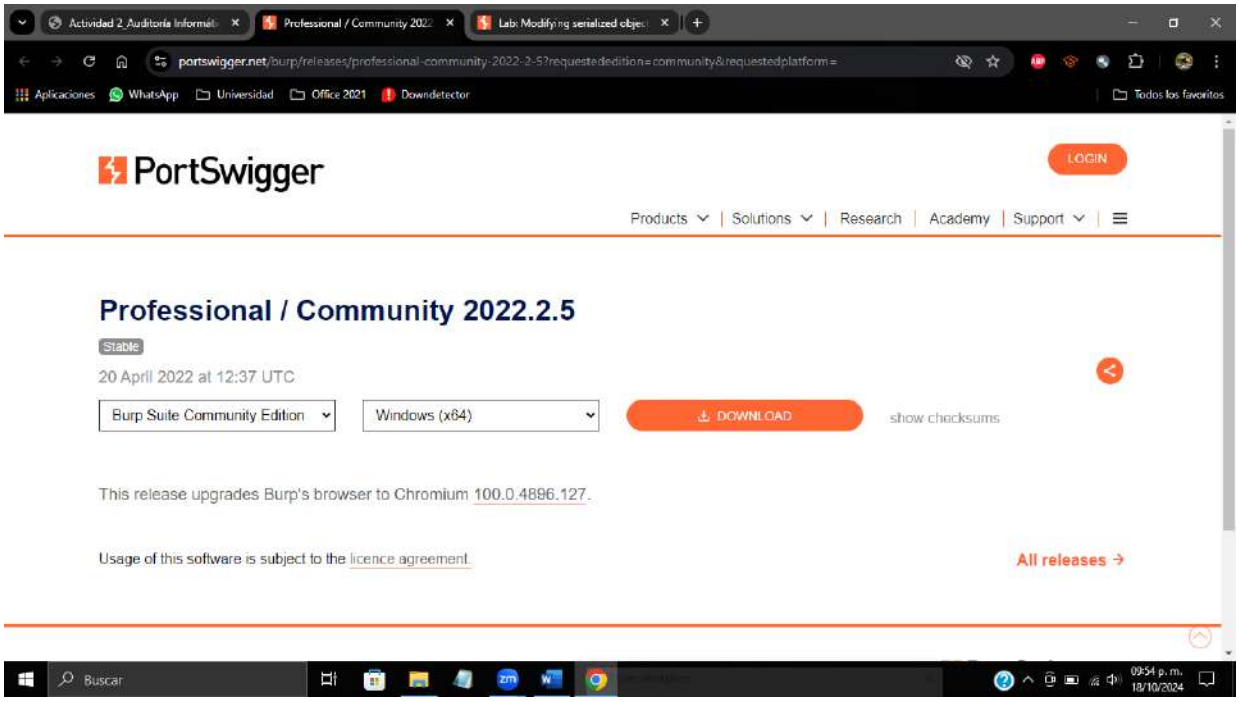
Se instala y se abre wireshark



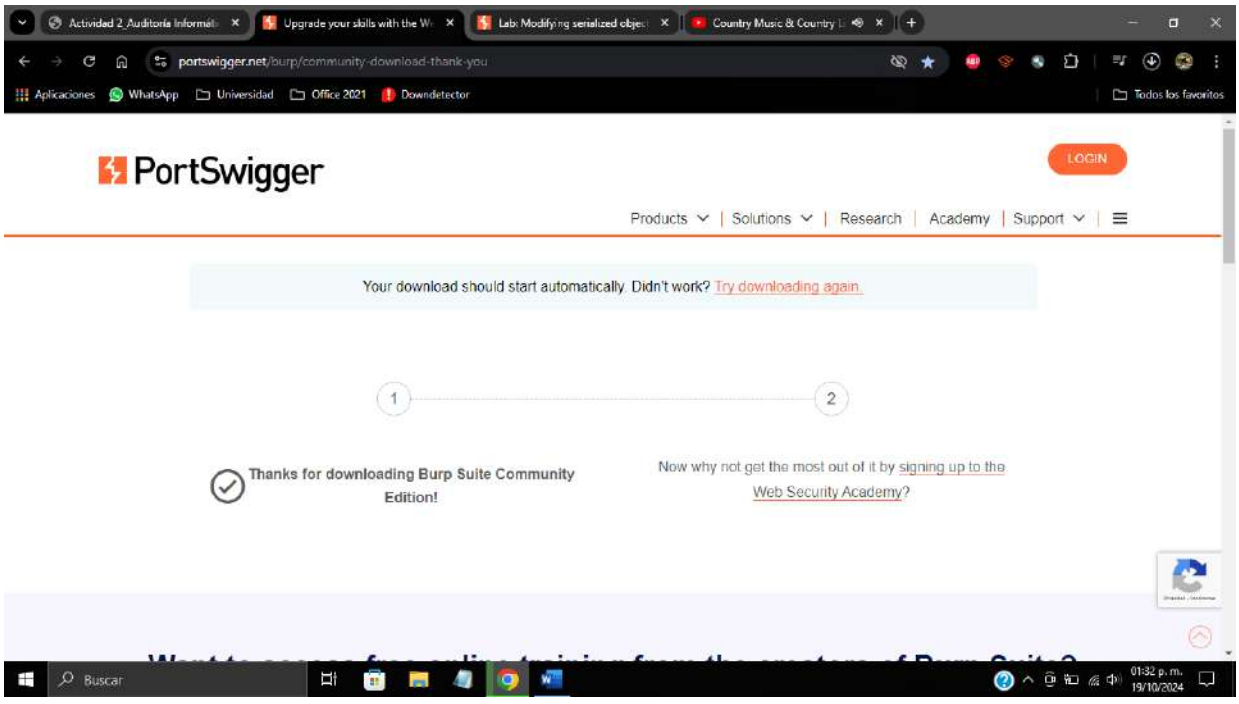


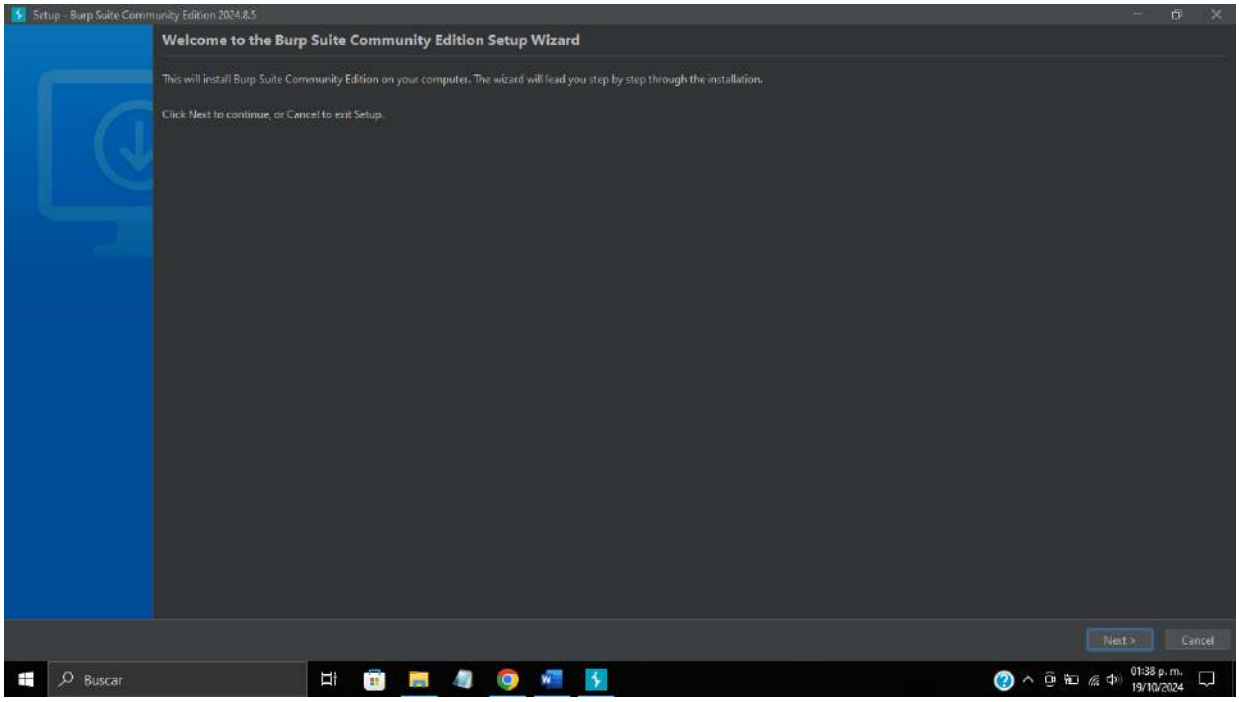
Se ingresa al apartado de WiFi para monitorear el tráfico de red

Etapa 2 – Ataque al sitio

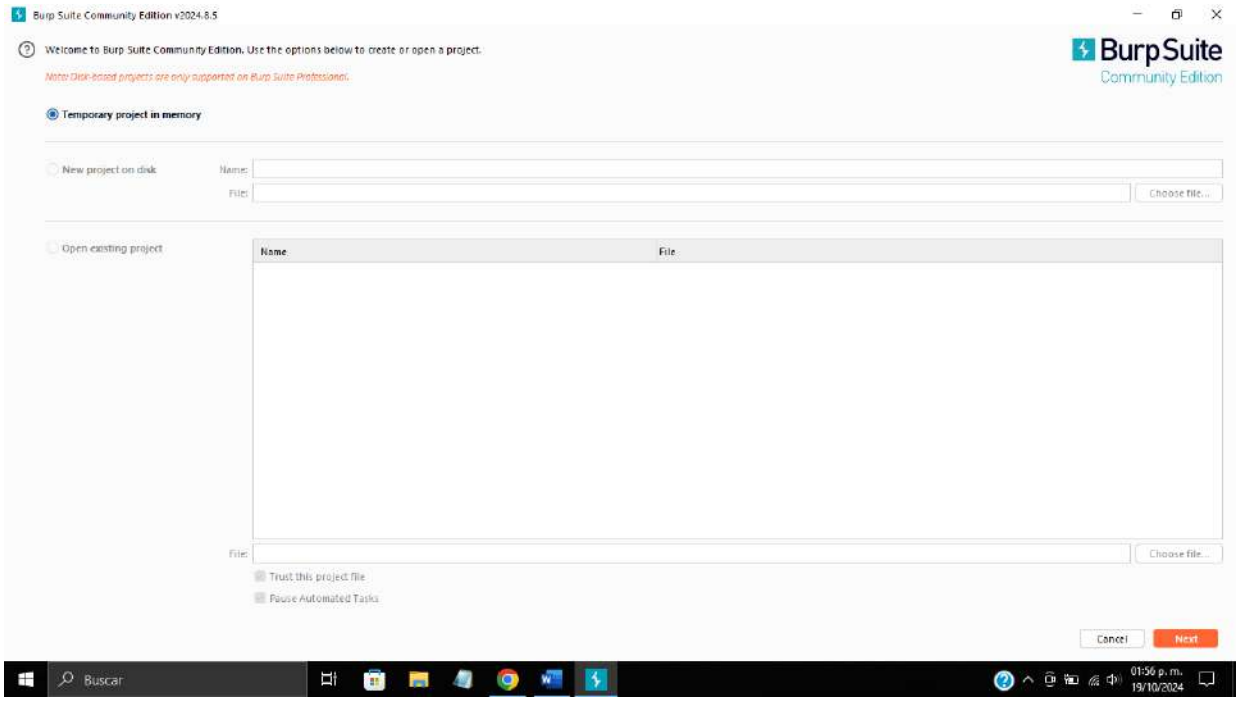


Se ingresa a PortSwigger y se descarga el programa Burp Suite Comunity Edition

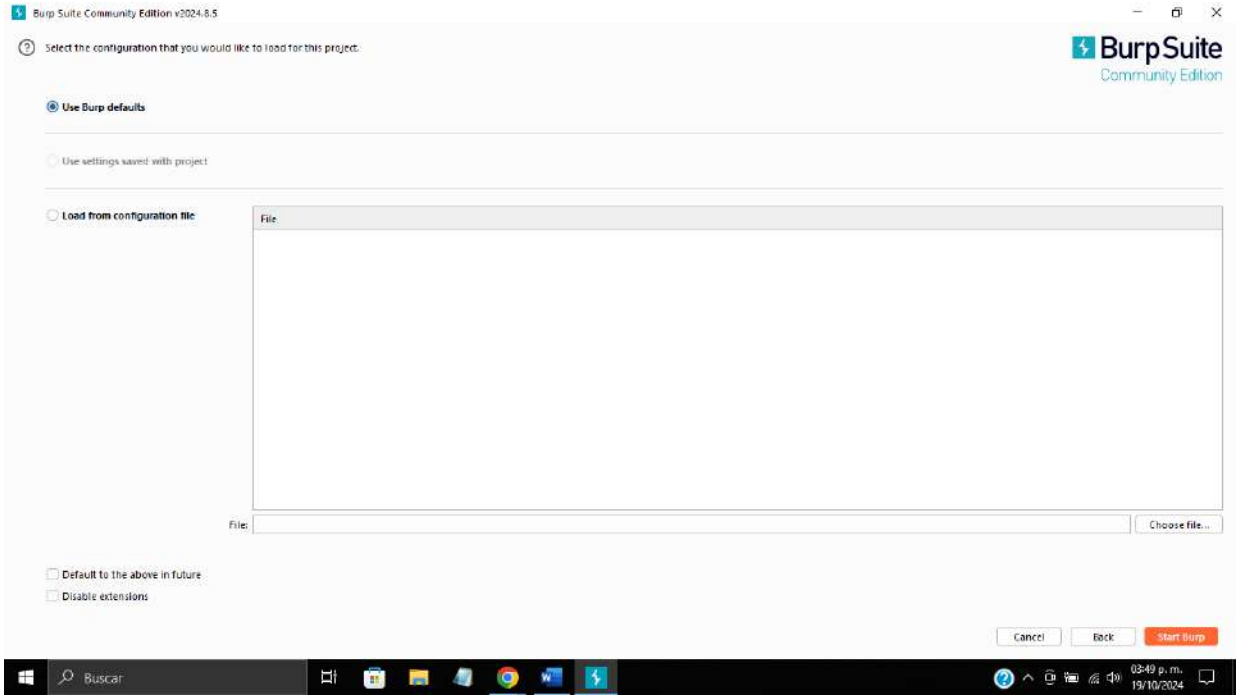




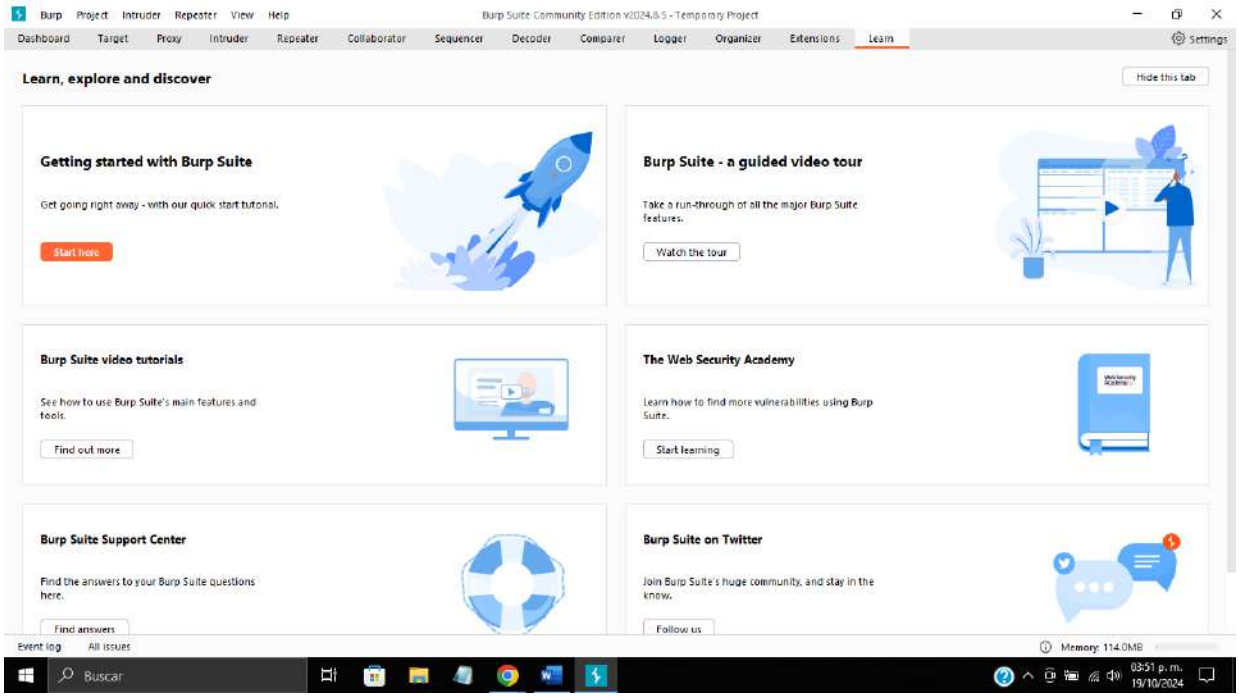
Para posteriormente instalarlo en el equipo y abrir el programa

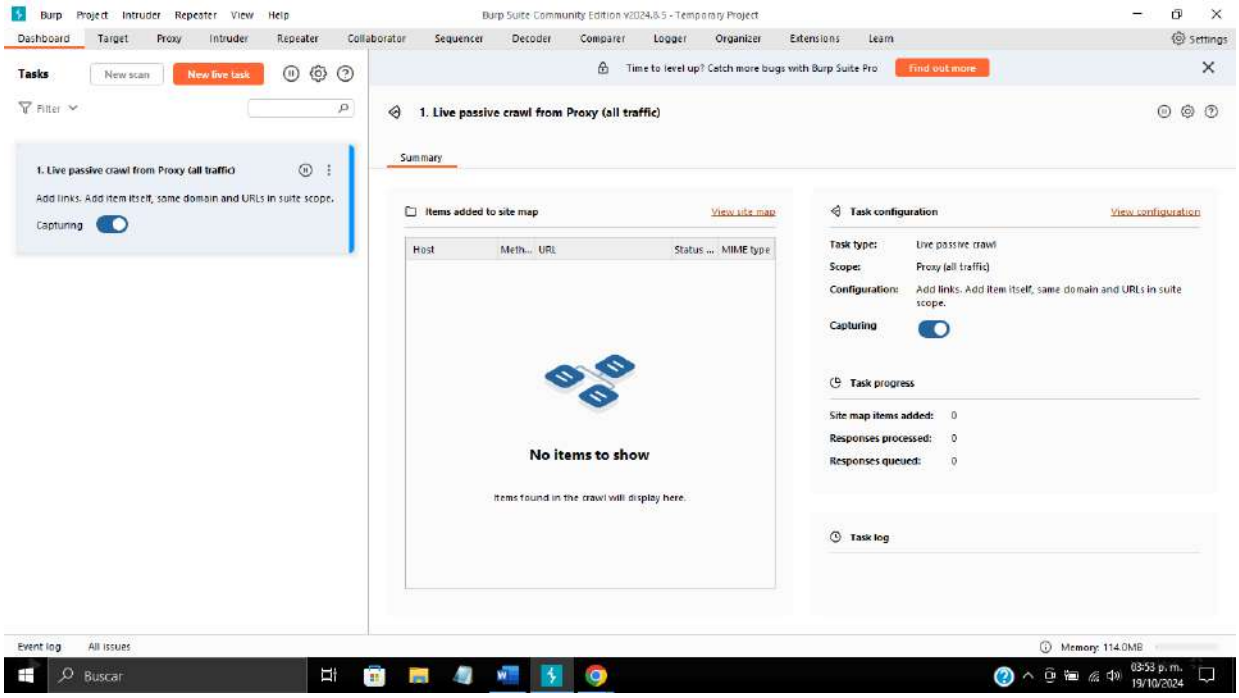


Se deja todo por default

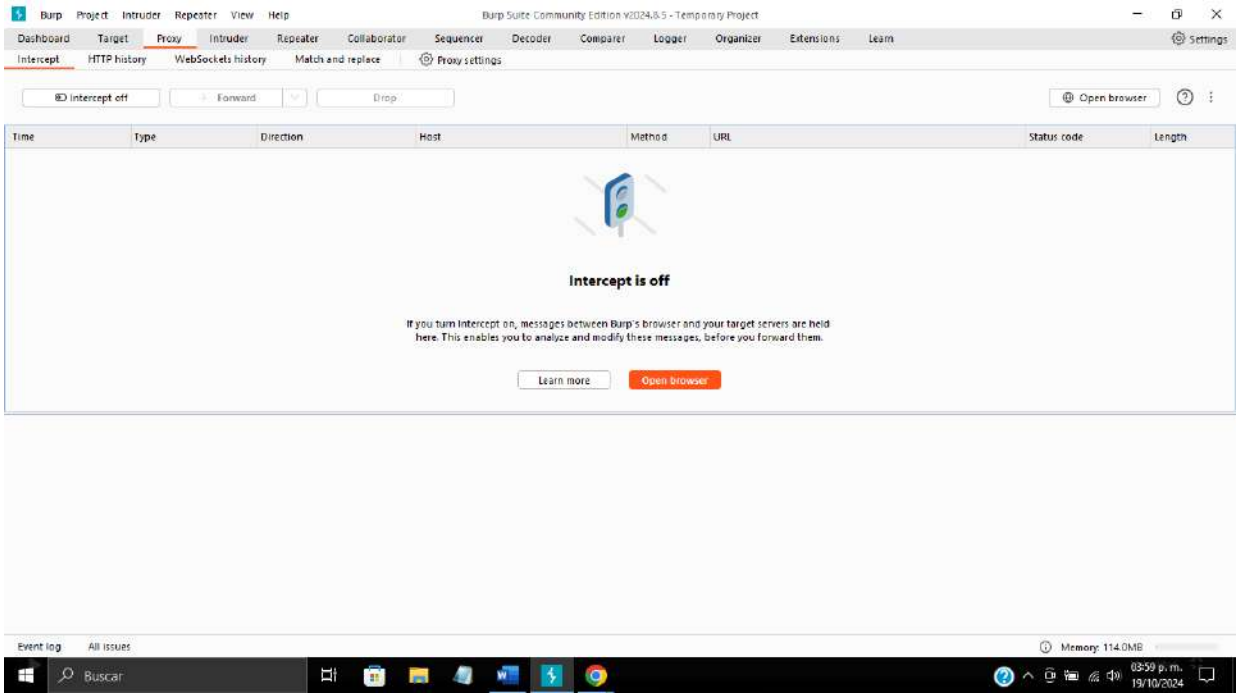


Se inicia el Burp

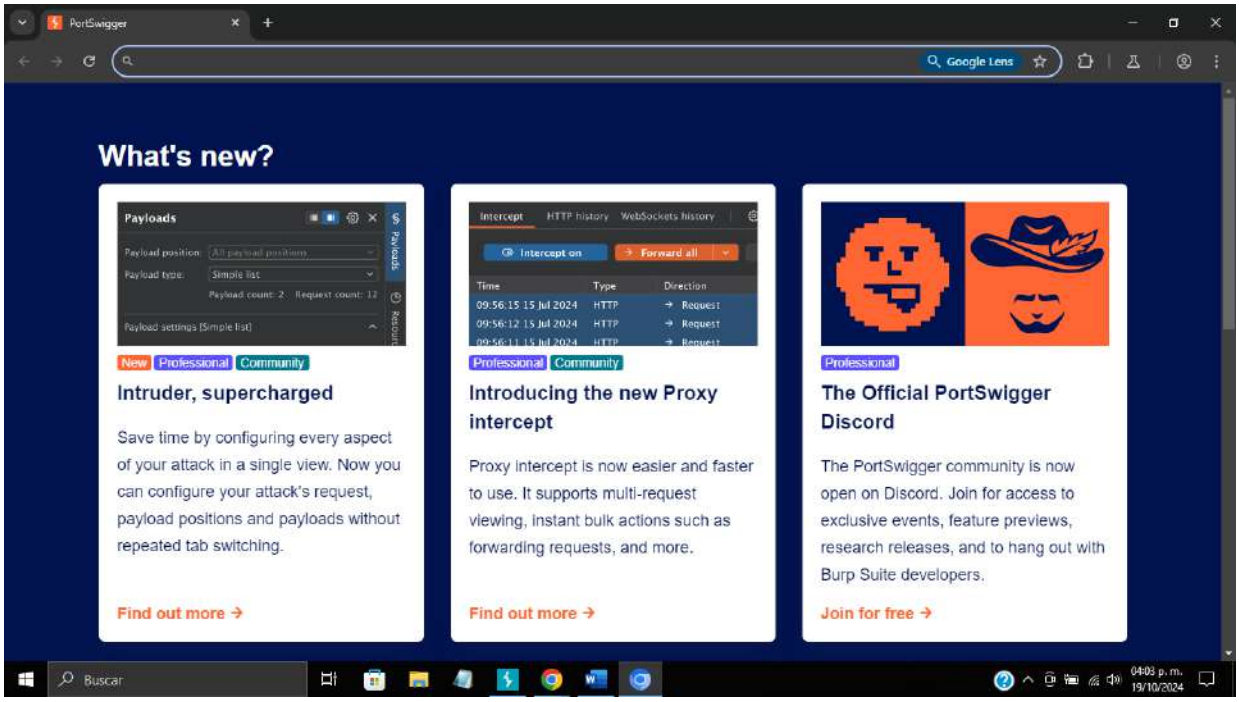


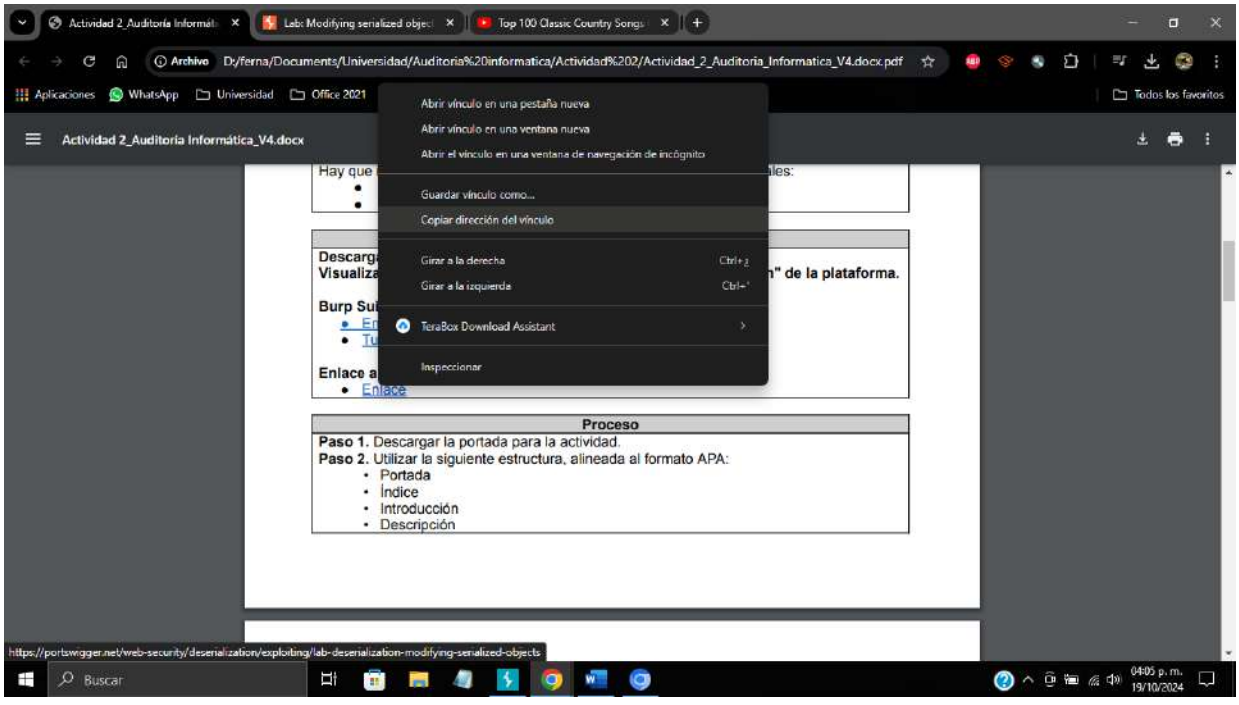


Se ingresa al Dashboard

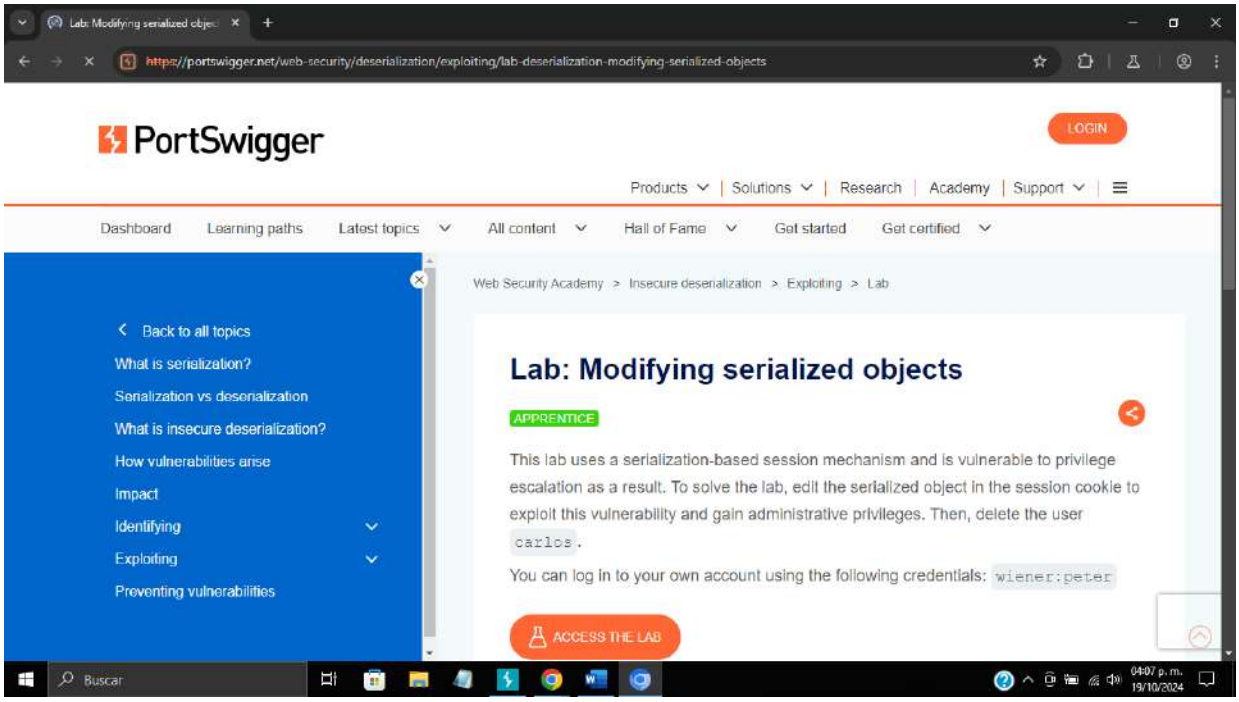


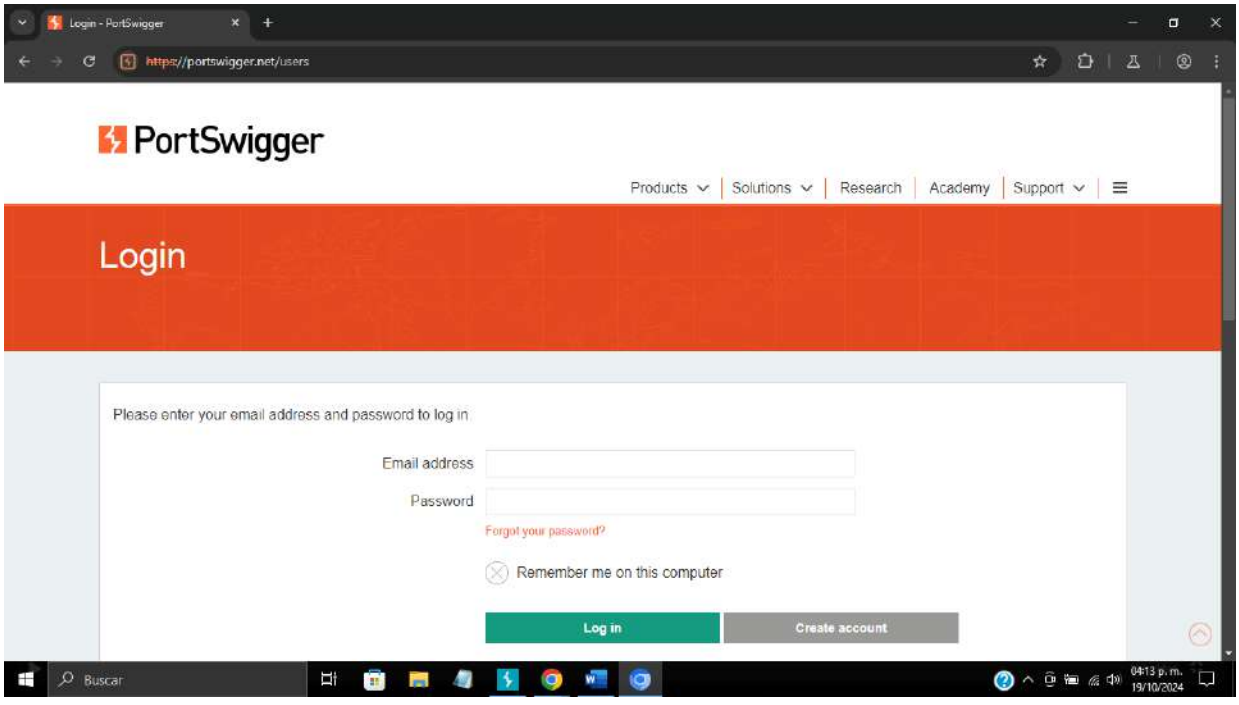
Se selecciona el apartado proxy para abrir el navegador



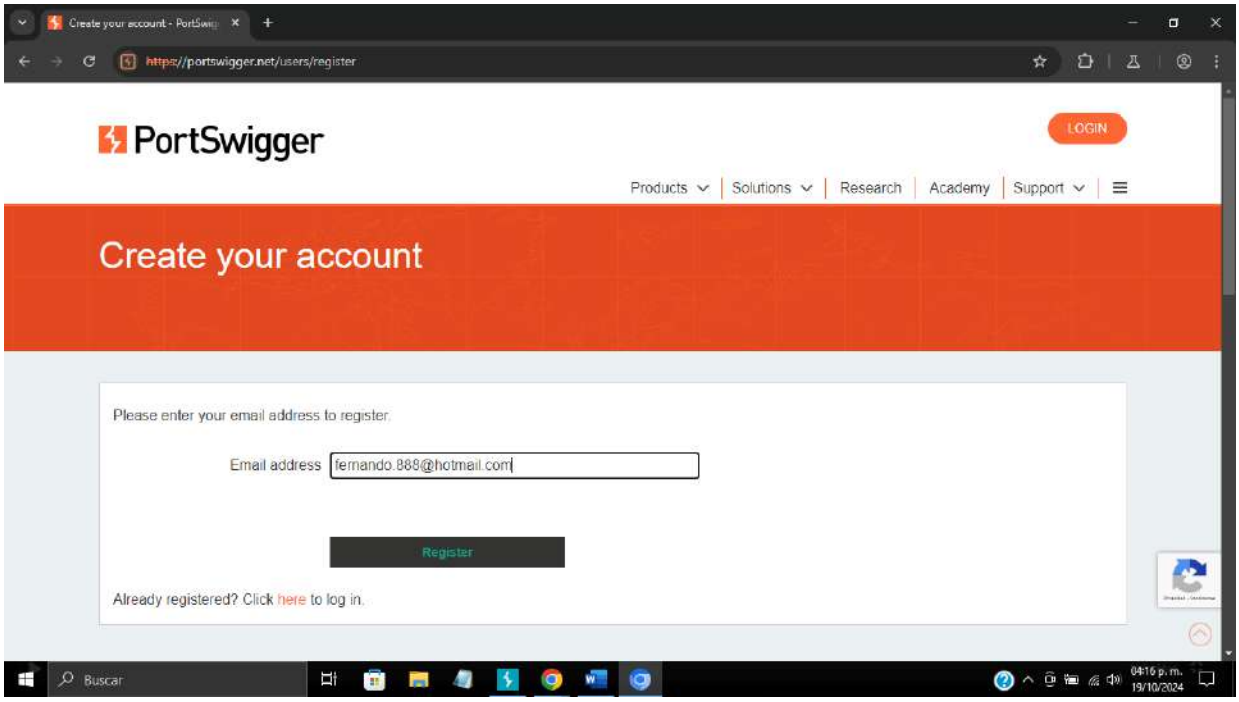


Se copia el enlace de la práctica y se pega el enlace en el navegador que se abrió de Burt para la practica

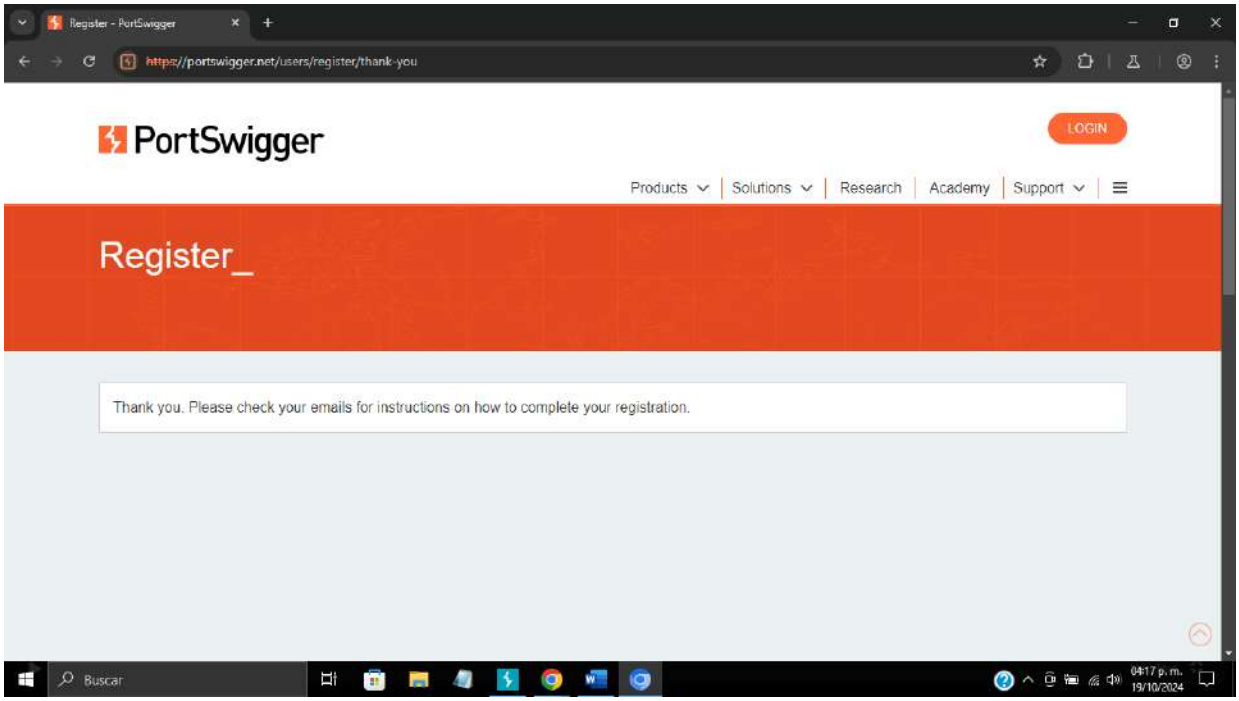


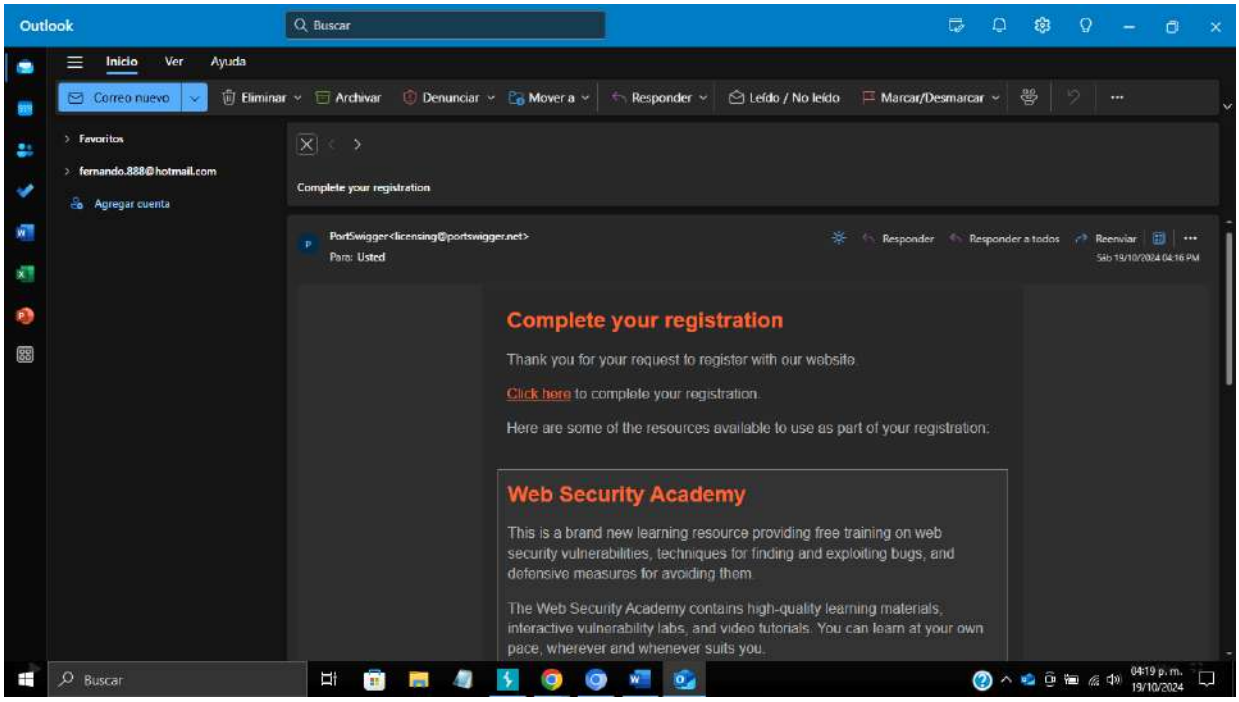


Se crea una nueva cuenta de usuario en PortSwigger para accesar al laboratorio de practica

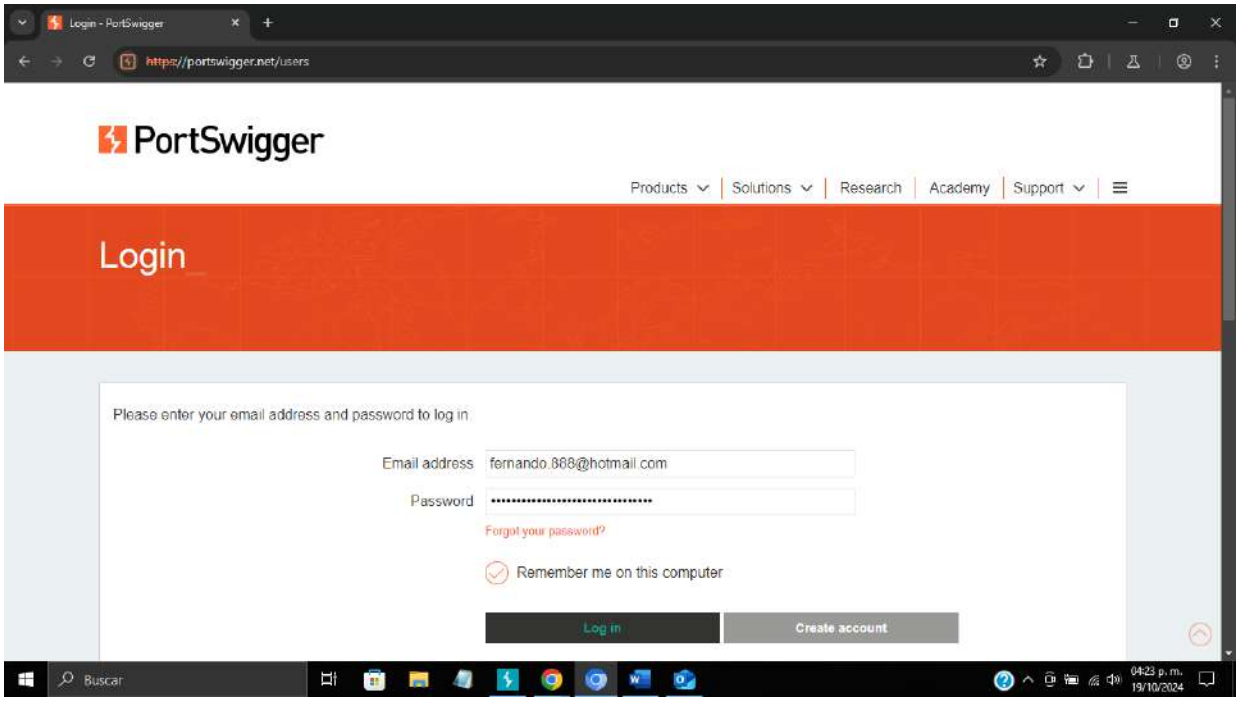


Y se siguen las instrucciones para completar el registro

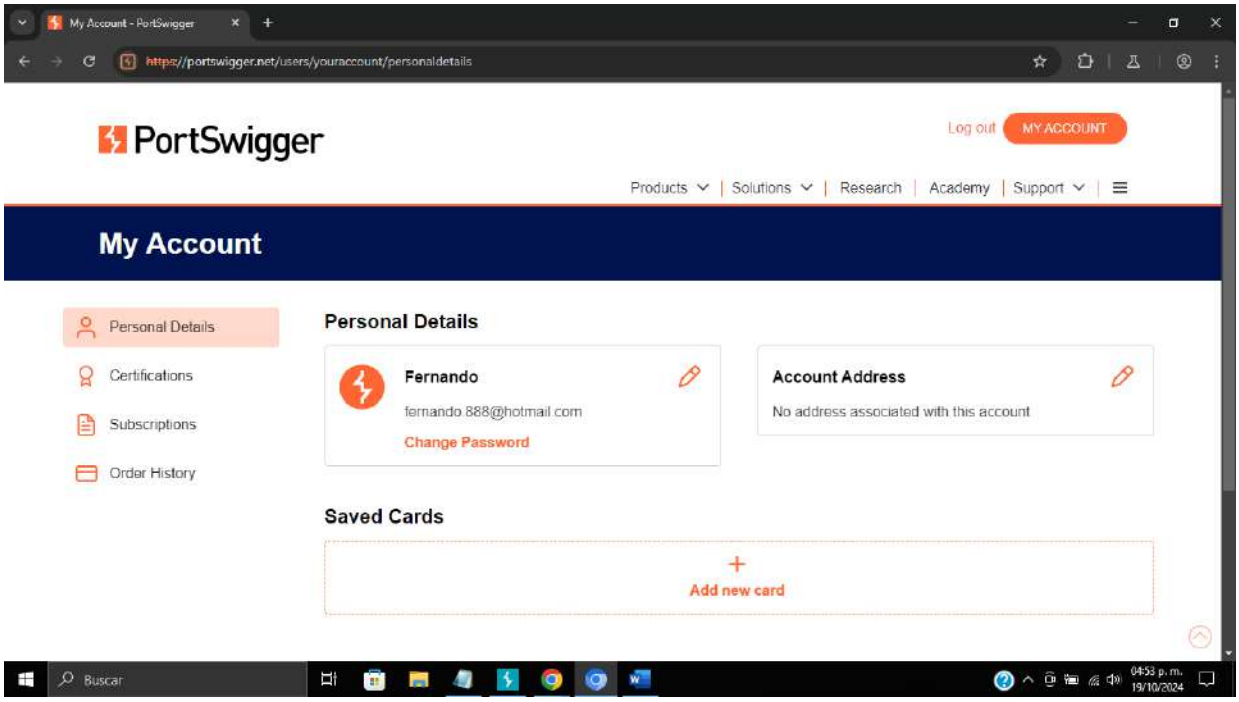




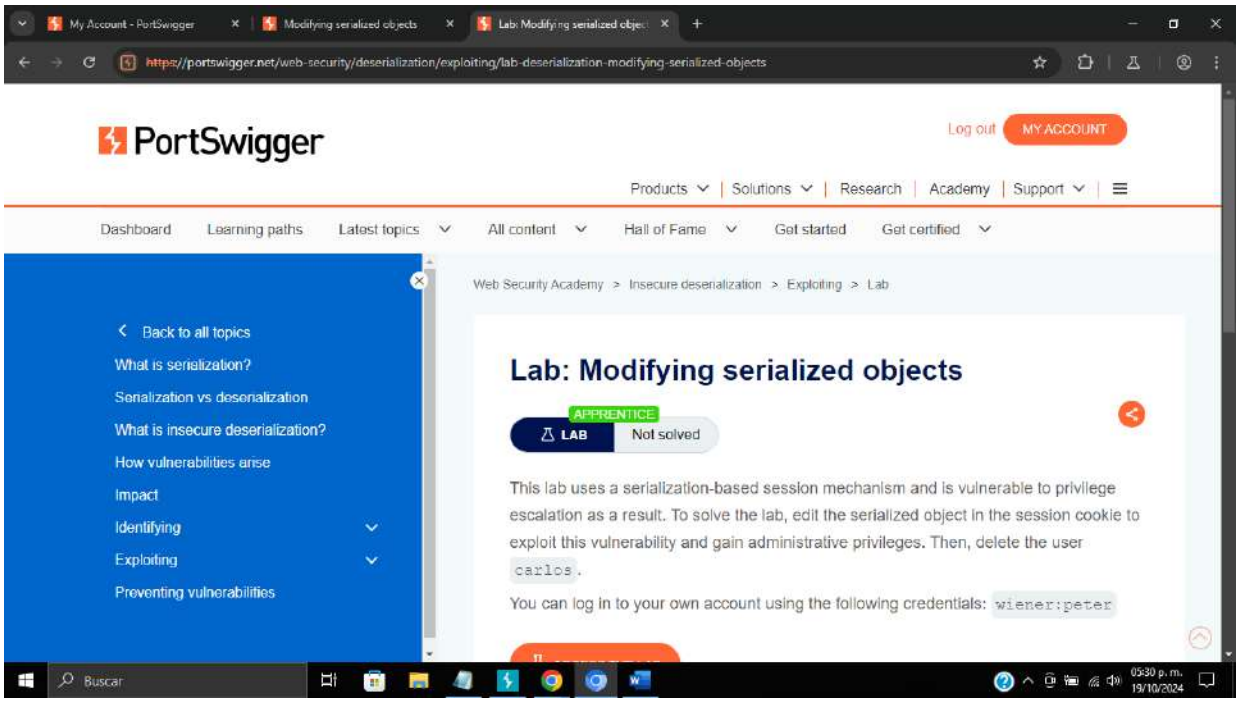
Confirmando la cuenta desde el correo



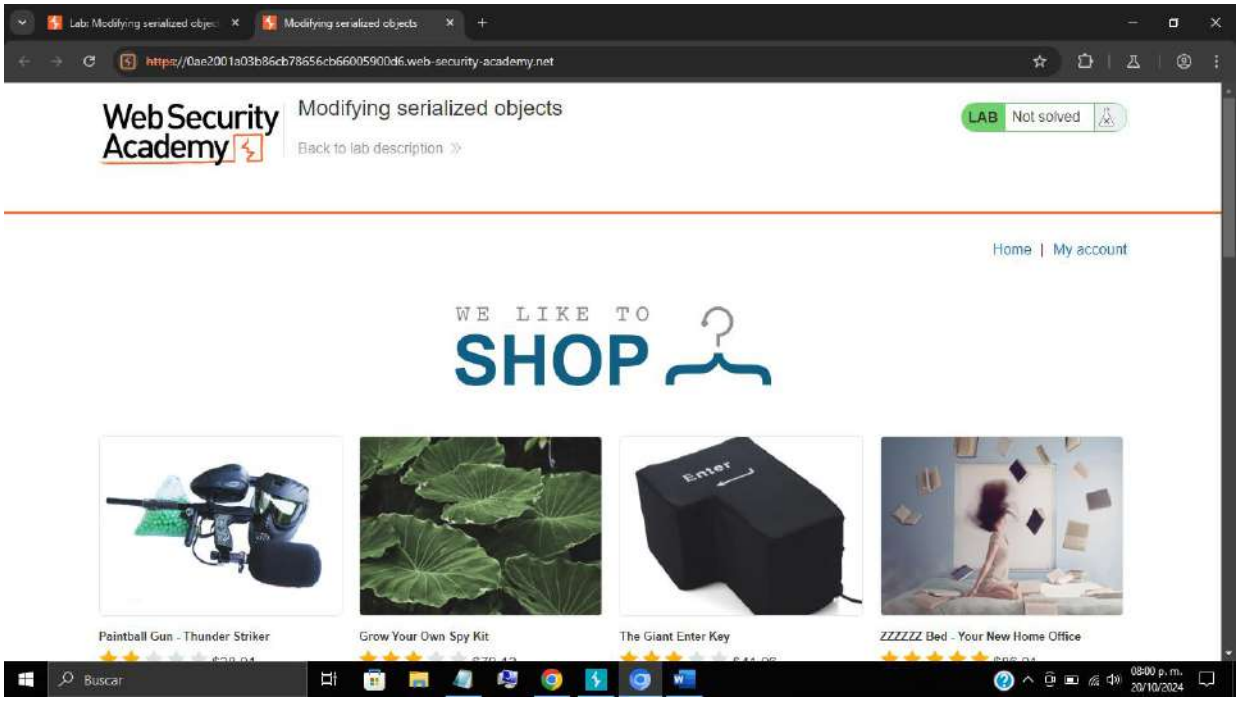
Se inicia sesión



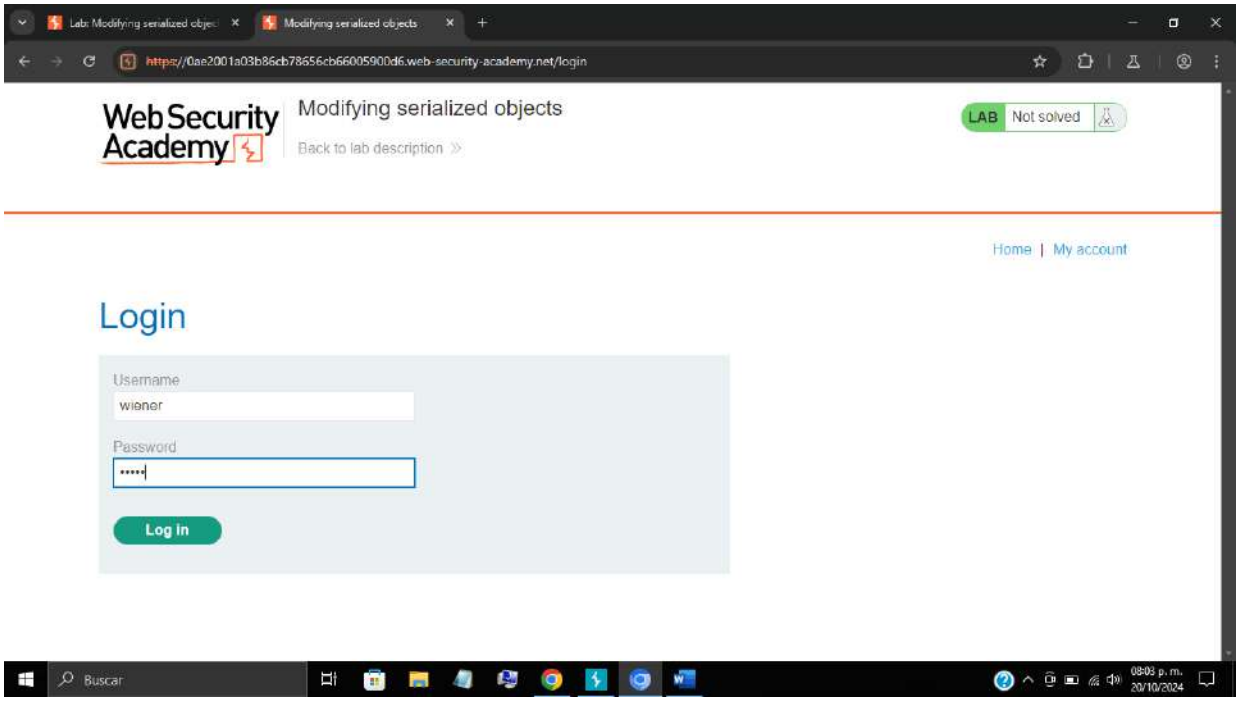
Una vez que se tiene el acceso se ingresa al enlace de la practica

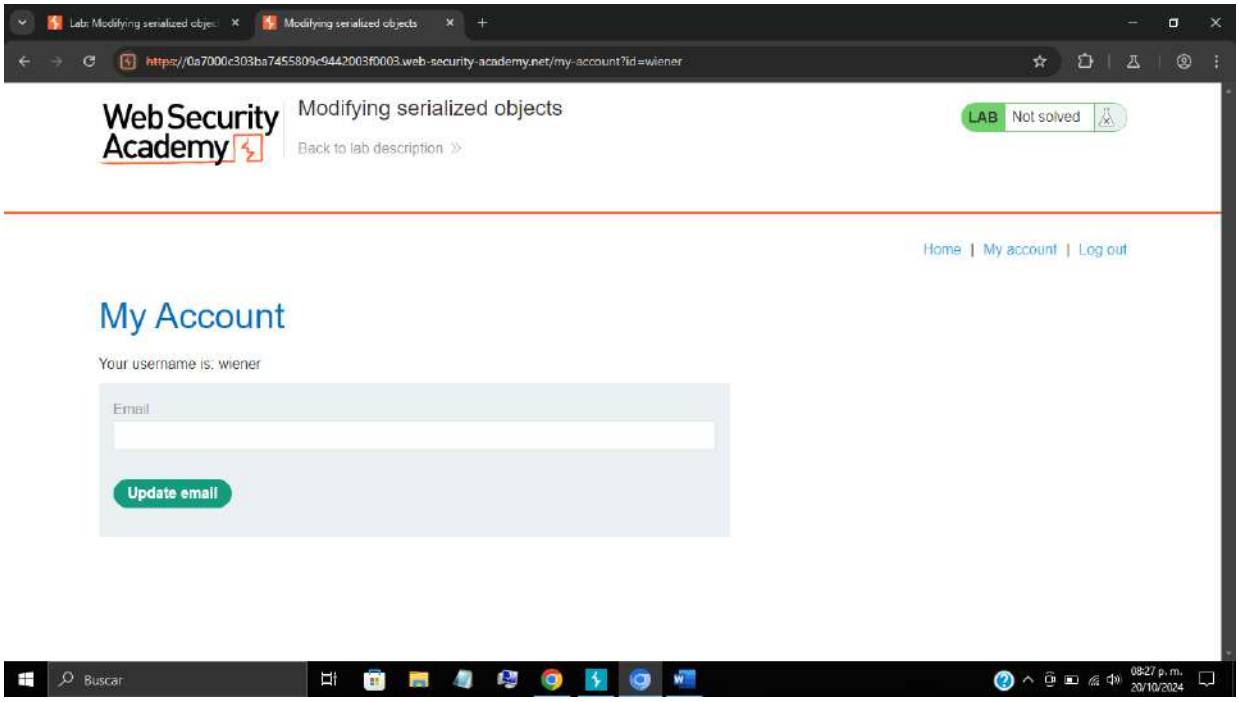


Y se ingresa al laboratorio de practica

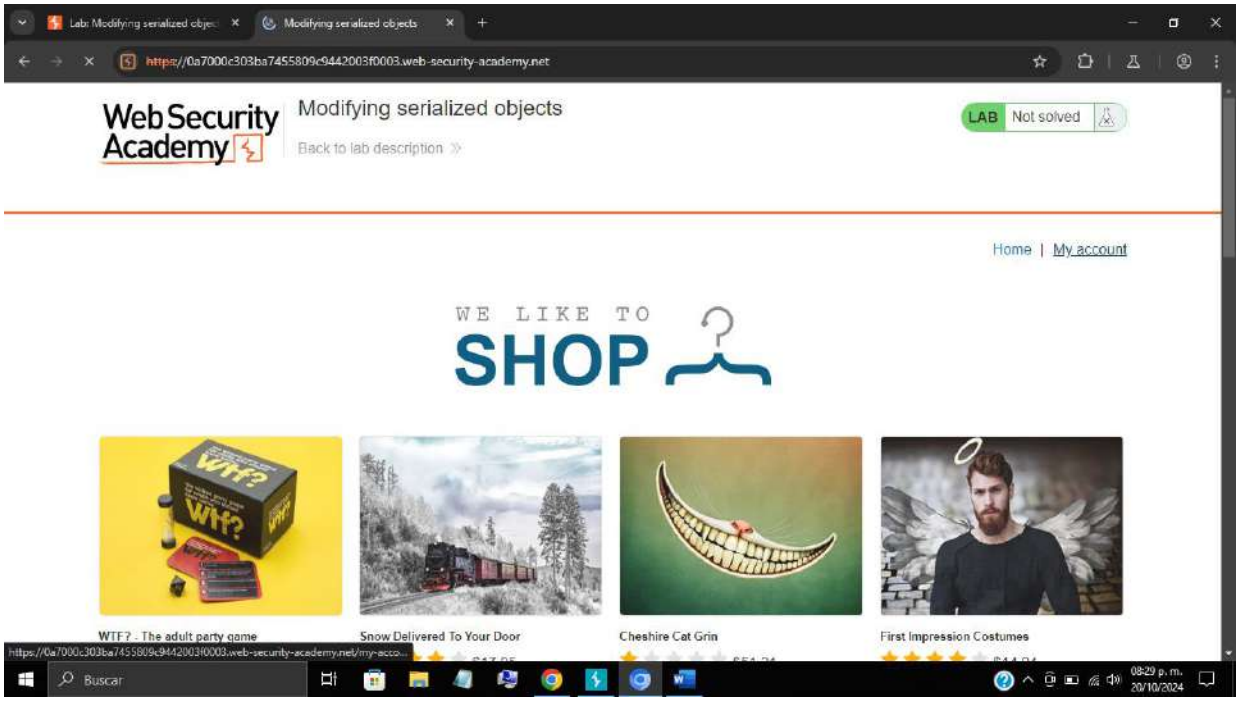


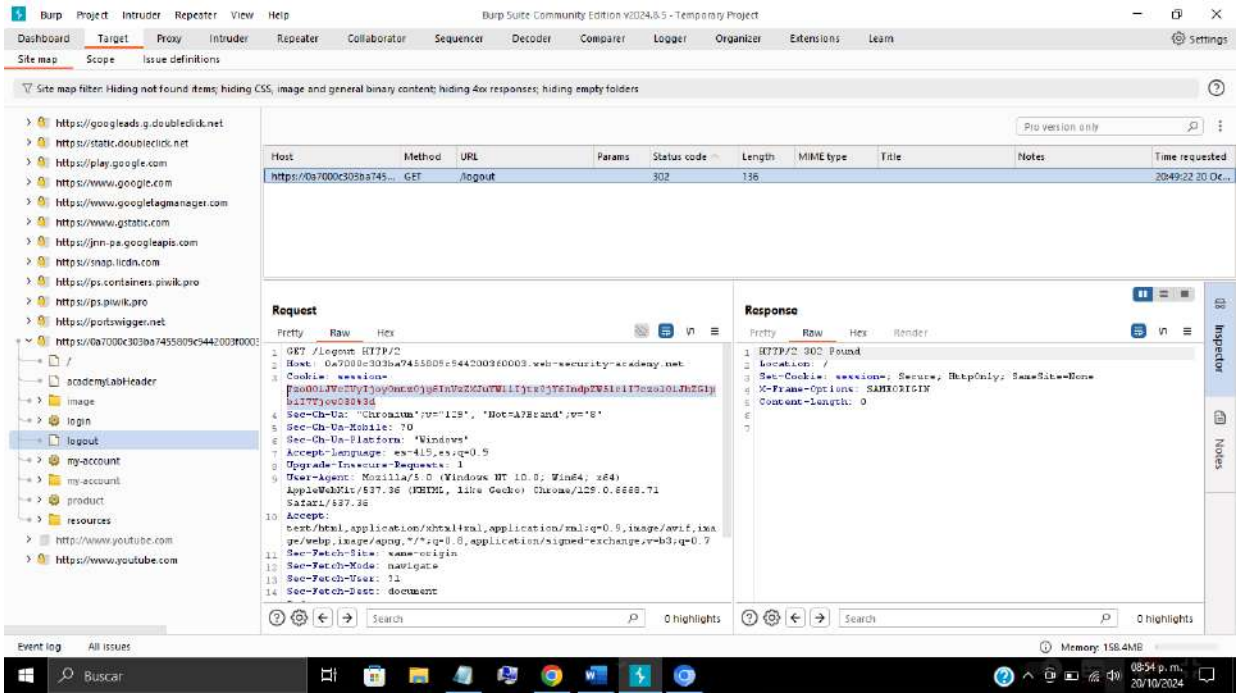
Para posteriormente ingresar el usuario y contraseña proporcionados



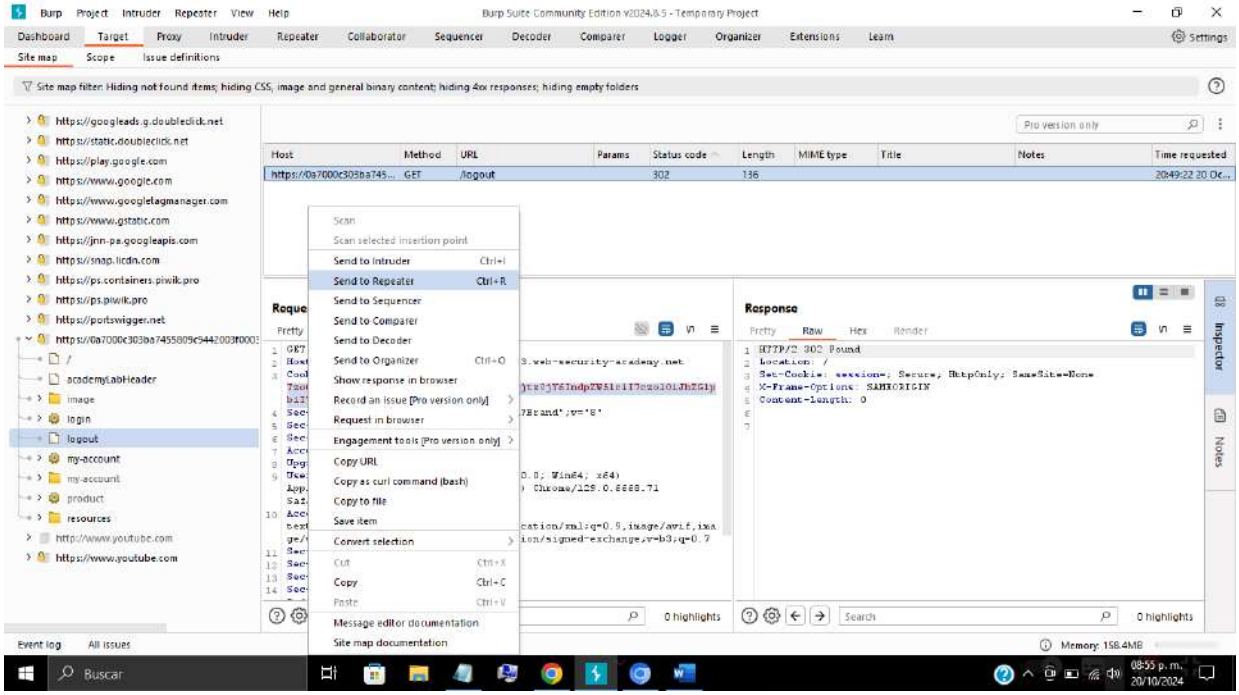


Una vez ingresado en el apartado log out cerramos sesión

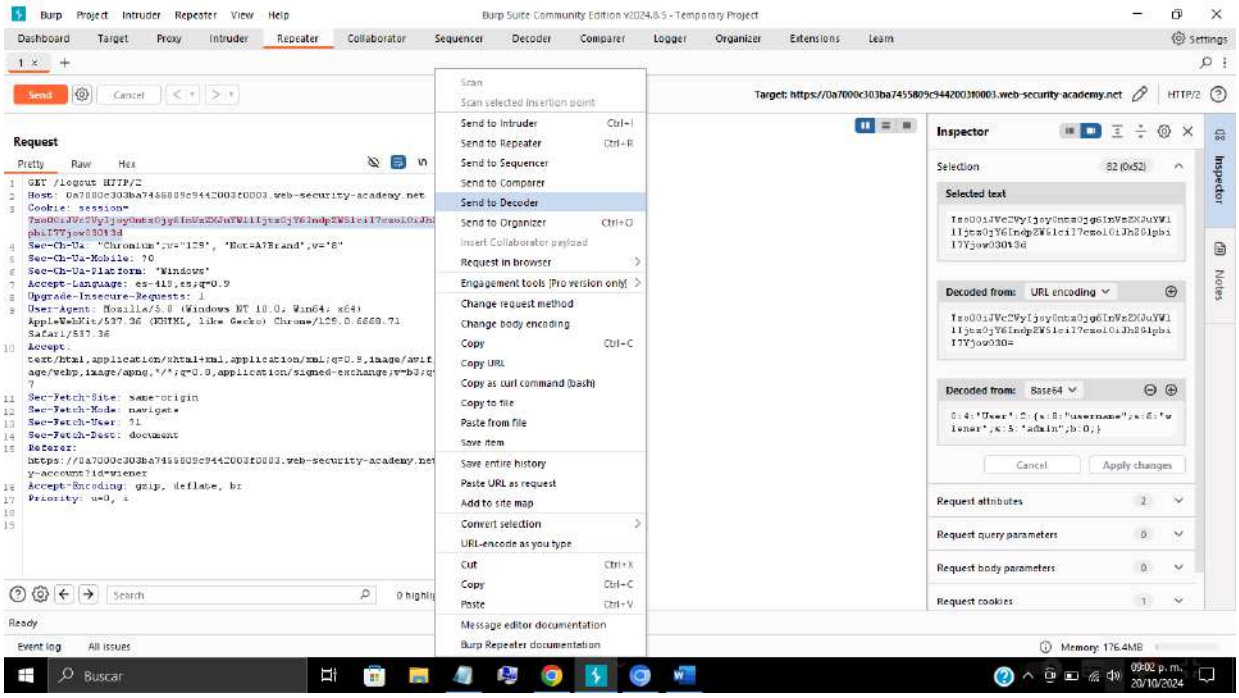




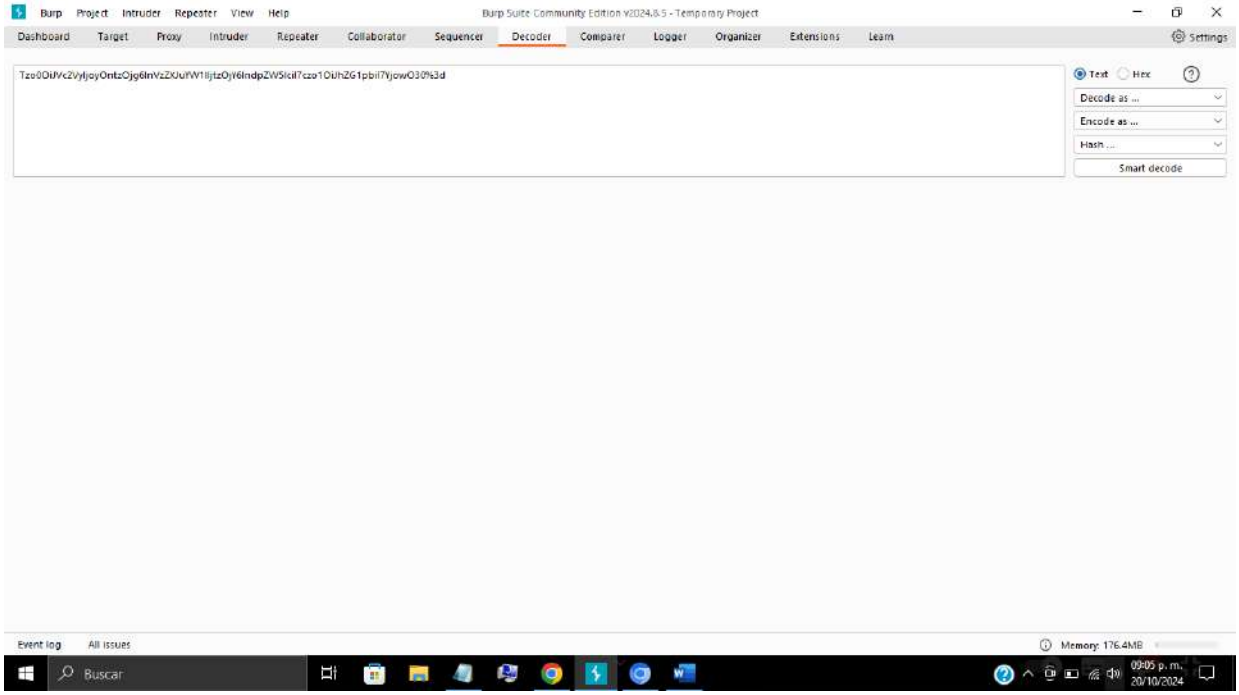
En Burt se puede observar en el target y sitio del mapa la alerta de vulnerabilidad



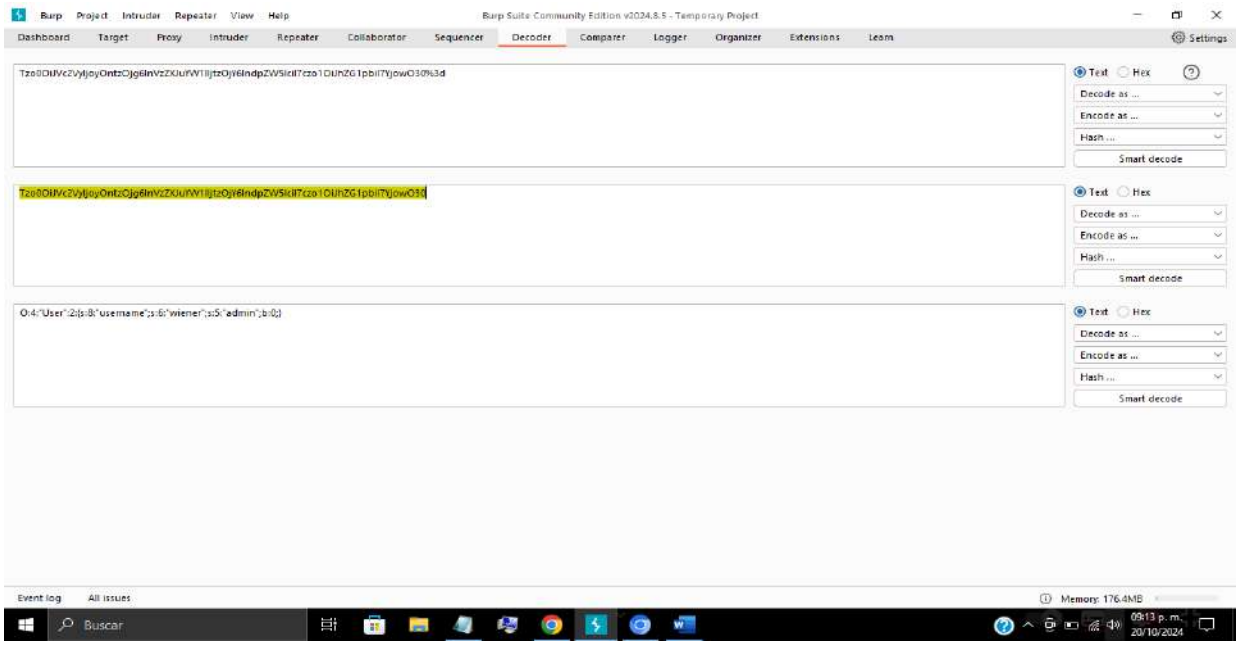
Se envía al repetidor



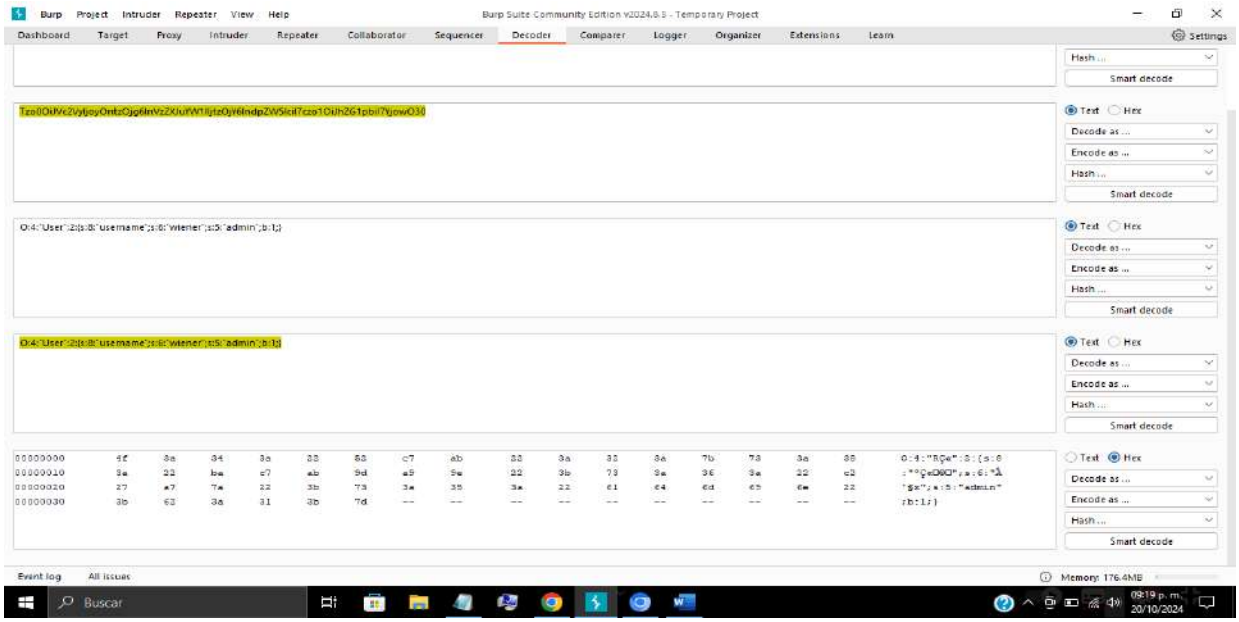
Se envía el enlace al decodificador



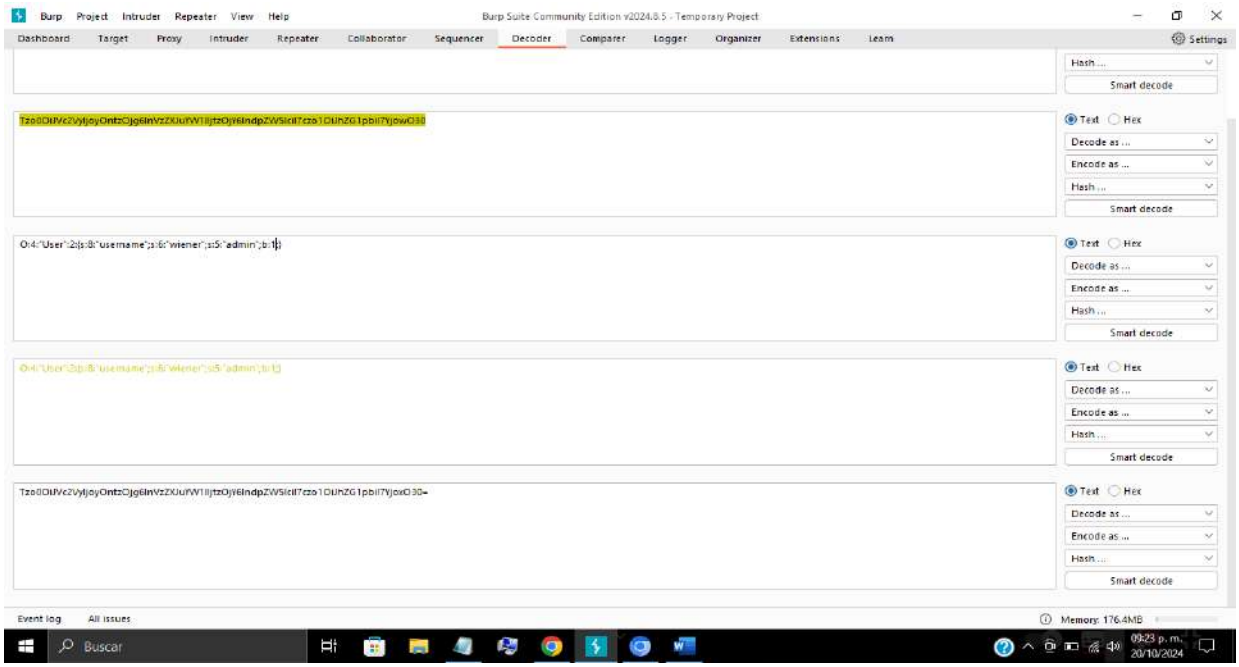
Se modifica la terminación del enlace eliminando %3d



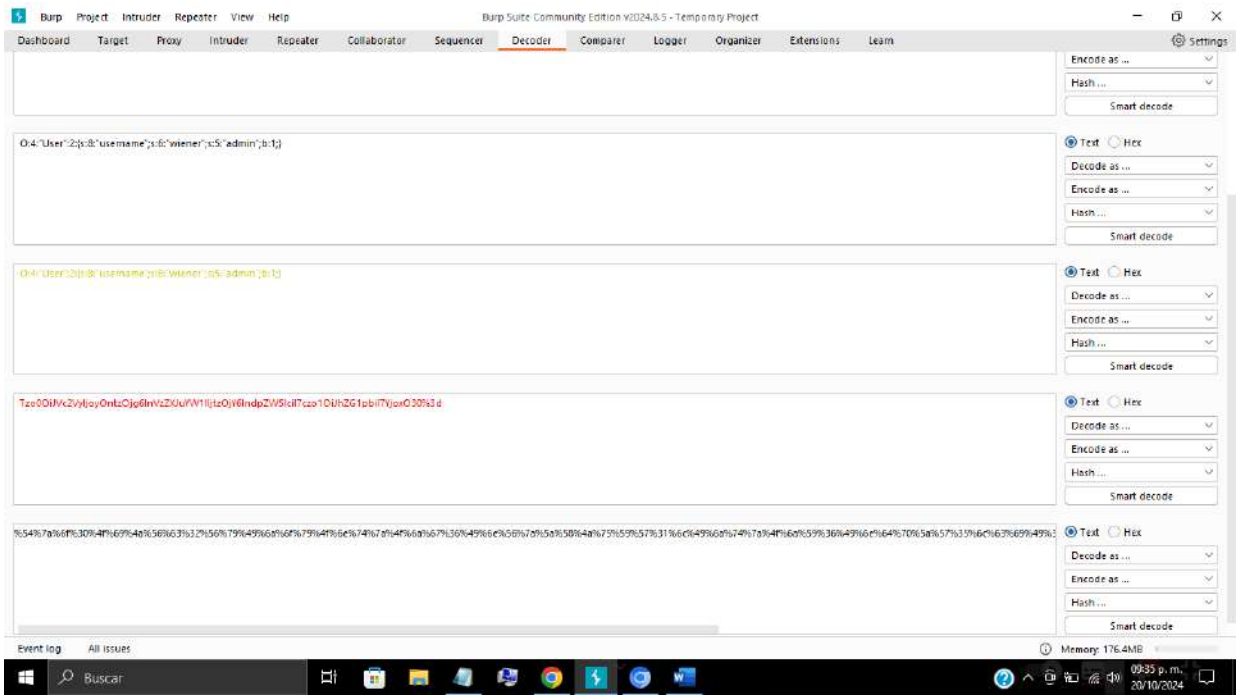
Para posteriormente decodificarlo a Base64



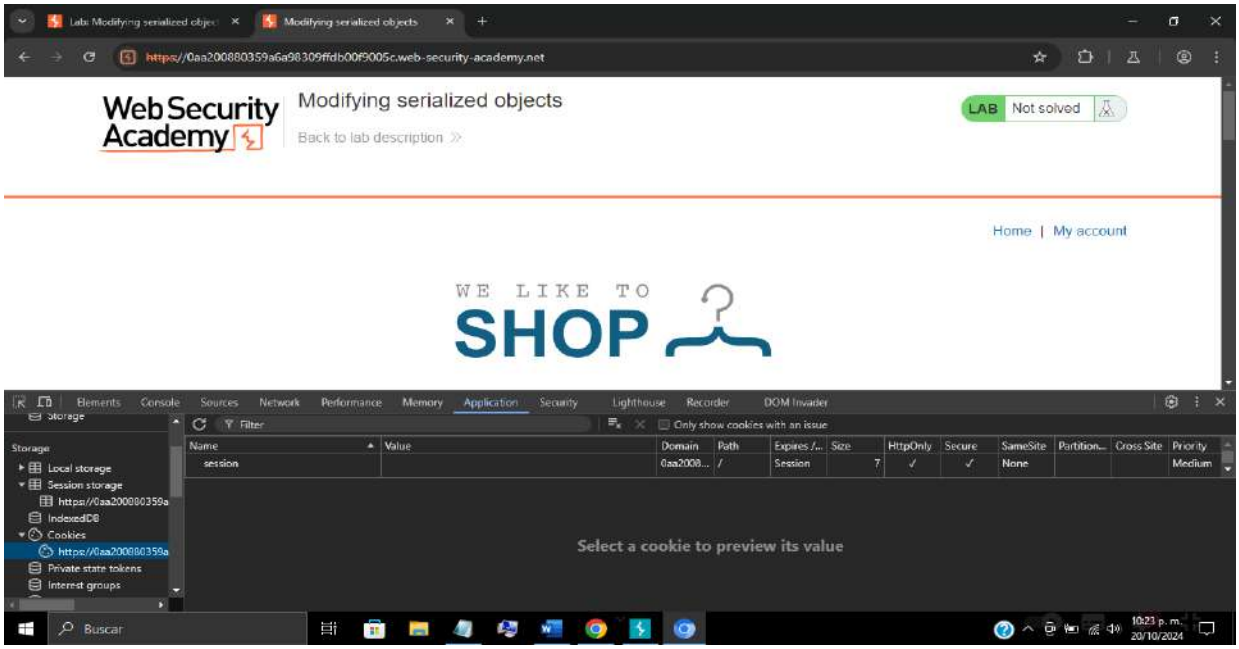
Se modifica el enlace decodificado cambiando el 0 por 1 para cambiar los privilegios del usuario (O:4:"User":2:{s:8:"username";s:6:"wiener";s:5:"admin";b:**0**;}), quedando: (O:4:"User":2:{s:8:"username";s:6:"wiener";s:5:"admin";b:**1**;}), igual a Base64



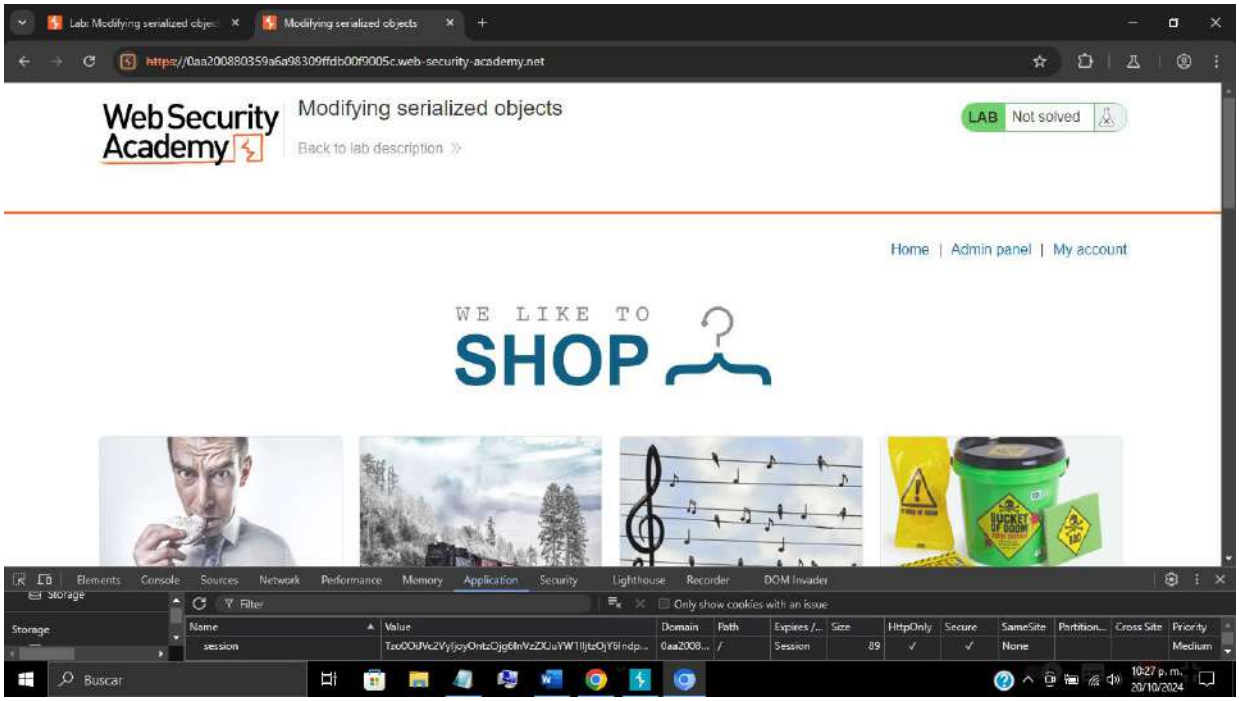
Posteriormente se codifica a Base64 para obtener el enlace modificado



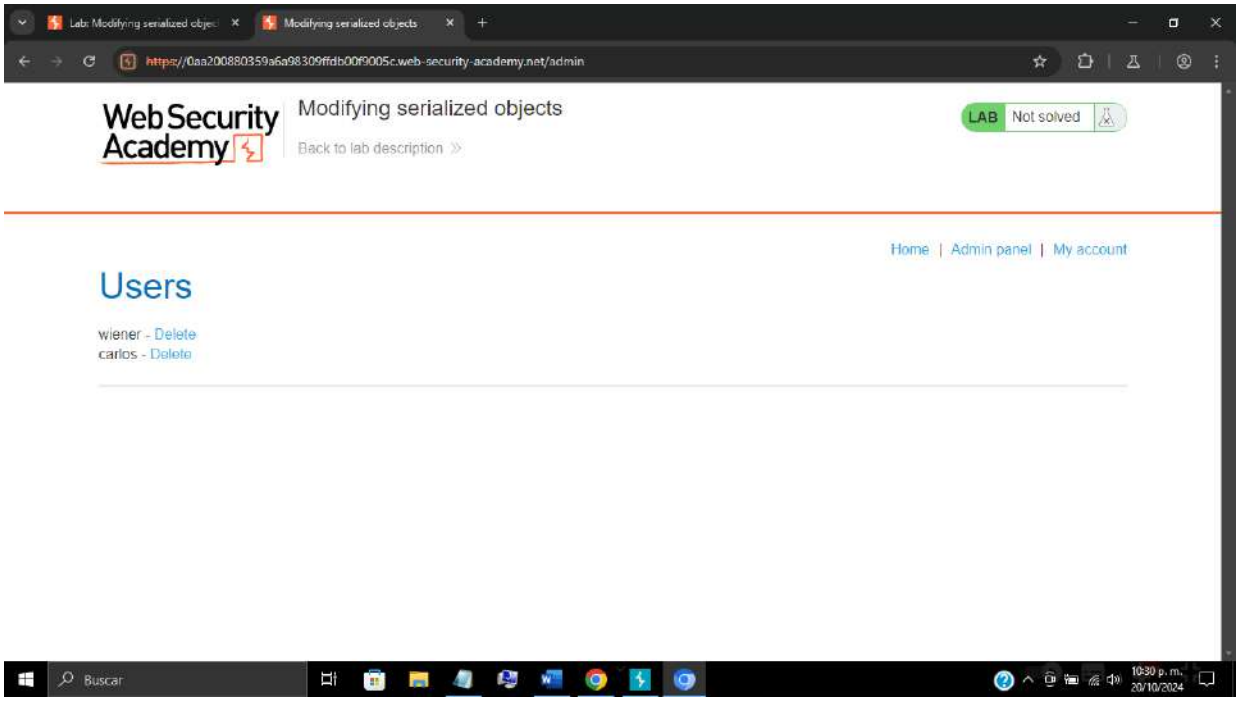
Y se modifica la terminación del enlace agregando los caracteres que se habían eliminado anteriormente %3d y se copia el enlace



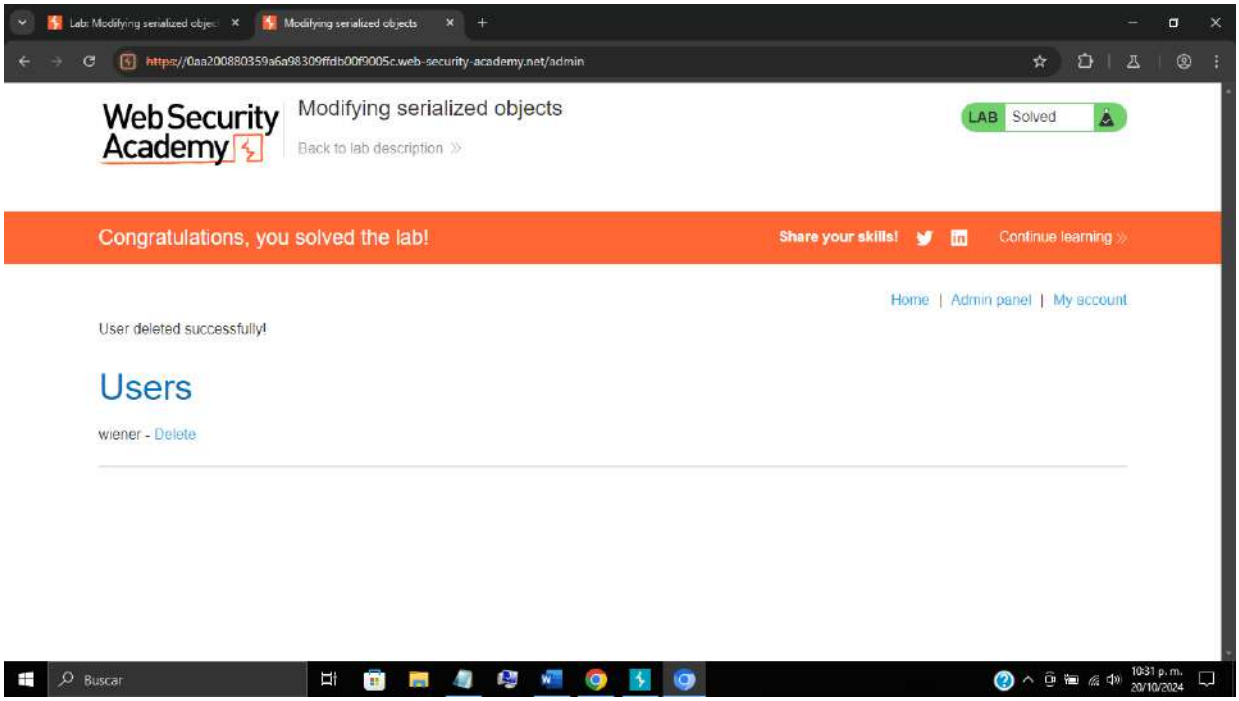
En el navegador se abren las herramientas para desarrolladores y se abren las propiedades de las cookies para pegar el enlace modificado en el campo value



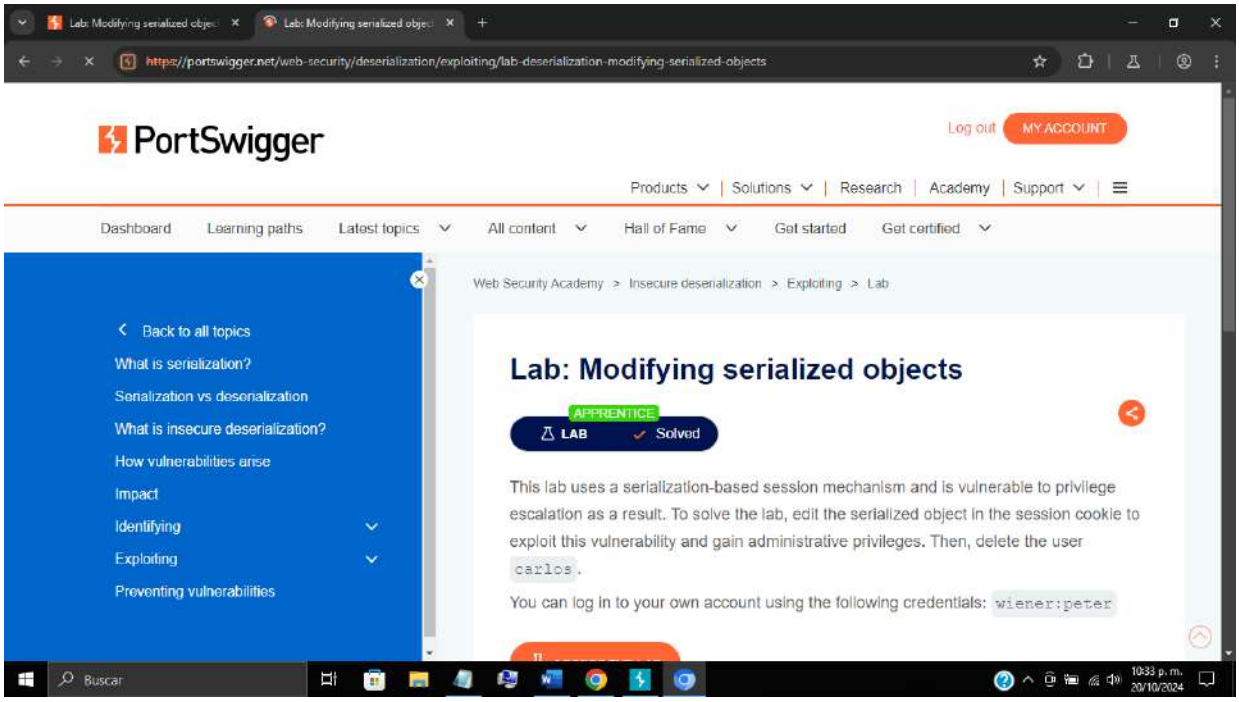
Se refresca la pagina para poder visualizar el panel de administrador



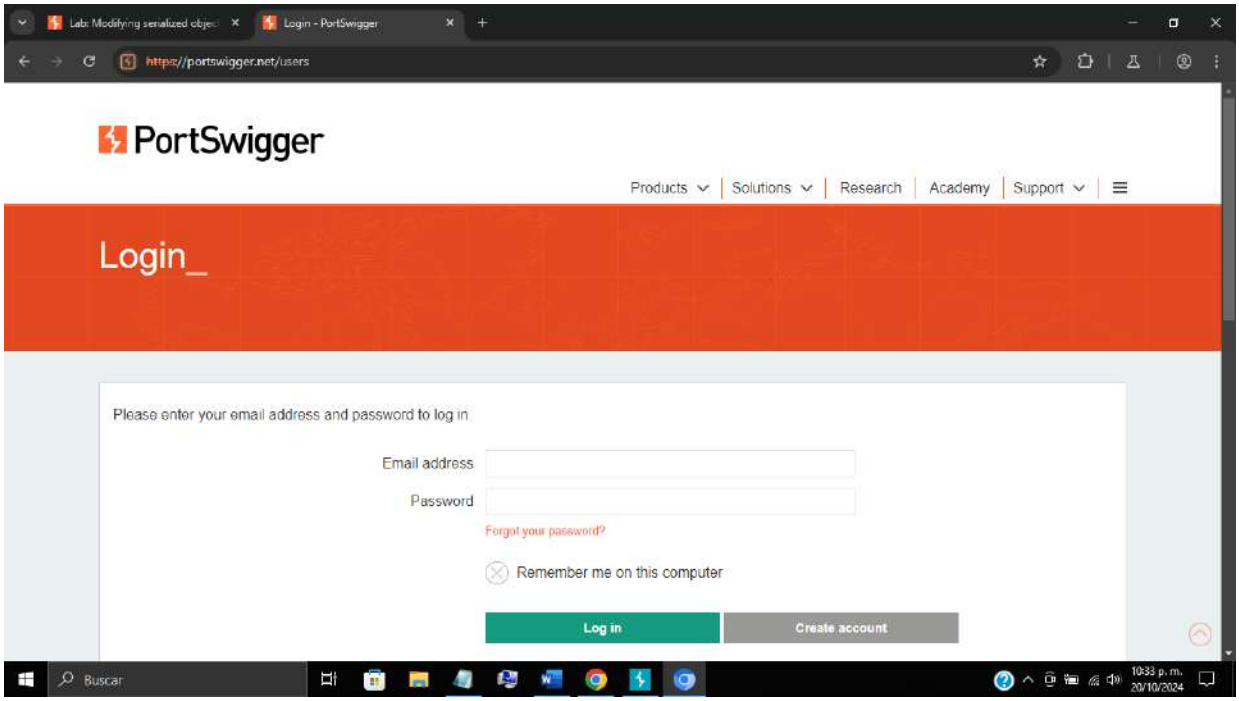
Se da click en el panel de administrador y nos muestra la lista de usuarios



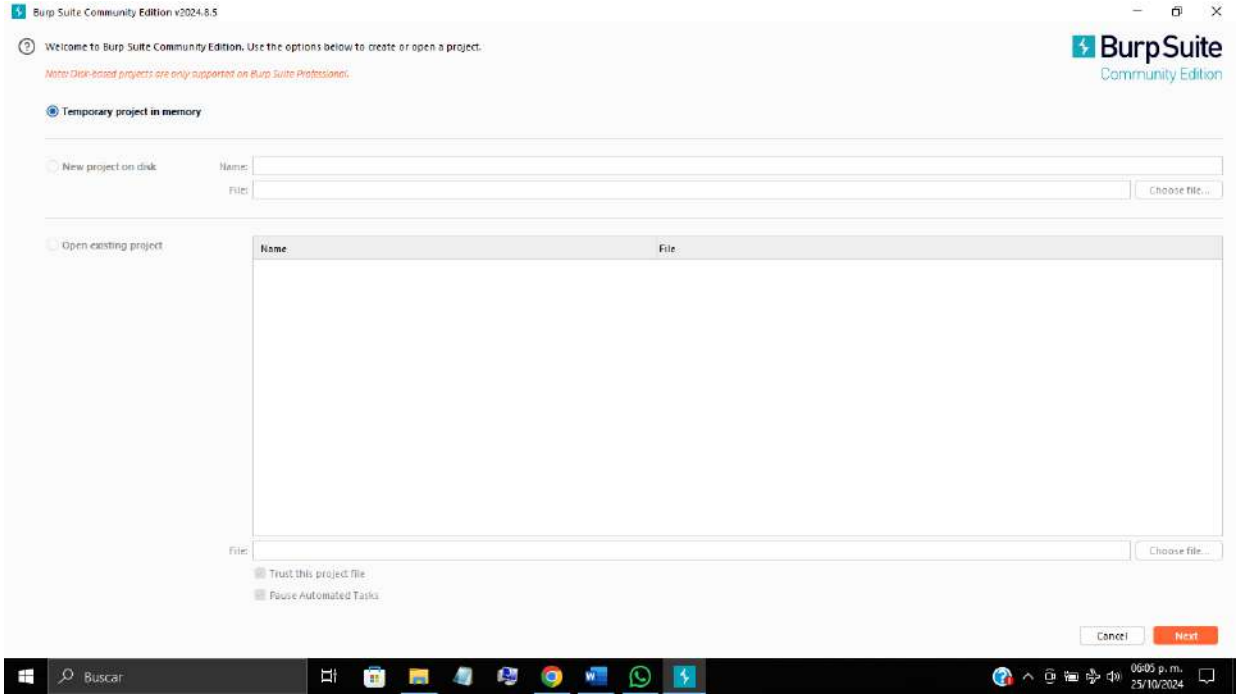
Se elimina el usuario Carlos para concretar la prueba



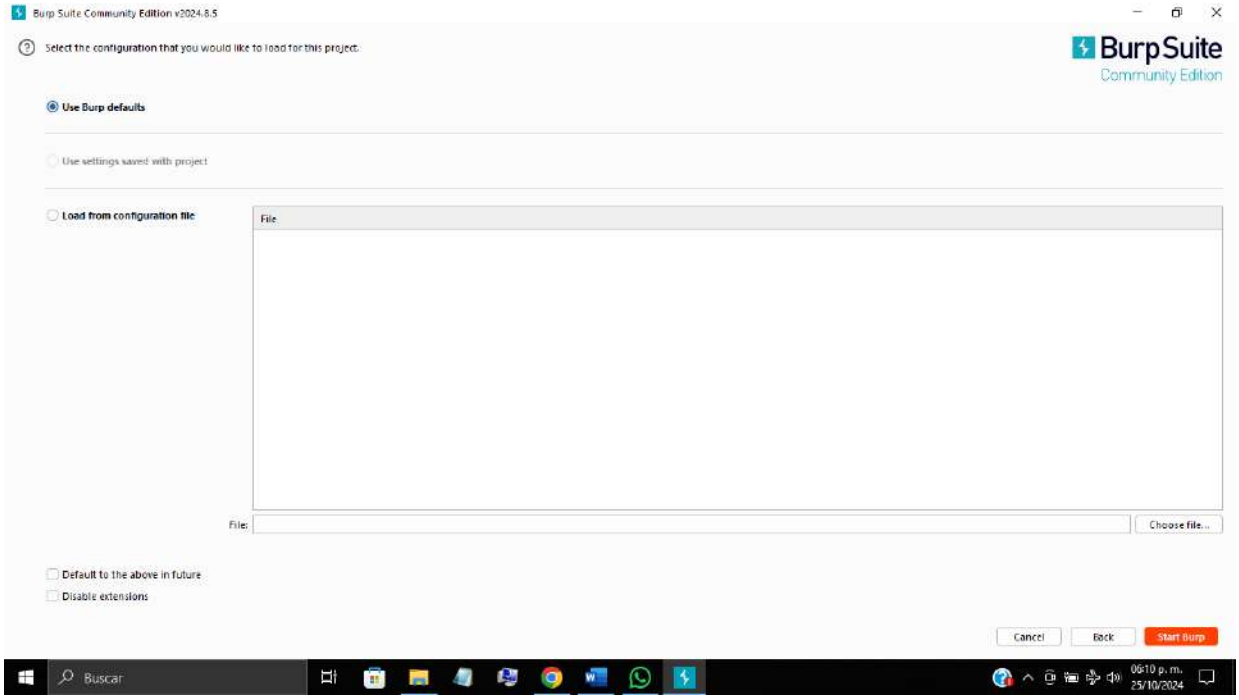
Y se cierra sesión

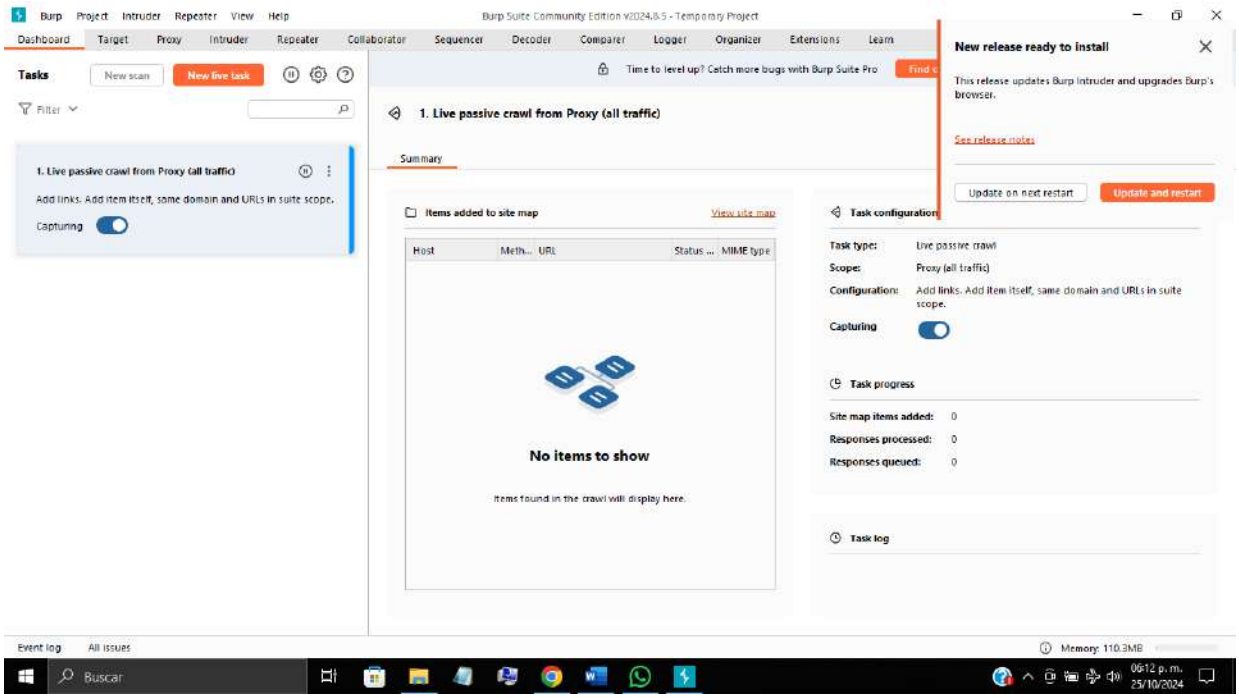


Etapa 3 – Ataque al sitio

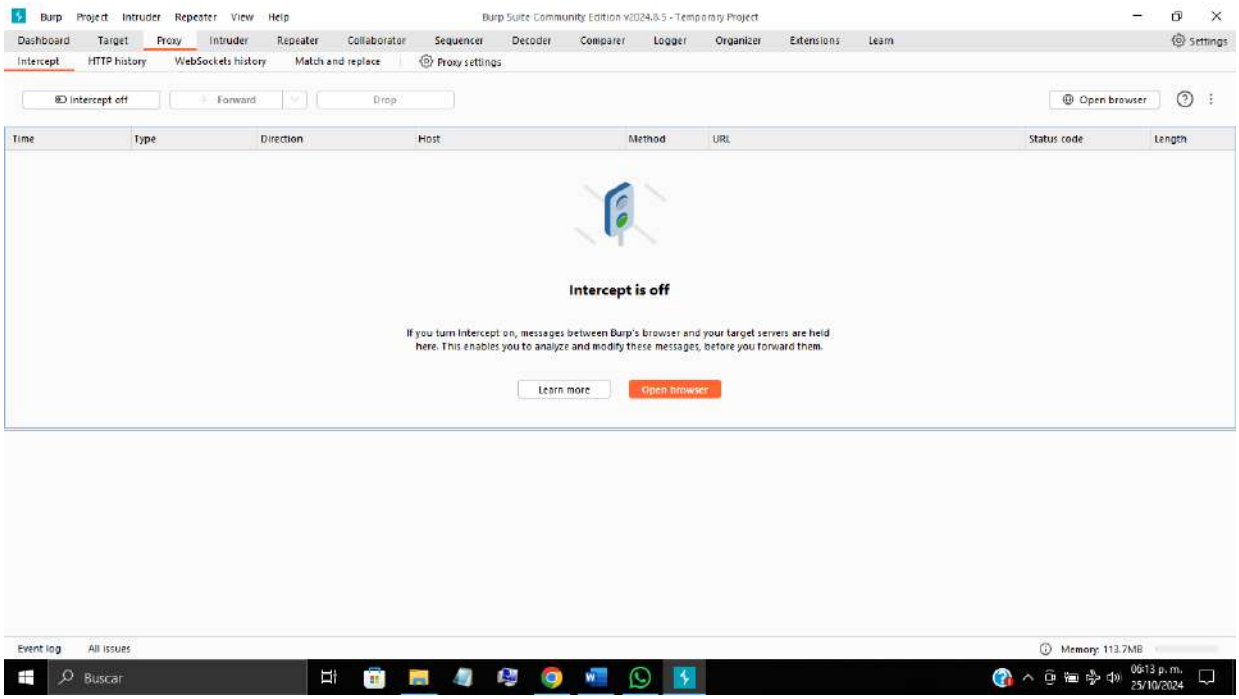


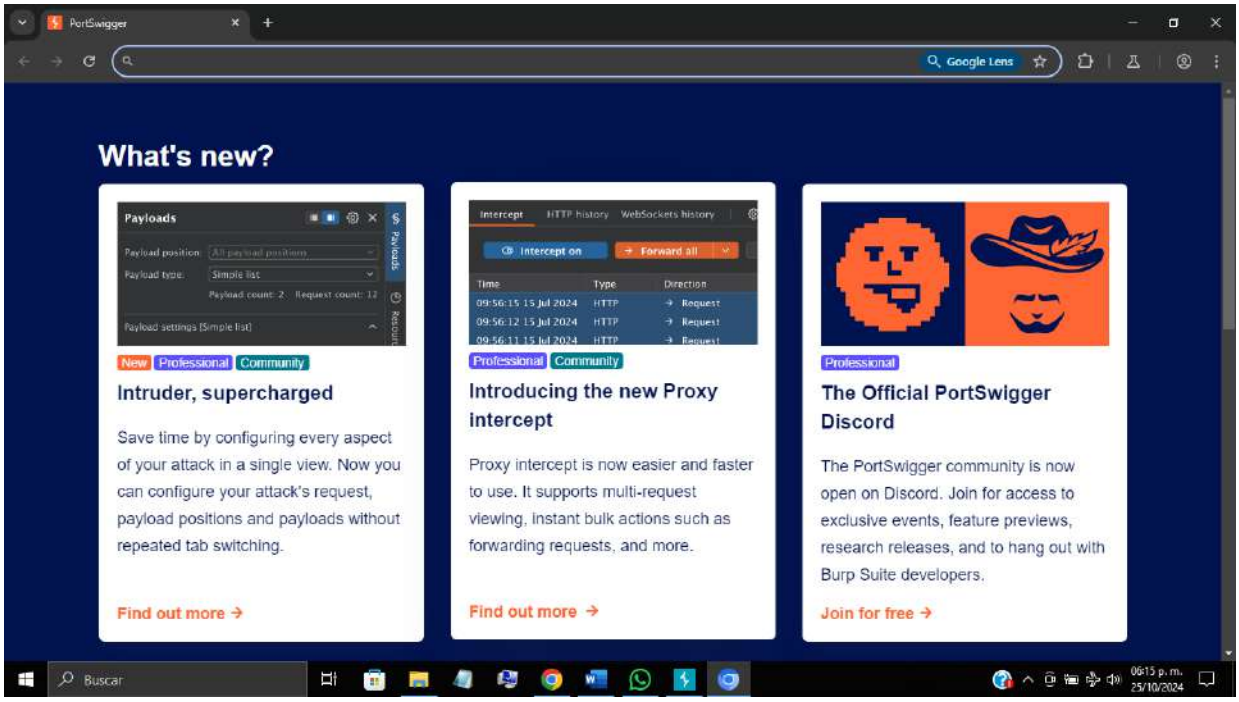
Se ingresa a Burp Suite



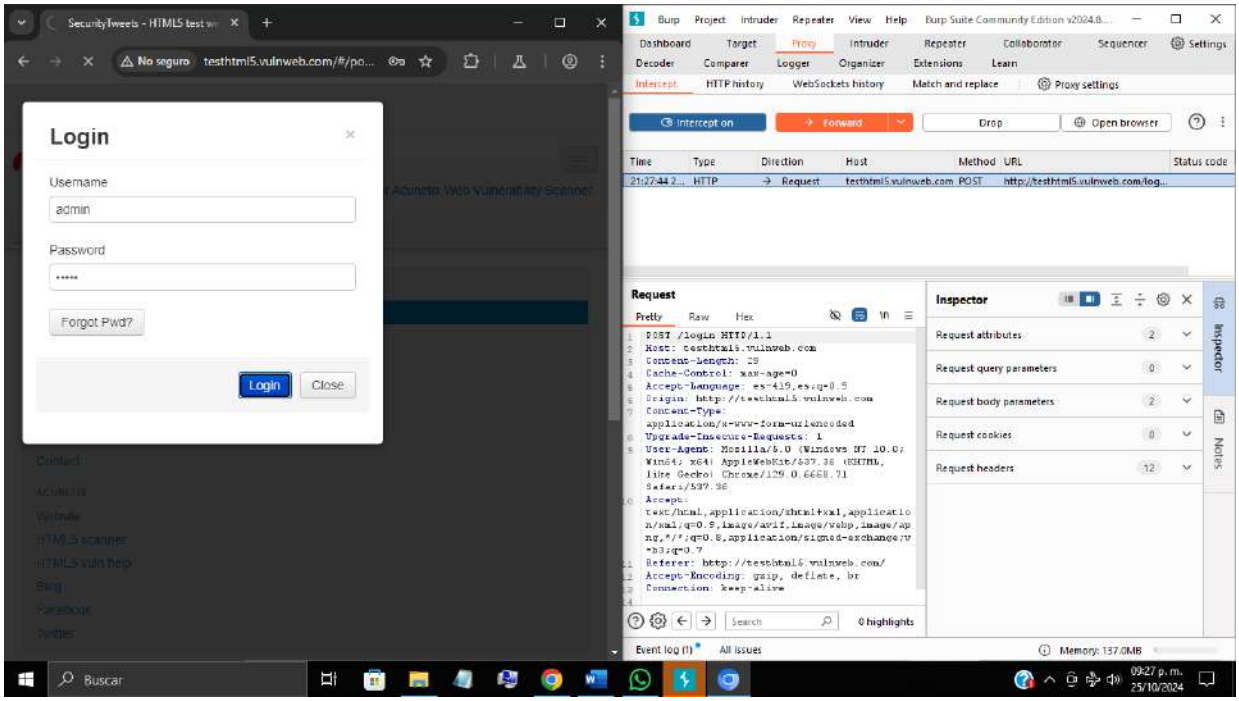


Se crea el proyecto nuevo y en el apartado proxy se abre el navegador

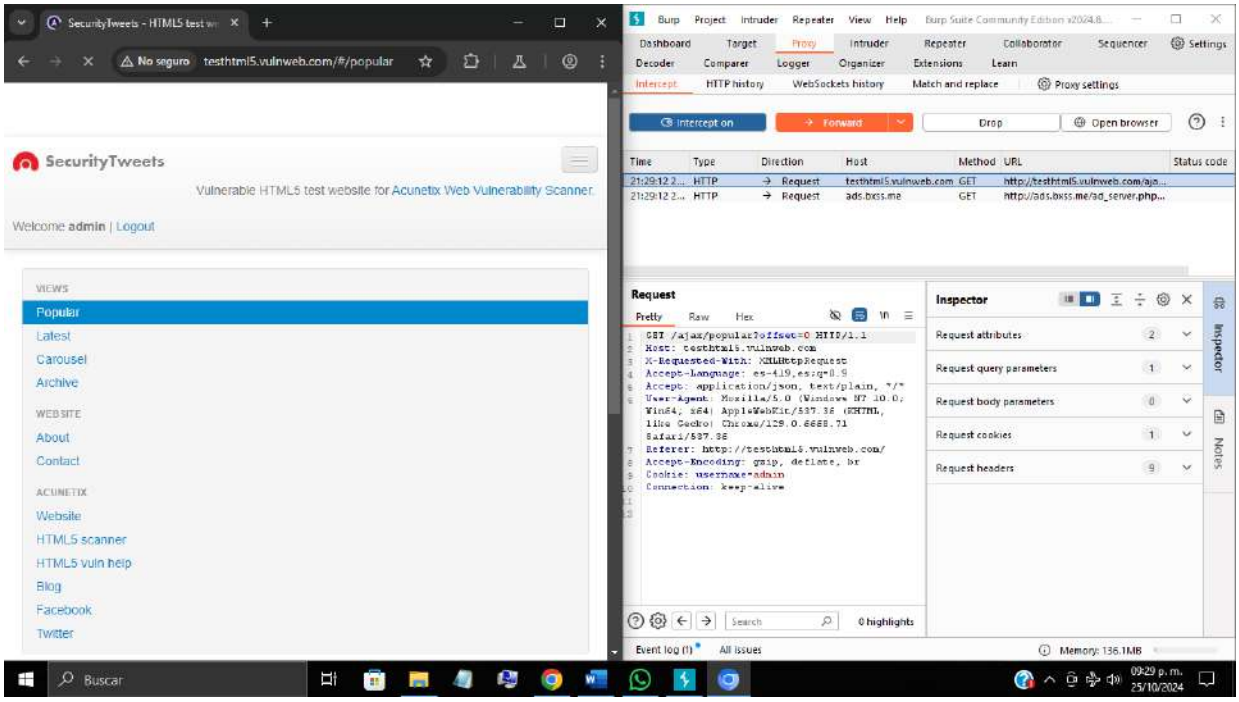




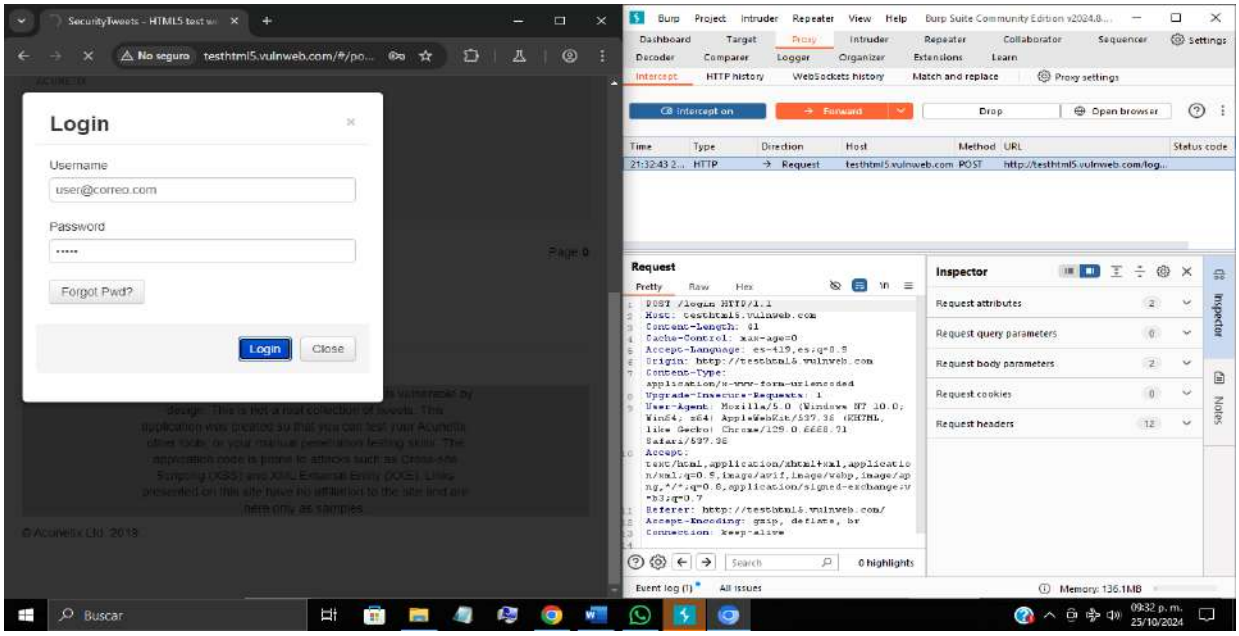
Se ingresa al enlace para probar la funcionalidad de Burp Suite al encender el interceptor



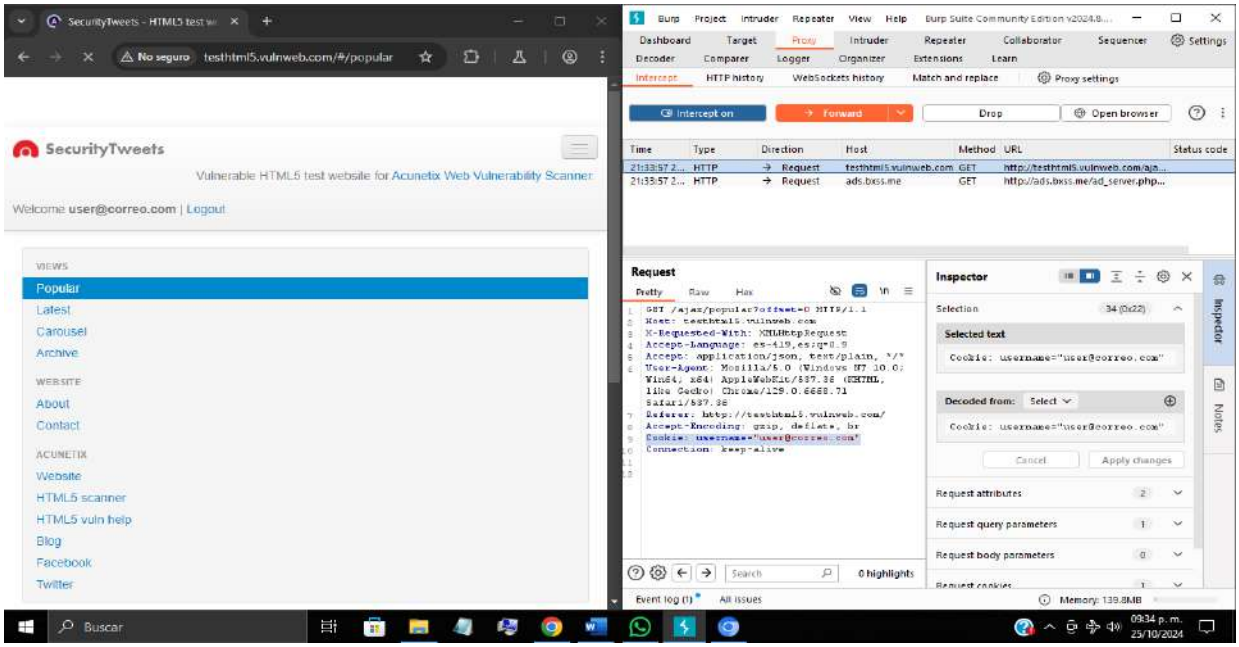
Se ingresan los datos sin que cargue la pagina hasta que se da click en el boton forward



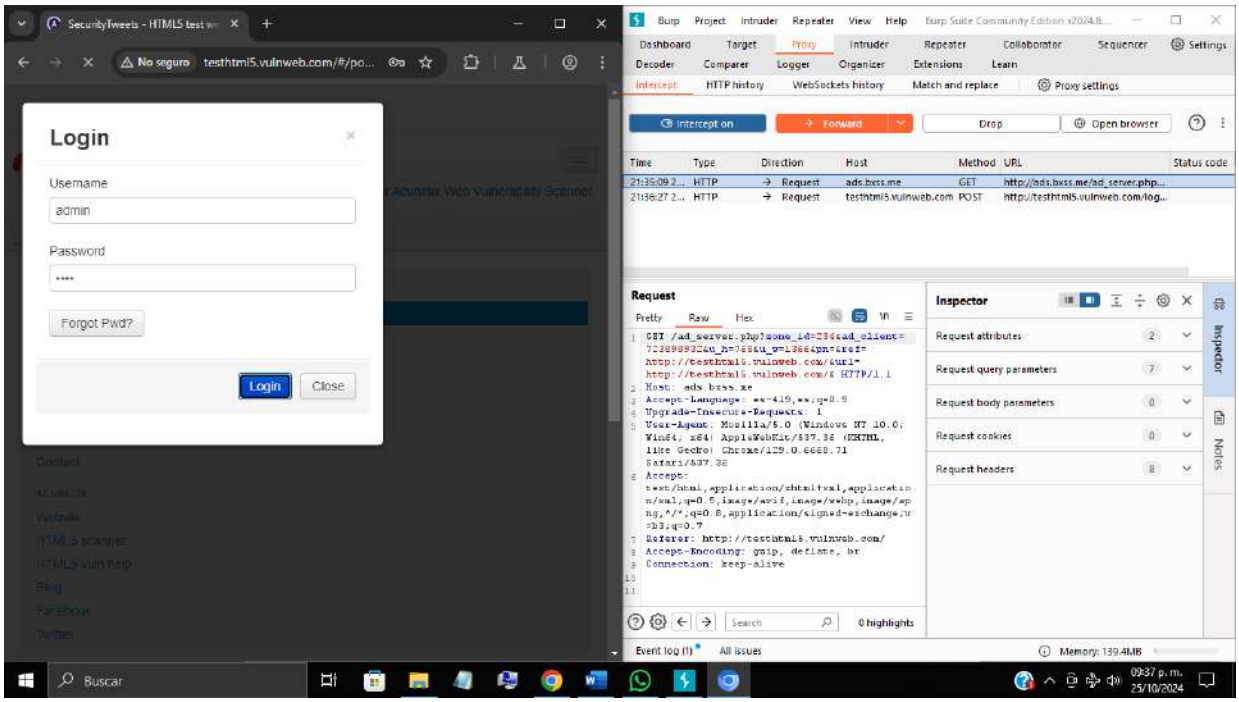
Ingresando a la pagina al mismo tiempo que se intercepta el trafico en Burp Site, se cierra sesion



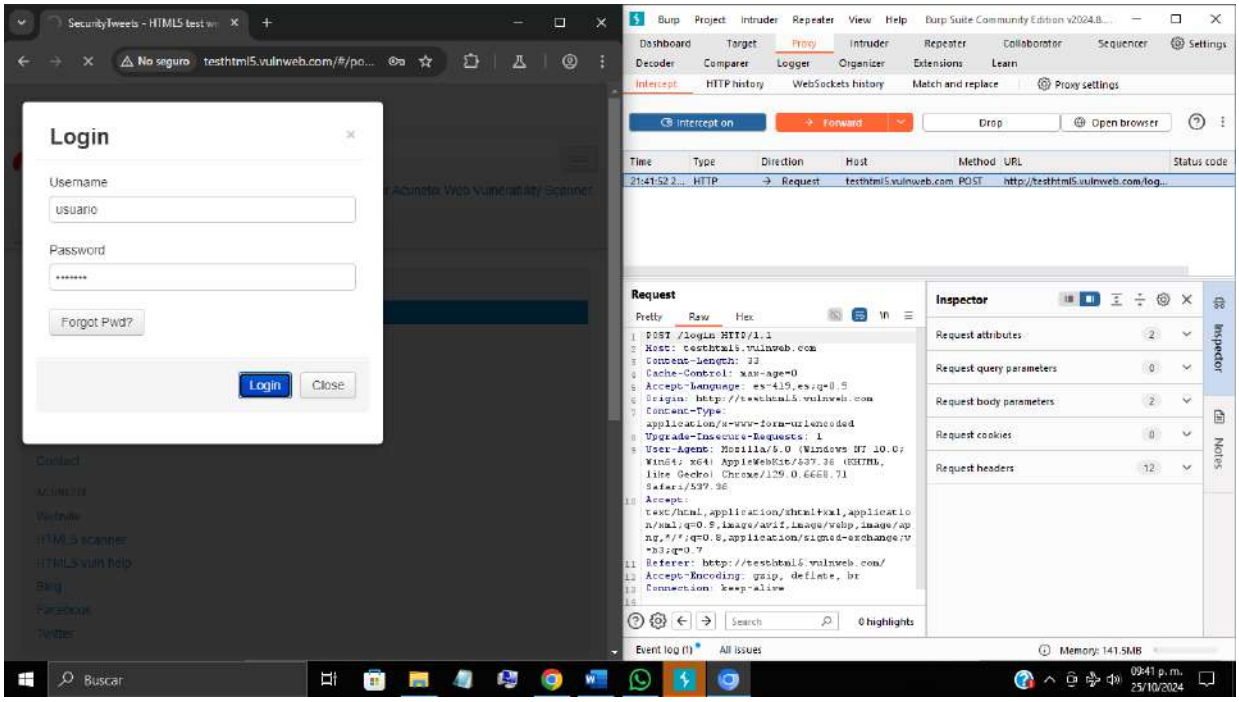
Se ingresa nuevamente cambiando los datos del usuario



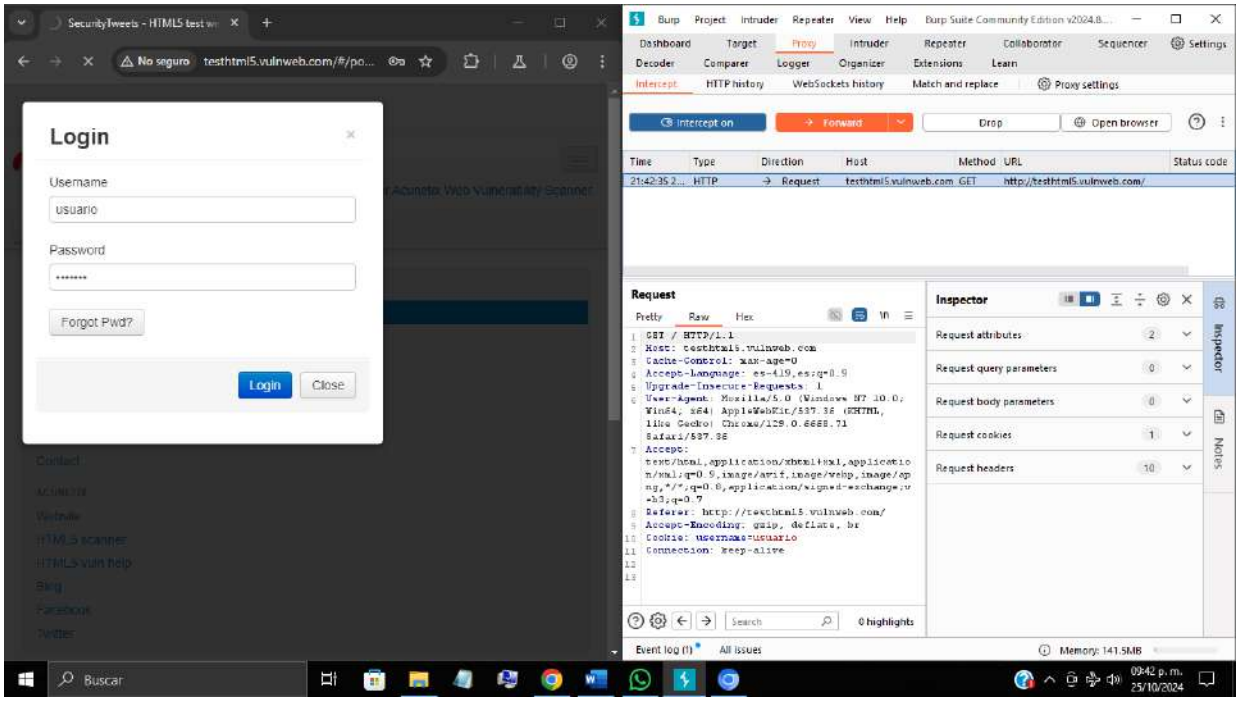
Permitiendo el ingreso a la pagina

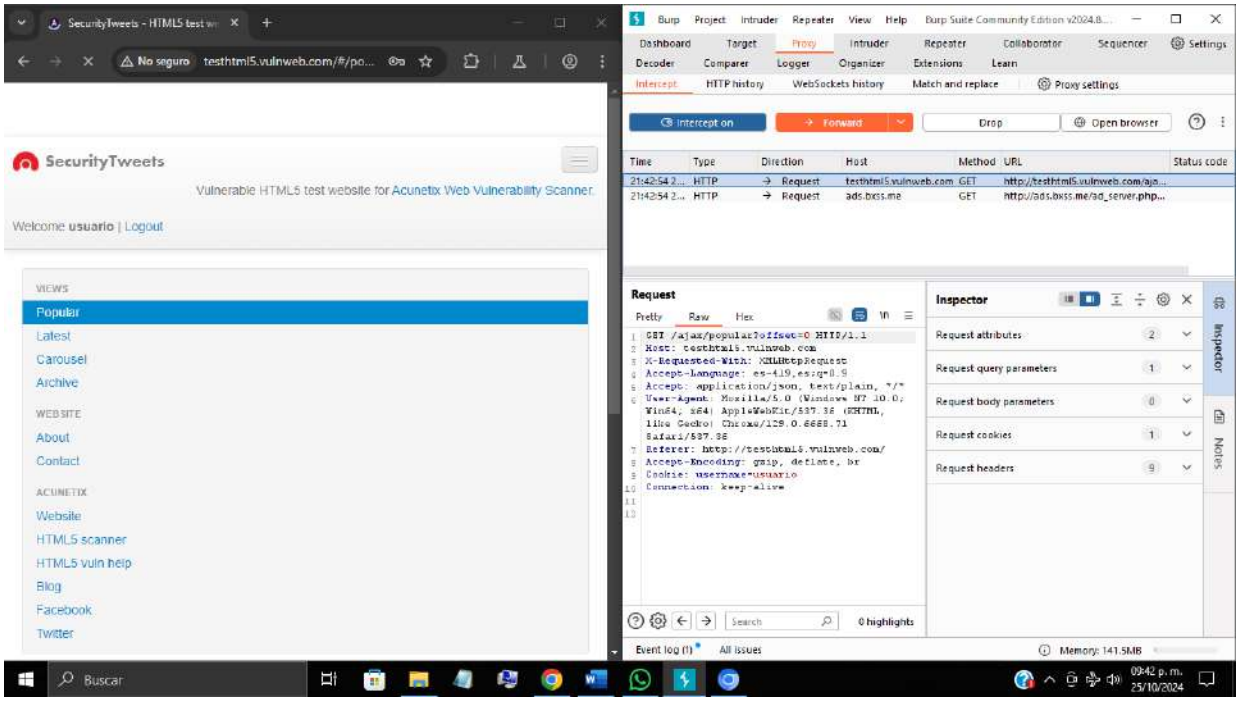


Se ingresa el usuario admin con contraseña diferente, marcando error, y despues de varios forward permite el acceso a la pagina

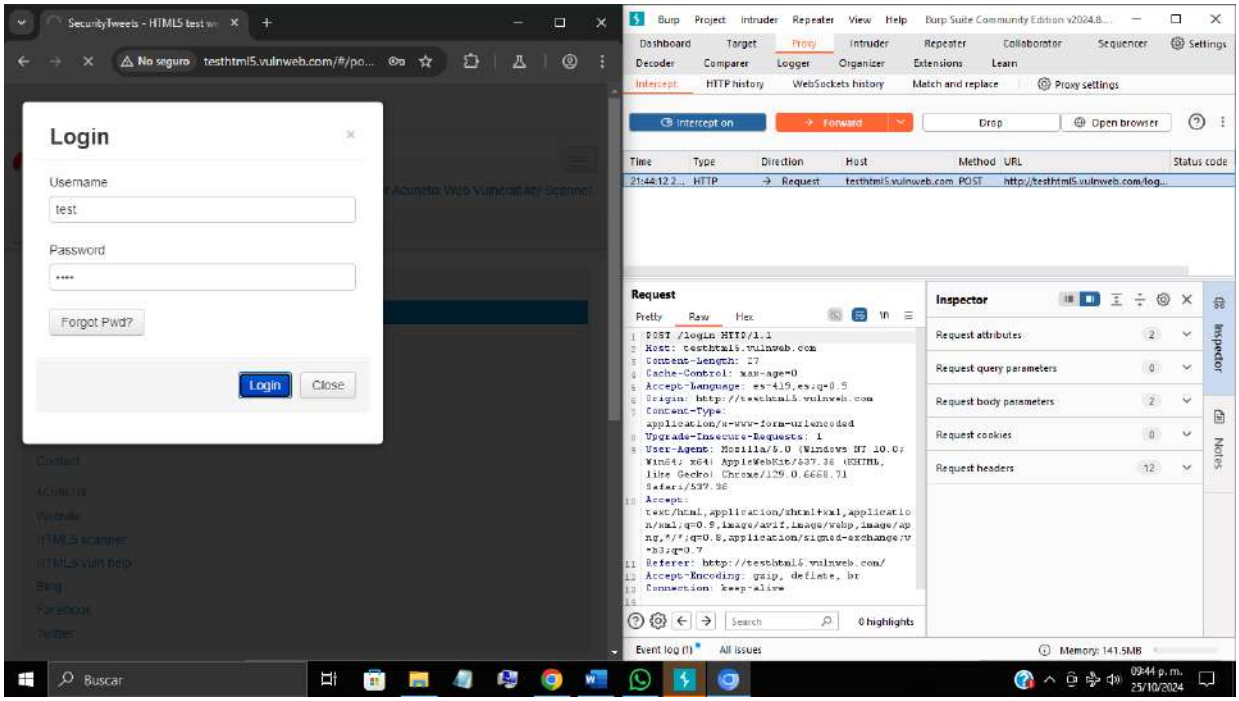


Se intenta el inicio de sesion con datos de usuario y contraseña diferentes

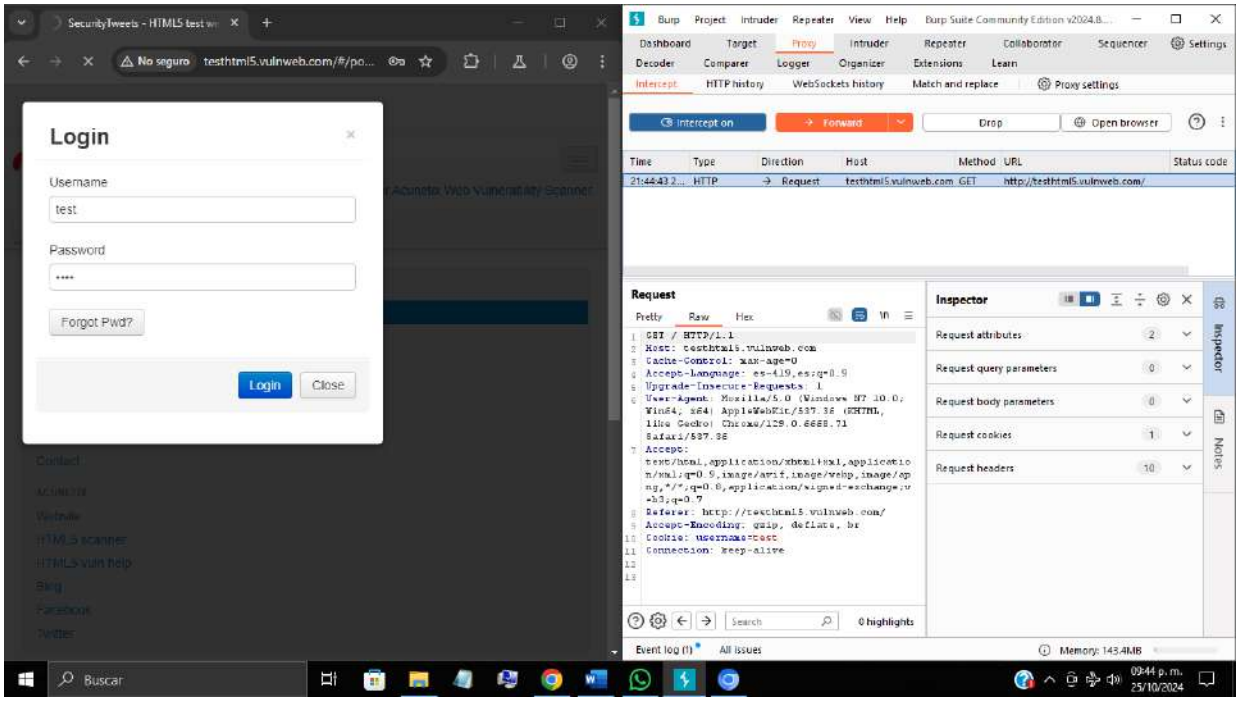




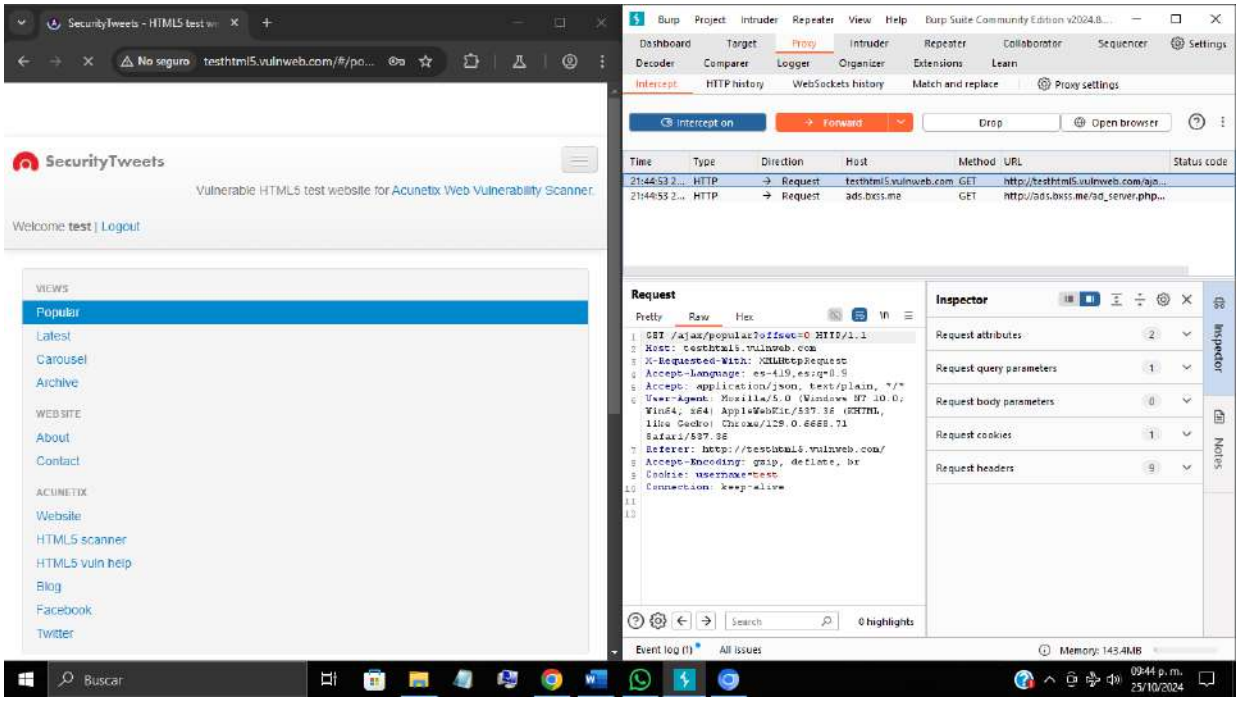
Permitiendo el ingreso a la pagina despues de varios forward

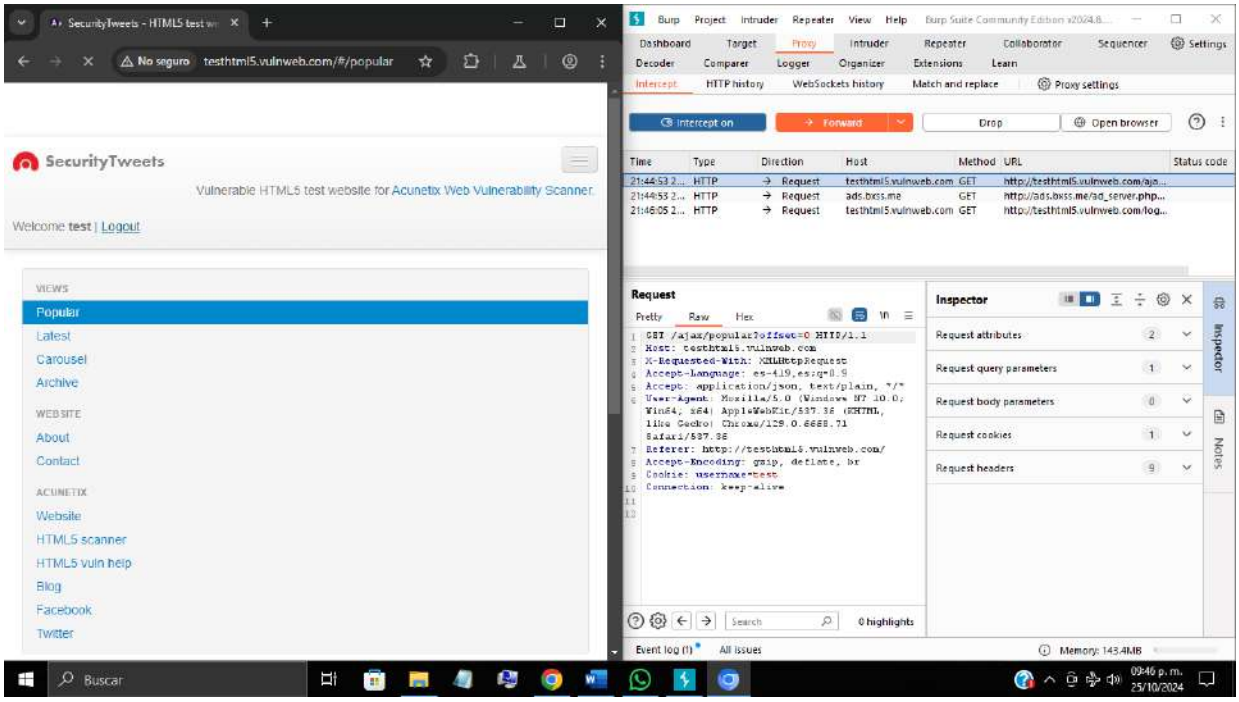


Se ingresan los datos con otra cuenta registrada

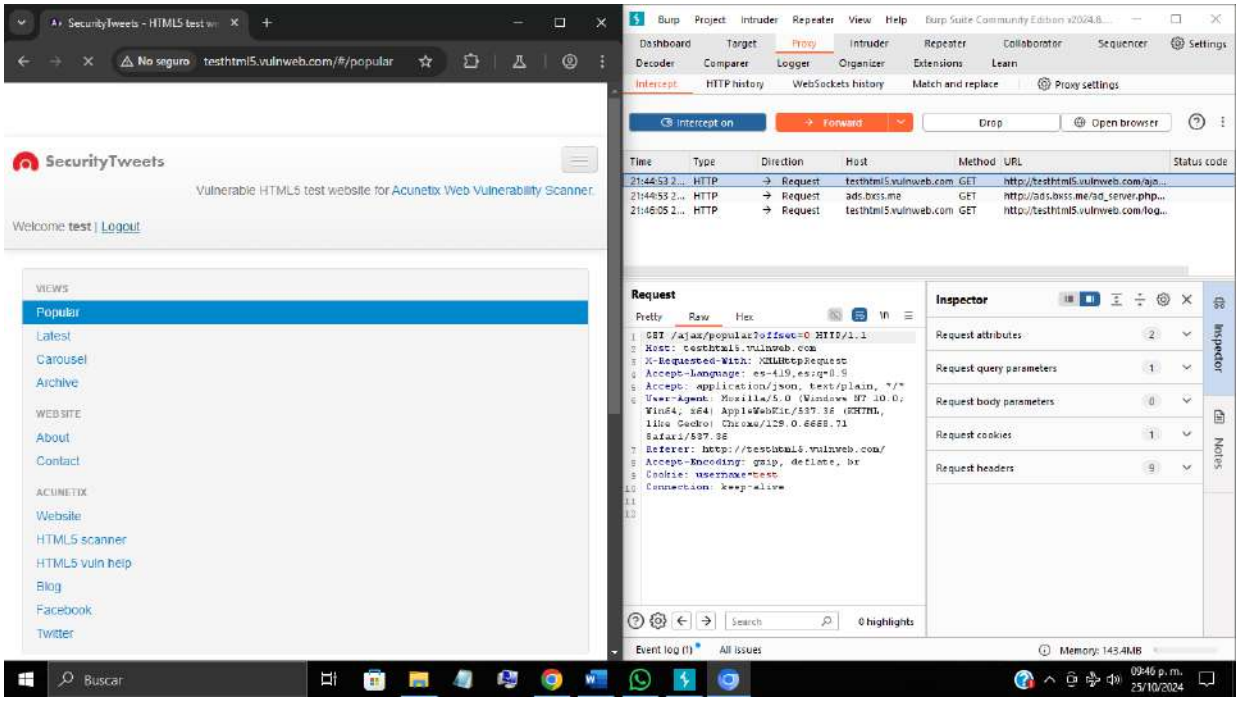


Logrando iniciar sesion sin ningun problema





Y se cierra sesion como en todos los pasos anteriores



Enlace de GitHub: https://github.com/Chifer888/Auditoria-informatica.git

Conclusión.

En conclusión, ambos conceptos son importantes para garantizar la seguridad y la usabilidad en sistemas que manejan sesiones de usuario y acceso a recursos sensibles, y se justifica como una práctica esencial para garantizar la seguridad, la experiencia del usuario, el cumplimiento de normativas, la eficiencia del sistema y la implementación de políticas de seguridad adecuadas, la deserialización insegura es una de las principales vulnerabilidades incluidas en las listas de amenazas de seguridad como OWASP Top 10, donde se supone que una aplicación web recibe datos de un cliente en formato JSON para deserializarlos y crear un objeto en el servidor y si la aplicación no valida adecuadamente ese JSON, el atacante puede modificar los datos para incluir script malicioso, que al deserializar los datos, el servidor ejecuta un script, causando daños significativos, y para mitigar el riesgo de Cros Site Scripting XSS, es fundamental validar y sanitizar todas las entradas del usuario, implementando una política estricta de Content Security Policy (CSP), codificando adecuadamente los datos mostrados en las páginas web.

¿Qué aprendo?

Que aún tengo mucho por aprender para poder gestionar el funcionamiento de las redes y poder detectar las intrusiones ajenas a los equipos, que es sumamente importante establecer el tipo de seguridad que deben tener nuestros diseños para garantizar la seguridad y evitar que los datos de los usuarios sean interferidos por terceros al momento de autenticarse e ingresar a los sitios seleccionados que requieran de un inicio de sesión, aun cuando en la actualidad ya se contempla desde el momento de la elaboración de los diseños, al no permitir publicar las páginas sin establecer el tipo de seguridad a menos de requerir alguna remuneración de suscripción para poder publicarlas, que es de suma importancia dar la prioridad adecuada y que se merece a la privacidad y seguridad al momento de diseñar páginas en la web para que no sean vulnerables a intrusiones no deseadas, que al ser la primera vez que se utiliza Burp Suite Community Edition se invierte demasiado tiempo para entender cómo funciona su proceso, y que las personas con los conocimientos necesarios y la habilidad en el uso de esta herramienta le tomaría no más de cinco minutos para vulnerar y modificar la información privilegiada.

Referencias

- *ChatGPT*. (n.d.). https://chatgpt.com/c/670afc6c-da4c-8003-83b6-7c97e3dd35fa

- *Wireshark · Go Deep*. (n.d.). Wireshark. https://www.wireshark.org/

- Luis Peralta Molina. (2021, August 2). *Como descargar e instalar Wireshark* [Video]. YouTube. https://www.youtube.com/watch?v=L07PeLPtvPo

- Abel Albuez. (2018, June 26). *Subir pagina con PHP + Base de datos (MYSQL) a un Hosting* [Video]. YouTube. https://www.youtube.com/watch?v=5tXxoRKdCWo

- 000webhost. (n.d.). *Mejor Hosting Gratis de Mexico 2020 | Hosting Gratuito*. Free Web Hosting. https://mex.000webhost.com/

- *ChatGPT*. (n.d.). https://chatgpt.com/c/670f3505-b99c-8003-99e2-bdbfca4c3fcc

- *Caja de herramientas de Google Admin*. (n.d.). https://toolbox.goo-gleapps.com/apps/main/

- Emanuele Picariello. (2022, June 29). *Insecure Deserialization vulnerabilities: Lab #1 by PortSwigger - Modifying Serialized Objects* [Video]. YouTube. https://www.youtube.com/watch?v=tm3u3Dw8I_4

- *Videoconferencias, conferencias web, seminario web, uso compartido de pantalla*. (n.d.). Zoom. https://academiaglobal-mx.zoom.us/rec/play/My8s_p5AihIzs8Wmz9-adK9FlJnvJA9MLerRNwL-GGpk2ZzbFtR6o8iAhBb4YrkD4c5hgR43I_35RWqG.Vtbw-qG4_jDuIBec?canPlayFromShare=true&from=share_recording_detail&continueMode=true&componentName=rec-play&originRequestUrl=https%3A%2F%2Facademiaglobal-mx.zoom.us%2Frec%2Fshare%2FKFTbRnIAl6ez2vvcpaLMiNdmhEwViz-BXYnxJlZ75njOevOCYS0xYB_dgHHFZ_L6y.dun0bhS2J89l5IYu

- *Acunetix Web Vulnerability Scanner - Test websites*. (n.d.). http://www.vulnweb.com/