TRABAJO COLABORATIVO CONTEXTUALIZADO

SEGURIDAD DE SOFTWARE

INTEGRANTES:

Diego Andres Ospino Molina

TUTOR: LUIS ANTONIO SARRUF DURANGO

UNIVERSIDAD DE CARTAGENA

CERETE/CORDOBA

2025

---

1. Introducción

El presente informe documenta las actividades realizadas en cumplimiento del trabajo final colaborativo del curso, con el objetivo general de fortalecer las competencias en seguridad de software. La metodología empleada se dividió en dos áreas de enfoque principales, tal como se detalla en este documento.

La primera fase consistió en un análisis de reconocimiento pasivo (Fase 1 de la guía), donde se identificaron las tecnologías web y se analizó la comunicación del protocolo HTTP de tres sitios web de distintos sectores (educativo, comercial y gubernamental). Para esto, se utilizaron herramientas de *fingerprinting* como Wappalyzer y las herramientas de desarrollador del navegador[5].

La segunda área de enfoque fue la realización de pruebas de penetración (Fase 3 de la guía) en un entorno de laboratorio controlado y seguro. Utilizando la máquina virtual vulnerable Metasploitable 2.0 como objetivo , se emplearon herramientas como Nmap, y OWASP ZAP para identificar y reportar vulnerabilidades críticas, alineadas con el marco de referencia OWASP Top Ten. Este informe omite deliberadamente la Fase 2 (Desarrollo de Aplicación Web) para centrarse en profundidad en las fases de reconocimiento y pentesting.

---

2. Desarrollo

A continuación, se presentan en detalle los hallazgos de las fases 1 y 3.

2.1. Fase 1: Análisis de Tecnologías Web y Protocolo HTTP

Para esta fase, se seleccionaron los siguientes tres sitios web representativos de los sectores solicitados. Se utilizó la extensión de navegador Wappalyzer para el *fingerprinting* de tecnologías y las herramientas de desarrollador del navegador para la inspección de las cabeceras y estados del protocolo HTTP.

Tabla 1: Resumen de Tecnologías y Análisis HTTP

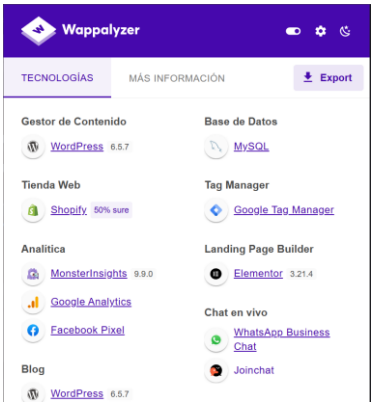
Sitio Web (Sector)	**Tecnologías Identificadas (Evidencia: Wappalyzer + Herramientas para desarrollador)**	**Análisis de Protocolo HTTP (Estados y Cabeceras)**
https://hotelseven.com.co/ (Comercial)	*** CMS: WordPress (Confirmado por**	*** Método: GET con estado 304 Not**

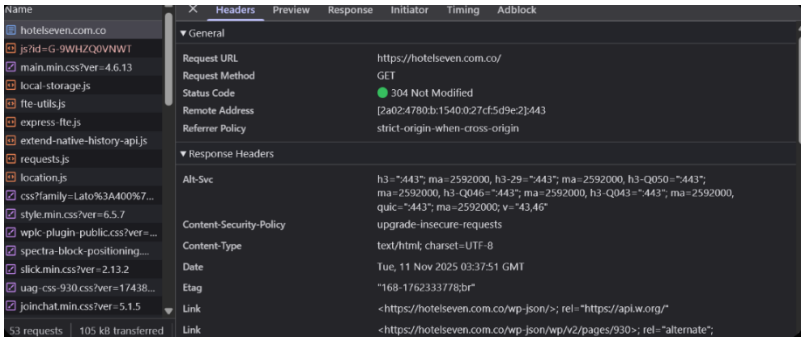
Sitio Web (Sector)	Tecnologías Identificadas (Evidencia: Wappalyzer + Herramientas para desarrollador)	Análisis de Protocolo HTTP (Estados y Cabeceras)
	cabecera Link en F12) * Servidor Web: LiteSpeed * Lenguaje: PHP/8.1.32 * Plataforma: Hostinger	Modified (Cargado desde caché). * Interacción: No se utiliza POST. Los formularios de contacto abren clientes de correo o WhatsApp.
https://www.monteria.gov.co/ (Gubernamental)	* CMS: WordPress 6.5.7 * Builder: Elementor 3.21.4 * Servidor Web: nginx	* Método: GET con estado 200 OK. * Cabeceras: Fuerte política de seguridad (HSTS, X-Frame-Options). * Interacción: El buscador usa GET con parámetros de consulta.

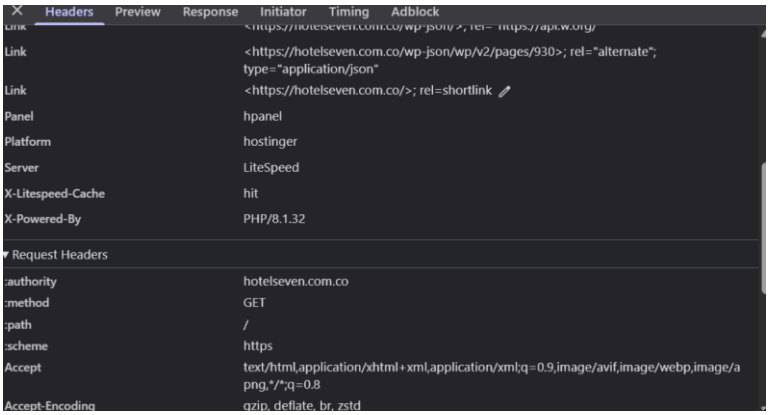
Sitio Web (Sector)	Tecnologías Identificadas (Evidencia: Wappalyzer + Herramientas para desarrollador)	Análisis de Protocolo HTTP (Estados y Cabeceras)
	* Lenguaje: Oculto (X-Powered-By: n/a)	
httpshttps://cdigital.cun.edu.co/ (Educativo)	* Plataforma (CMS): Moodle (Confirmado por cookies y cabecera X-Redirect-By) * Servidor Web: Apache/2.4.56 * PaaS: Amazon Web Services (AWS)	* Método: GET con estado 200 OK. * Interacción: El formulario de "Acceder" usa el método POST y genera una redirección 303 See Other.

EVIDENCIAS:

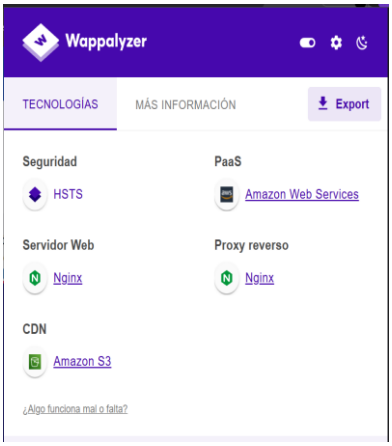
https://hotelseven.com.co/

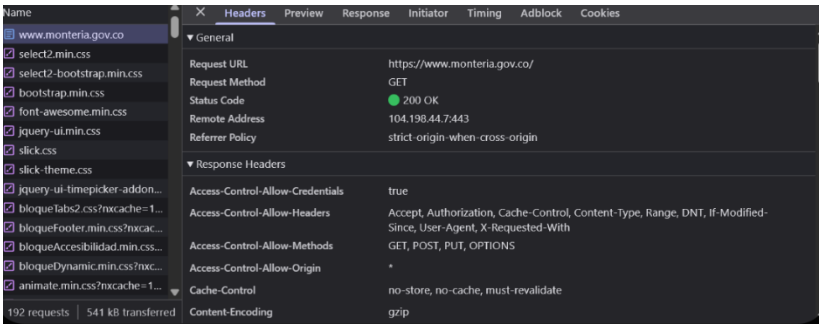


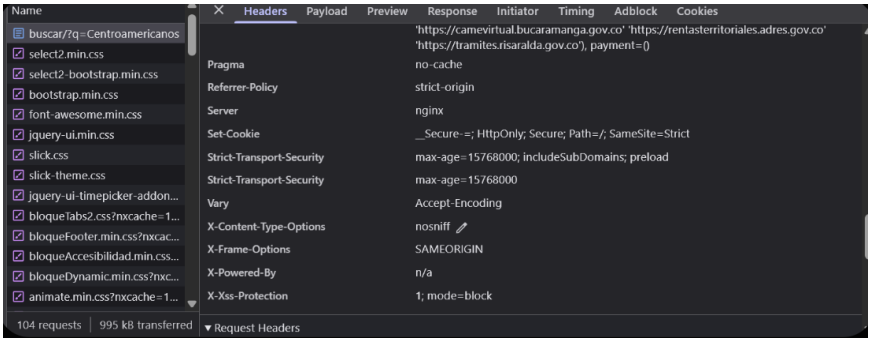




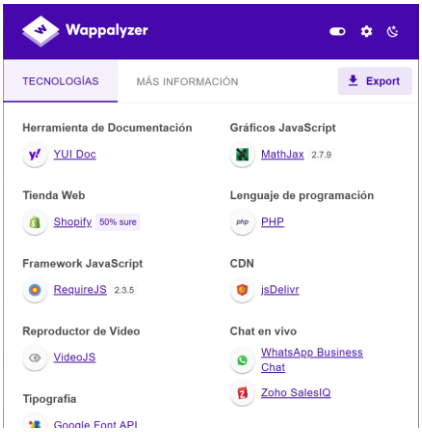
https://www.monteria.gov.co/

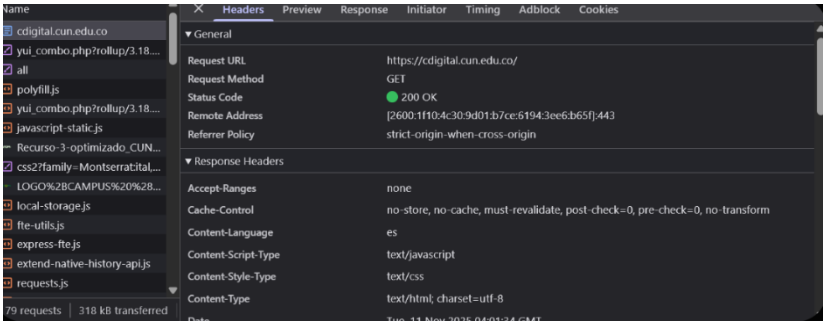


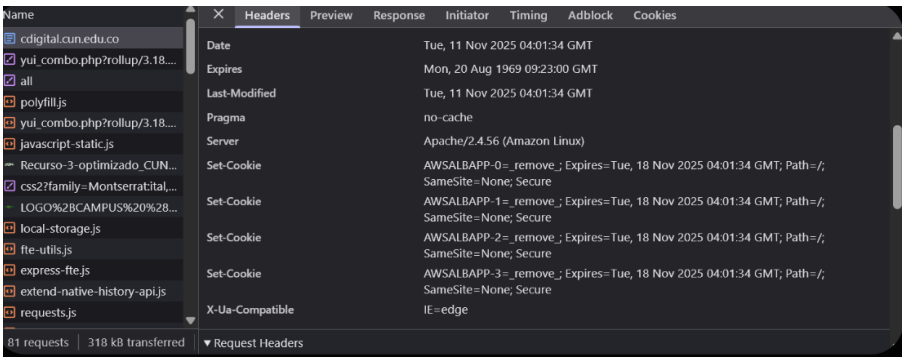


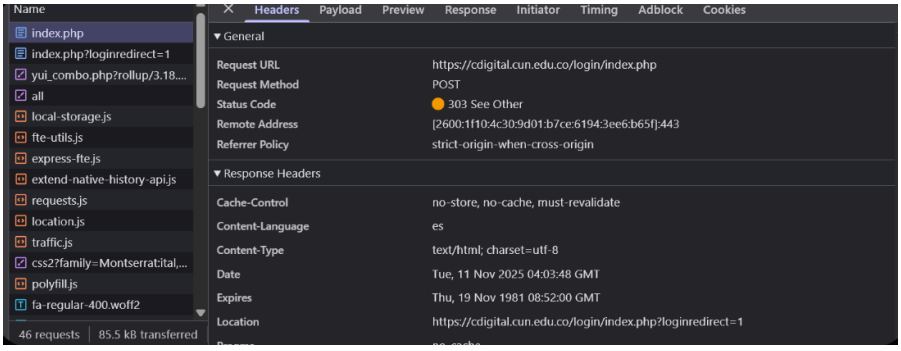


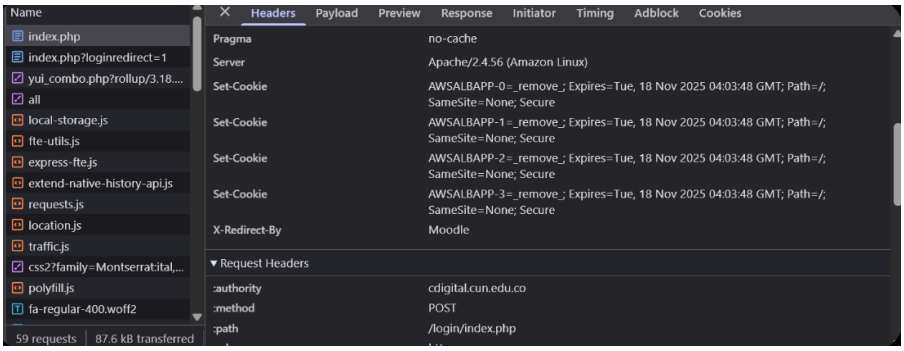
https://cdigital.cun.edu.co/











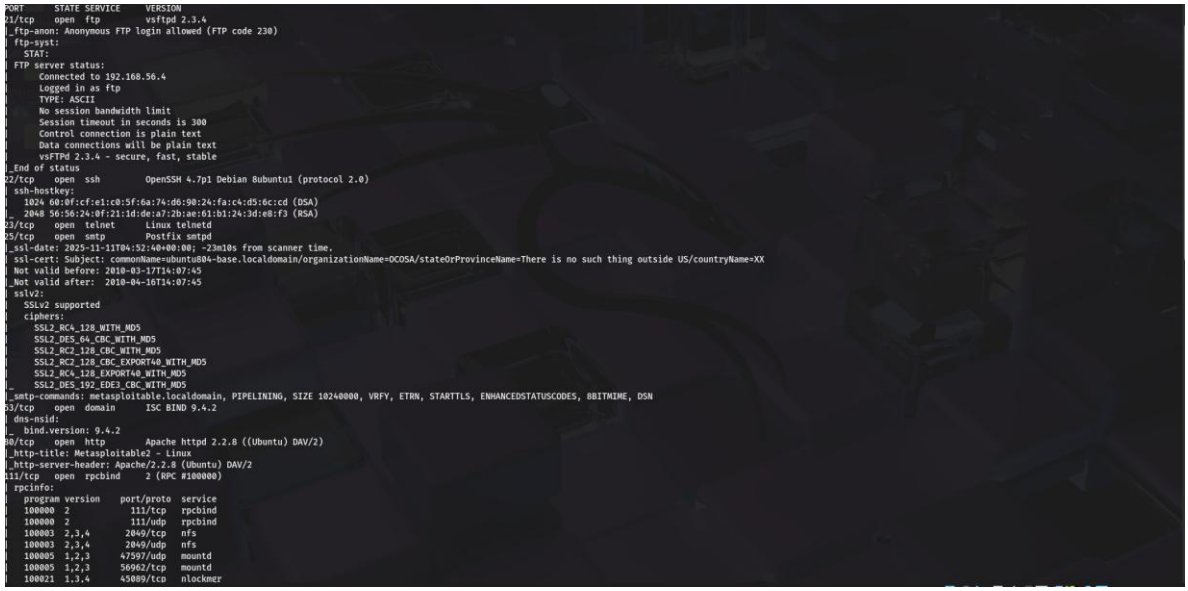
2.2. Fase 3: Pentesting y Vulnerabilidades Identificadas

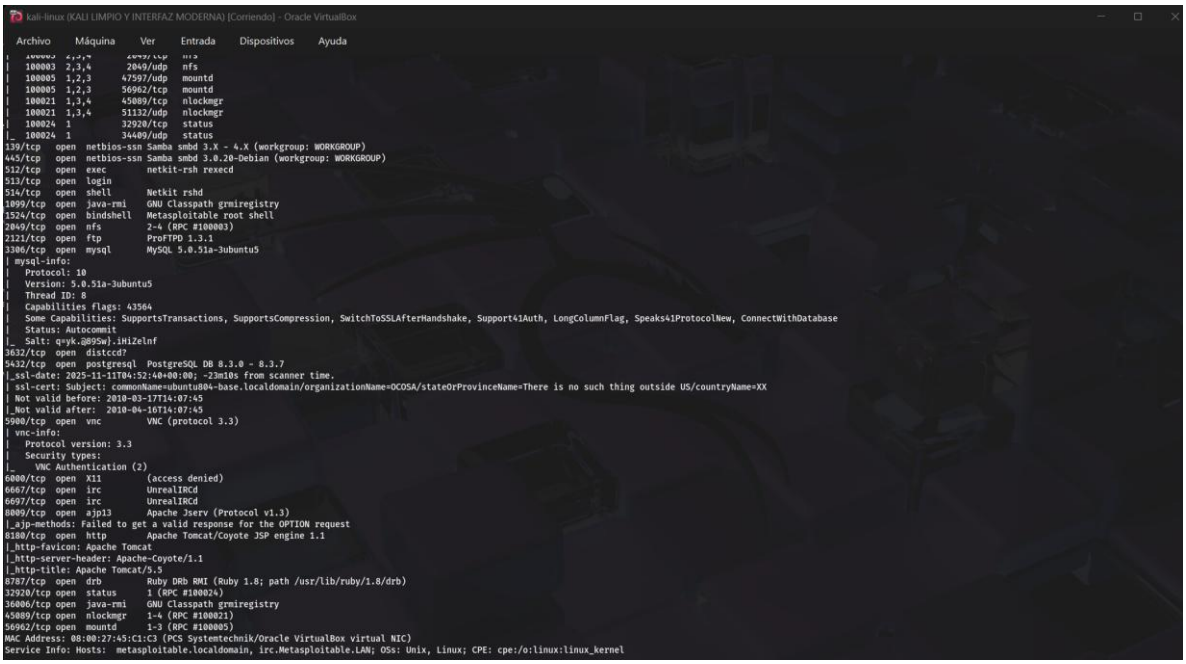
Esta fase se ejecutó en un entorno de laboratorio aislado utilizando VirtualBox. La máquina atacante fue una instancia de Kali Linux (IP 192.168.56.4) y la máquina objetivo fue la máquina virtual vulnerable Metasploitable 2.0 (IP 192.168.56.5). El objetivo fue identificar un mínimo de cuatro vulnerabilidades siguiendo el marco OWASP Top Ten, utilizando las herramientas Nmap, OWASP ZAP y Burp Suite.

2.2.1. Escaneo de Red e Infraestructura (Nmap)

Se inició la fase de pentesting con un escaneo de red exhaustivo utilizando la herramienta Nmap. Se utilizó el comando nmap -sV -sC -p- 192.168.56.5 para descubrir todos los puertos abiertos, identificar las versiones de los servicios y ejecutar scripts básicos de detección de vulnerabilidades.

"Resultados del escaneo de Nmap a Metasploitable 2.0, mostrando servicios y versiones."





El escaneo de Nmap reveló numerosos servicios obsoletos y mal configurados. A continuación, se detallan las vulnerabilidades más críticas identificadas con esta herramienta:

Vulnerabilidad 1: Componentes Vulnerables y Desactualizados (A06:2021)

- **Herramienta**: Nmap
- **Puerto y Servicio**: 21/tcp - vsftpd 2.3.4
- **Descripción**: Esta versión específica (2.3.4) de vsftpd es mundialmente famosa por contener una vulnerabilidad de puerta trasera (backdoor) (CVE-2011-2523). Un atacante puede obtener control total del sistema (shell) con solo enviar una secuencia de caracteres específica en el nombre de usuario.
- **Solución Propuesta:** Actualizar inmediatamente el servicio vsftpd a la última versión estable y parcheada. Si el servicio de FTP no es esencial para el negocio, deshabilitarlo por completo.

Vulnerabilidad 2: Fallos Criptográficos (A02:2021)

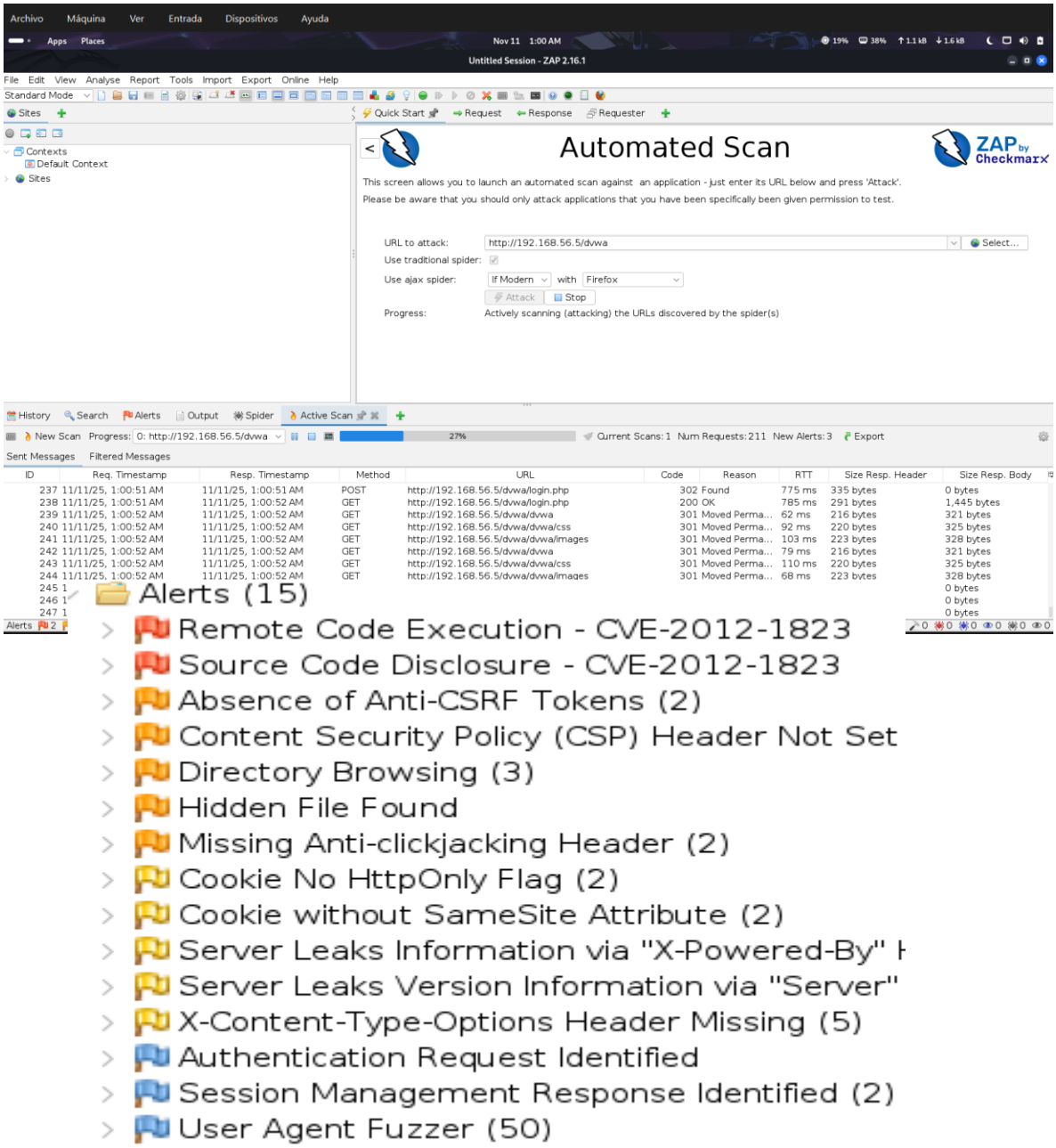
- **Herramienta:** Nmap
- **Puerto y Servicio:** 23/tcp - Telnet
- **Descripción:** Telnet es un protocolo de administración remota que no cifra el tráfico. Todas las comunicaciones, incluyendo credenciales de inicio de sesión (como msfadmin/msfadmin), se envían como texto plano. Esto permite a un atacante en la red capturar las credenciales fácilmente.

- **Solución Propuesta:** Deshabilitar permanentemente el servicio Telnet. Reemplazarlo por el servicio ssh (disponible en el puerto 22/tcp) para toda la administración remota, ya que SSH cifra toda la comunicación.

2.2.2. Escaneo de Aplicaciones Web (OWASP ZAP)

Posteriormente, se utilizó la herramienta OWASP ZAP para realizar un escaneo de seguridad de aplicaciones dinámicas (DAST) contra los servicios web alojados en el servidor.

"Resultados del Escaneo Activo de OWASP ZAP, mostrando una alerta de Ejecución Remota de Código."



El escaneo reveló una vulnerabilidad crítica de inyección de comandos:

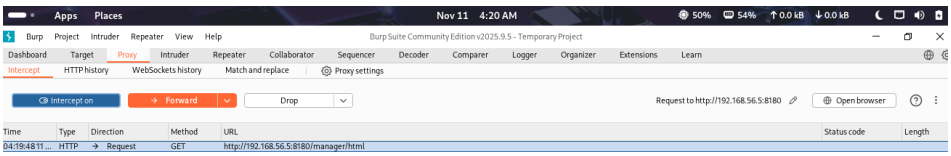
Vulnerabilidad 3: Inyección (A03:2021) - Ejecución Remota de Código

- **Herramienta:** OWASP ZAP
- **Descripción**: ZAP identificó una vulnerabilidad de Ejecución Remota de Código (CVE-2012-1823) en el script php.cgi del servidor. Esta vulnerabilidad permite a un atacante enviar una solicitud HTTP maliciosa que engaña al intérprete de PHP para que ejecute comandos del sistema operativo en el servidor, lo que resulta en un control total.
- **Solución Propuesta:** Actualizar la versión de PHP en el servidor a una versión parcheada que no sea vulnerable a CVE-2012-1823.

2.2.3. Interceptación y Análisis Manual (Burp Suite)

Finalmente, se utilizó Burp Suite para interceptar y analizar manualmente las solicitudes a los servicios web que requieren autenticación. Esta técnica se utilizó para confirmar las vulnerabilidades de autenticación encontradas por Nmap.

"Burp Suite interceptando un intento de login con credenciales por defecto ('tomcat:tomcat') en el puerto 8180."



Vulnerabilidad 4: Fallos de Identificación y Autenticación (A07:2021)

- **Herramienta: Burp Suite (y Nmap)**

- **Puerto y Servicio**: 8180/tcp - Apache Tomcat/Coyote JSP engine 1.1
- **Descripción:** Usando Burp Suite para interceptar un intento de inicio de sesión, se confirmó que la consola de administración de Tomcat (/manager/html) es accesible con las credenciales de administrador por defecto (tomcat / tomcat). Un atacante puede autenticarse y subir una aplicación web maliciosa (archivo .WAR) para ejecutar código.
- **Solución Propuesta:** Cambiar inmediatamente las credenciales por defecto del administrador de Tomcat por contraseñas robustas y únicas, e implementar una política de bloqueo de cuentas.

---

3. Conclusiones

La realización de este proyecto ha sido fundamental para consolidar la teoría de la seguridad de software en un contexto práctico, conectando directamente los objetivos del curso con la aplicación real de herramientas de ciberseguridad.

La **Fase 1**, de reconocimiento pasivo sobre sitios web productivos, demostró la gran cantidad de información que las aplicaciones exponen públicamente. Se constató que, si bien algunas configuraciones (como en el sitio de la alcaldía) implementan buenas prácticas de seguridad para ocultar detalles del backend (X-Powered-By: n/a), la mayoría de los sitios revelan abiertamente su pila tecnológica (CMS, servidor, lenguaje). Este análisis subraya una verdad fundamental: el primer paso de un atacante se basa en información pública, y la ofuscación de esta información es la primera línea de defensa.

La **Fase 3**, de pentesting en un entorno controlado, fue una demostración práctica del riesgo crítico que suponen los sistemas desactualizados y mal configurados. El escaneo con **Nmap** reveló una superficie de ataque inmensa, donde vulnerabilidades como el uso de protocolos inseguros (Telnet, **A02**) y software con puertas traseras conocidas (vsftpd, **A06**) representan un riesgo inminente.

El uso de **OWASP ZAP** y **Burp Suite** complementó este análisis, moviéndonos de la infraestructura a la aplicación. El hallazgo de una vulnerabilidad de Ejecución Remota de Código (A03) con ZAP demostró el poder de las herramientas DAST para identificar fallos de inyección críticos que no son visibles a simple vista. Asimismo, la interceptación con Burp Suite confirmó que las credenciales por defecto (Tomcat, **A07**) son un punto de fallo de autenticación tan peligroso como un error en el código.

En conjunto, ambas fases ilustran que la seguridad no es un solo producto, sino un ciclo de vida. Comienza con la configuración segura de la infraestructura (parchear Nmap) y debe continuar con un análisis profundo de la aplicación (probar con ZAP y Burp) para defenderse eficazmente contra el marco de referencia **OWASP Top Ten**.

---

4. Bibliografía

OWASP Foundation. (2021). *OWASP Top Ten 2021*. Open Web Application Security Project. https://owasp.org/www-project-top-ten/

OWASP Foundation. (2023). *OWASP ZAP*. Open Web Application Security Project. https://www.zaproxy.org/