**Dibujo de una persona

Descripción generada automáticamente con confianza baja**



**Ataques informáticos Rocío Carlos y Pablo**

**Rocío Ceballos Mateos**

**Carlos González Martín**

**Pablo Méndez Montero**

**ÍNDICE**

**Introducción y explicación de la práctica. --------------------------------- Pág. 3**

**Consideraciones previas. ------------------------------------------------------- Pág. 4**

**Primera máquina (Injection). --------------------------------------------------- Pág. 8**

**Segunda máquina (Obssession). ------------------------------------------- Pág. 16**

**Tercera máquina (BreakMySSH). ------------------------------------------- Pág. 19**

**Conclusión. ------------------------------------------------------------------------- Pág. 21**

**Webgrafía. -------------------------------------------------------------------------- Pág. 21**

**INTRODUCCIÓN**

En este trabajo, vamos a explorar una pequeña parte del mundo de la seguridad informática a través de un ejercicio práctico: la resolución de tres máquinas de la plataforma DockerLabs.

Estas máquinas virtuales, especialmente configuradas para simular entornos vulnerables, nos permitirán poner en práctica nuestras habilidades en la identificación y explotación de vulnerabilidades comunes.

Para llevar a cabo esta tarea, utilizaremos un conjunto de herramientas esenciales en el ámbito de la seguridad informática: Docker, Nmap y Wireshark que también introduciremos brevemente en el primer punto del trabajo

Los objetivos principales del trabajo son:

* **Familiarizarnos** con las herramientas Docker, Nmap y Wireshark.
* **Adquirir** habilidades prácticas en la identificación y explotación de vulnerabilidades comunes en sistemas informáticos.
* **Analizar** el comportamiento de las herramientas utilizadas y su aplicación en escenarios reales.
* **Fortalecer** nuestros conocimientos en seguridad informática y hacking ético.

Al finalizar este trabajo, comprenderemos un poco mejor el día a día de los profesionales de la seguridad informática y estaremos mejor preparados para proteger los sistemas y datos de la futura empresa donde tengamos nuestro puesto de trabajo.

La estructura general del trabajo será la siguiente:

1. Introducción a las herramientas (Docker, Nmap, Wireshark)
2. Descripción de las máquinas DockerLabs a resolver
3. Metodología empleada en la resolución de cada máquina
4. Recomendaciones de seguridad para prevenir cada ataque
5. ***Consideraciones previas***

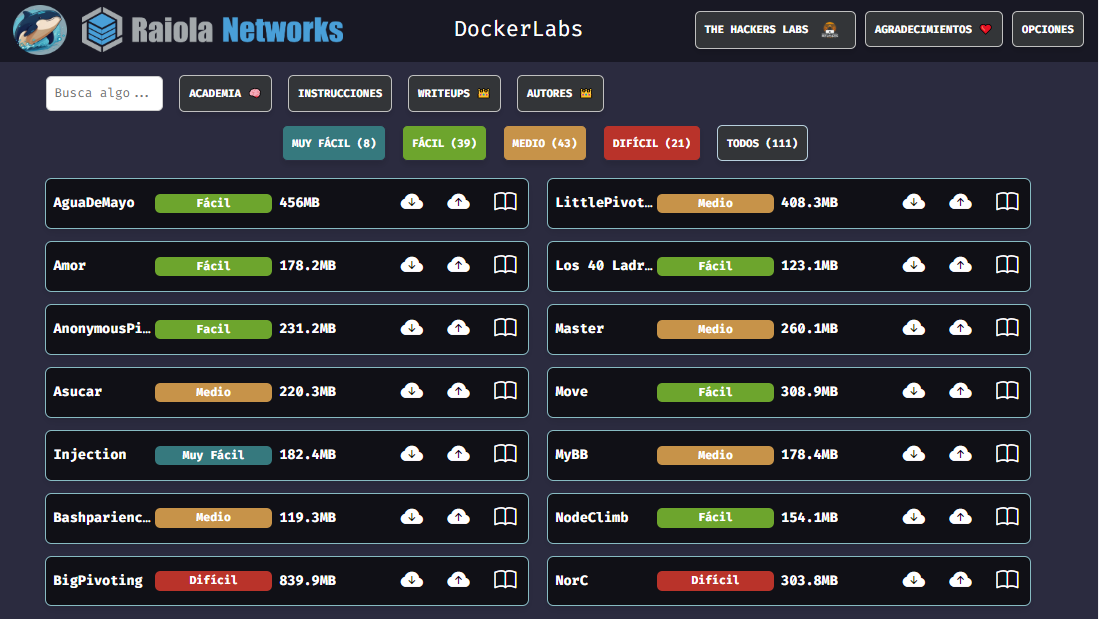
Antes de comenzar hablaremos un poco de los recursos y herramientas, que vamos a ver a lo largo del trabajo:

**DockerLabs.es**

Se trata de una web creada por un informático español llamado Mario Álvarez, que podemos encontrar en su canal de Youtube “El pingüino de Mario”.

Esta web nos ofrece una serie de laboratorios de ciberseguridad, (111 en el momento de escribir este trabajo).

Los laboratorios o máquinas, están divididos por dificultades (Muy fácil, Fácil, Medio y Difícil) por el nivel de conocimiento informático necesario para resolverlas.



Cada máquina consiste en un archivo .zip descargable desde mega que en su interior guarda un fichero .tar con un contenedor de Docker y un fichero .sh con un script que sirve para inicializar el contenedor.

El fichero del contenedor recibe el nombre de la máquina:

**nombre\_maquina.tar**

Mientras que el script de inicialización recibe siempre el mismo nombre:

**auto\_deploy.sh**

Sobra decir que debemos contar con una máquina virtual corriendo la distribución Kali Linux con Docker instalado si queremos resolver los laboratorios de DockerLabs.

Por último, cabe destacar que la función más didáctica de la página web es la sección de “Writeups” que tiene cada máquina (icono de libro a la derecha) donde los usuarios pueden colgar en forma de documento o de video sus guías de resolución para ayudar a nuevos usuarios a conseguirlas.

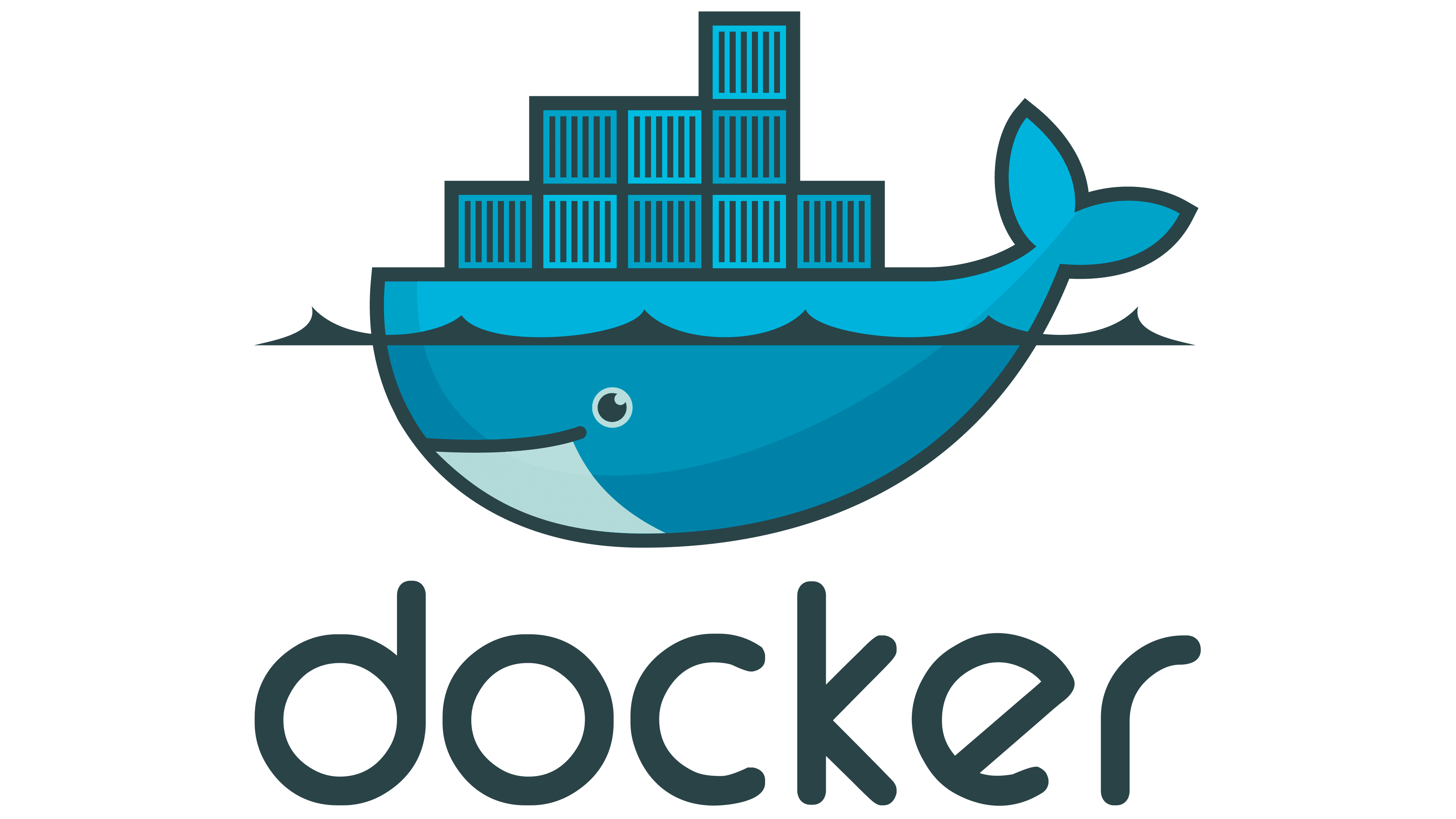
**Docker**

Hoy en día las aplicaciones informáticas cuentan con varios elementos. Uno de ellos es el código fuente pero también necesitan otras librerías o dependencias para su correcto funcionamiento.

Las dependencias son fragmentos de código, ajeno a la aplicación, que hacen posible el funcionamiento interno del código fuente de la aplicación, normalmente son dependencias relativas al lenguaje de programación o framework que se ha implementado en el desarrollo de la app pero también son imprescindibles otras dependencias diferentes que varían según el sistema operativo en el que esté ejecutando la aplicación.



Docker es una aplicación o servicio de código abierto que permite “containerizar” o “empaquetar” aplicaciones y sus dependencias en contenedores de tal manera que dichos contenedores almacenan todas las dependencias necesarias para la correcta ejecución de la aplicación en cuestión. Esto, entre otras cosas nos permite correr aplicaciones “encapsuladas” e independientes del sistema operativo utilizado ya que no necesitan interactuar con sus dependencias.

Podemos imaginar un contenedor como una caja que contiene todo lo necesario para que una aplicación funcione correctamente, sin importar el entorno en el que se ejecute.

Las máquinas de DockerLabs se descargan en estos contenedores y Docker tiene su base de funcionamiento en Linux por lo que deberemos descargar la herramienta desde la línea de comandos de Kali mediante el comando **sudo apt install docker.io**

La herramienta de Docker junto con la web DockerLabs nos ofrecen una forma fácil y rápida de desplegar máquinas con servicios vulnerables que nos sirvan como laboratorios de prácticas de ciberseguridad.

**Nmap**

Se trata de una herramienta de código abierto instalada por defecto en nuestra distribución Kali Linux. Nmap recibe su nombre por la abreviatura de Network Mapper ya que su función es precisamente esa, mapear o escanear la red y más en concreto las máquinas con una dirección IP determinada.

Nmap no solo puede descubrir los dispositivos disponibles en la red sino que también es capaz de escanear todos los puertos de un equipo de la red en busca de aquellos puertos que puedan estar abiertos.

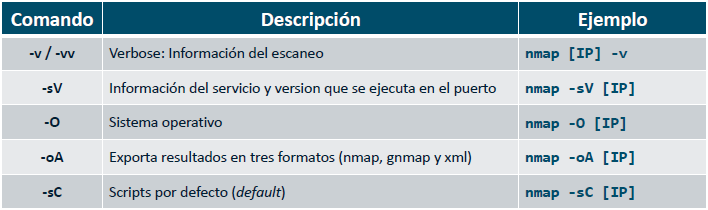


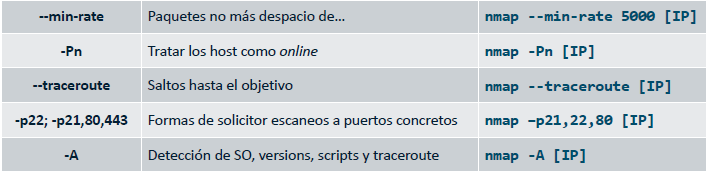
Además, Nmap es capaz de identificar el sistema operativo detrás del equipo y los servicios corriendo detrás de cada puerto con sus versiones que, de ser obsoletas, pueden presentar ciertas vulnerabilidades.

La herramienta cuenta con una versión con entorno gráfico denominada Zenmap pero la mas utilizada con diferencia es su versión de línea de comandos. Además será la que utilicemos durante el trabajo y para los ejemplos de esta introducción.

La sintaxis básica para el escaneo por defecto de un equipo es: **nmap [IP]** que por defecto hace el escaneo del Top1000 puertos más usados, si queremos escanear todos los puertos (65535) debemos ejecutar el comando con la opción -p- de la siguiente manera: **nmap -p- [IP]**

Con esta sintaxis básica, nmap solo nos devolverá aquellos puertos que encuentre abiertos sin más información, si queremos añadir valor al escaneo para conseguir más información podemos usar las diferentes opciones que tiene disponible el comando:





Algunas de las opciones para los tipos de escaneos más frecuentes son las siguientes:

**-sS**: TCP SYN (por defecto) Scan **-sT**: Connect() Scan

**-sA**: ACK Scan **-sU**: UDP sCAN

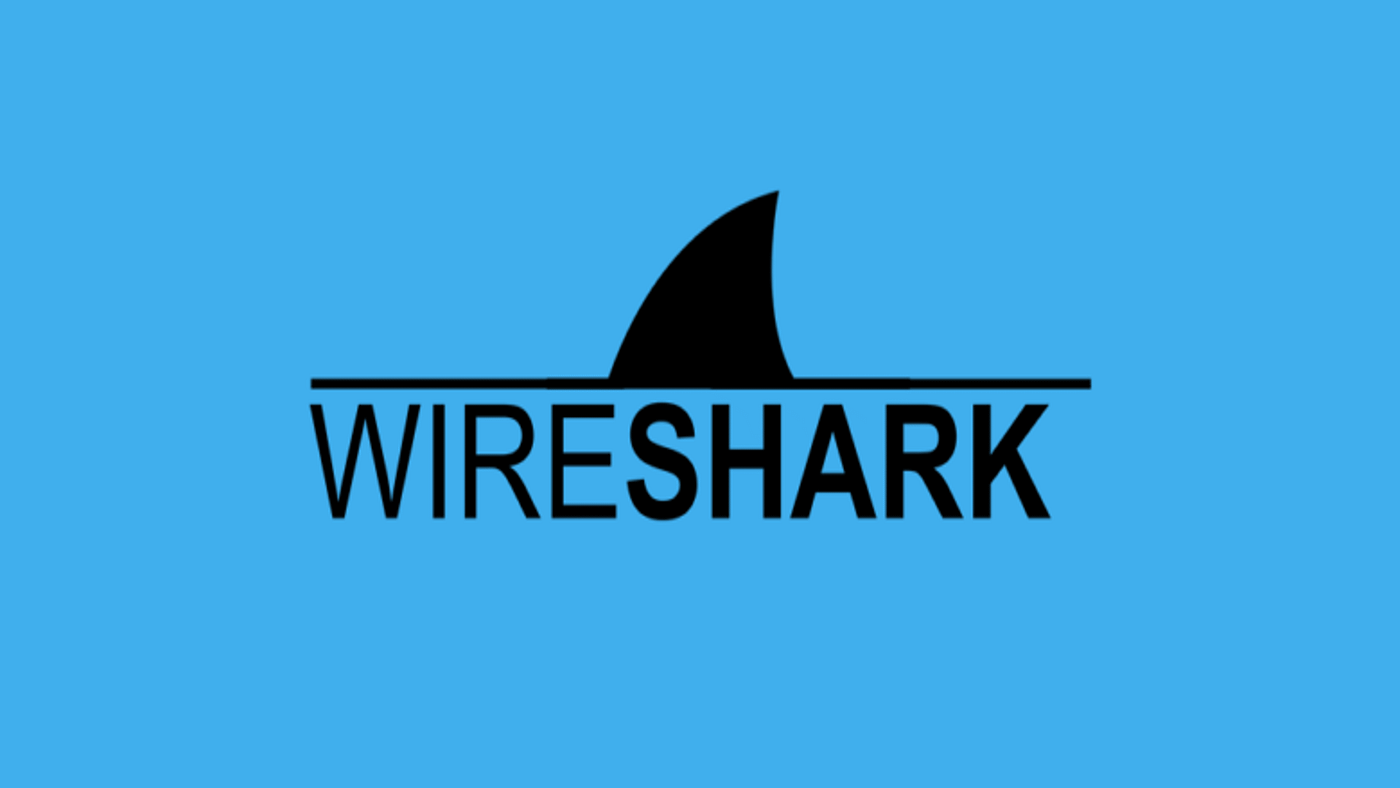
En definitiva, utilizamos la herramienta Nmap principalmente en la fase de escaneo o descubrimiento dentro de un ataque informático pero el uso para el que fue diseñada es brindar información de nuestra red para poder detectar posibles brechas o vulnerabilidades y poder ponerles solución.

**Wireshark**

**Wireshark** es una herramienta de análisis de tráfico de red, también conocida como sniffer. Es comúnmente utilizada por los administradores de redes y profesionales de la ciberseguridad pero como todas las herramientas de seguridad, tiene su cara B siendo utilizada también por los ciber delincuentes.

Wireshark también viene instalada por defecto en nuestra distribución Kali Linux y debido a su funcionamiento nos permite situarnos en una red y observar cada paquete de datos que circula por ella recibiendo información como su origen y destino o el contenido mismo del paquete.

Bien utilizada, esta herramienta puede ser muy didáctica ofreciendo una forma práctica de explorar el tráfico en tiempo real que nos permita observar el funcionamiento de la red y los protocolos de comunicación.



Algunas de las características principales que pueden sernos de gran utilidad son:

* Captura de paquetes en tiempo real y almacenaje en un fichero del historial para us posterior análisis.
* Filtrado de paquetes por diversos parámetros como protocolo de red, la IP o el puerto de origen y destino o la combinación de varias condiciones de filtrado.

Esto filtros nos ayudan a eliminar “ruido” para poder centrarnos en la búsqueda de lo que realmente nos interesa.

* Análisis detallado de cada paquete incluyendo multitud de información útil perteneciente a cada una de las capas del modelo TCP/IP
* Diversas opciones de visualización ya que Wireshark nos puede mostrar los datos capturados como diagramas de secuencia, gráficos de barras o en forma de análisis estadístico

1. ***Primera máquina (Injection).***

Esta máquina nos permite simular uno de los ataques más comunes y conocidos en el mundo de la ciber seguridad, se trata de una inyección SQL o SQL injection

La inyección SQL, a menudo abreviada como SQLi, es un tipo de ciber ataque en el que el atacante introduce código malicioso (SQL) en los campos de entrada de un formulario en una aplicación web con el objetivo de manipular la base de datos a la que dicha aplicación hará las consultas.

De esta forma el atacante "inyectara" una orden en el sistema, logrando que la base de datos ejecute acciones no autorizadas.

Las aplicaciones web funcionan usando la información proporcionada por el usuario en el formulario para construir parte de una consulta SQL y poder verificar su identidad, grabar datos nuevos, borrar o modificar algunos ya existentes en un proceso conocido como CRUD.

El atacante aprovecha este mecanismo para introducir en el campo de usuario parte de una sentencia SQL que cambiará convenientemente el significado de la consulta para que esta permita el acceso sea cual sea la contraseña introducida en su campo correspondiente.

Imaginemos un formulario de login que solicita un nombre de usuario y una contraseña. El atacante debe introducir el siguiente valor en el campo de nombre de usuario: **admin‘ OR 1=1-- -**

Esta entrada modificaría la consulta SQL original, permitiendo al atacante omitir la verificación de la contraseña y obtener acceso a la aplicación.

En el código fuente de la aplicación ya está predeterminada la consulta a la base de datos dentro de la tabla “usuarios” que verificará la existencia de ese nombre y contraseña para conceder o denegar el acceso.

Esta sentencia “SELECT” programada en el código simplemente deja un espacio concreto en la condición de la consulta que será rellenado concatenando el string de la consulta con los datos introducidos por el usuario.

Vamos a ver con un ejemplo el funcionamiento. En azul representamos la sentencia escrita en el código fuente de la aplicación y en rojo los datos recogidos del formulario para concatenarlos y así poder formar la consulta completa.

**SELECT nombre, contraseña FROM usuarios WHERE usuario =’ campo\_nombre ‘ AND contraseña = campo\_contraseña**

Ahora vamos a sustituir las variables campo\_nombre y campo\_contraseña por los valores reales introducidos por el atacante

**SELECT nombre, contraseña FROM usuarios WHERE usuario = ’admin‘ OR 1=1-- - AND contraseña = 1234**

Vamos a analizar la consulta que finalmente llega al servidor de base de datos:

* **admin'**: Proporciona un nombre de usuario factible y cierra la comilla simple que delimita el valor original del campo "usuario".
* **OR**: La cláusula OR proporciona una alternativa al valor del campo usuario para que el servidor valide si alguna de las dos condiciones que separa se cumplen.
* **1=1**: Esta condición siempre es verdadera, ya que 1 siempre es igual a 1. Esto implica que la parte de la condición original (usuario = 'admin') se vuelve irrelevante.
* **--**: Esto inicia un comentario de una sola línea en SQL. Todo lo que viene después (en este caso, la parte ' AND contraseña = '1234') es ignorado por el servidor de base de datos.

En resumen, ahora la consulta solo impone la necesidad de cumplir una condición para garantizar el acceso a la base de datos y nos hemos asegurado que dicha condición siempre se cumpla ya que 1 siempre es igual a 1, de esta manera conseguimos un acceso no autorizado a la aplicación Web que nos permita seguir buscando vulnerabilidades dentro de ella. Este es un claro ejemplo del concepto de seguridad en alcachofa.

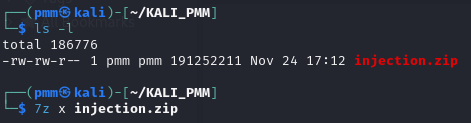
**Writeup del ataque con la máquina de DockerLabs**

El primer paso es crear una máquina virtual con Kali Linux y Docker instalado, recordemos que el comando de instalación para Docker es:

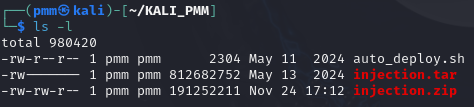
**sudo apt install docker.io**

Una vez que tenemos Docker instalado nos descargamos el fichero injection.zip de la web de DockerLabs y los descomprimimos con el comando:

**7z x injection.zip**



Comprobamos que se han extraído dos ficheros del .zip que corresponden con el contenedor (injection.tar) y el script de inicialización (aluto\_deploy.sh)

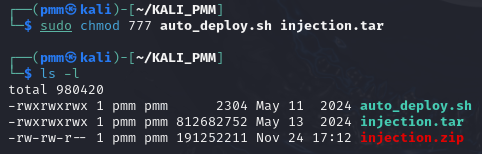


Observamos que los ficheros extraídos tienen una combinación de permisos que puede ser problemática si queremos manipularlos con un usuario que no sea root

Para solucionar este problema asignamos permisos totales a ambos ficheros con el comando

**Sudo chmod 777 auto\_deploy.sh injection.tar**

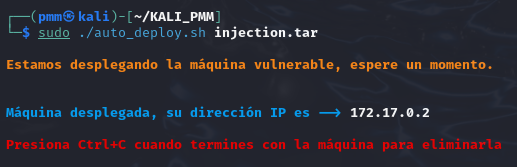
Y comprobamos de nuevo los permisos que tienen ahora los ficheros.



Para desplegar la máquina simplemente debemos ejecutar el script pasándole como parámetro el fichero del contenedor de Docker con el siguiente comando.

**./auto\_deploy.sh injection.tar**

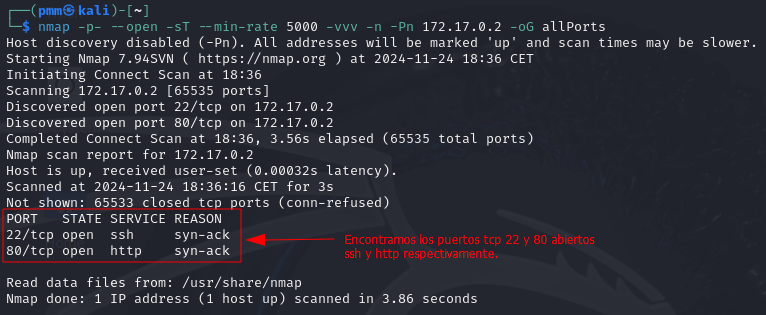
Y observamos que se nos informa de la dirección IP asignada servicio desplegado así como de las instrucciones para eliminar el contenedor cuando ya no necesitemos seguir usándolo. (simplemente pulsando Ctrl+c)



Ahora que tenemos el contenedor desplegado con su IP (172.17.0.2) podemos iniciar la primera fase del ataque denominada “**Reconocimiento**”. Para ellos vamos a utilizar la herramienta Nmap ejecutando el siguiente comando contra la IP del servicio:

**nmap -p- --open -sT --min-rate 5000 -vvv -n -Pn 172.17.0.2 -oG allPorts**

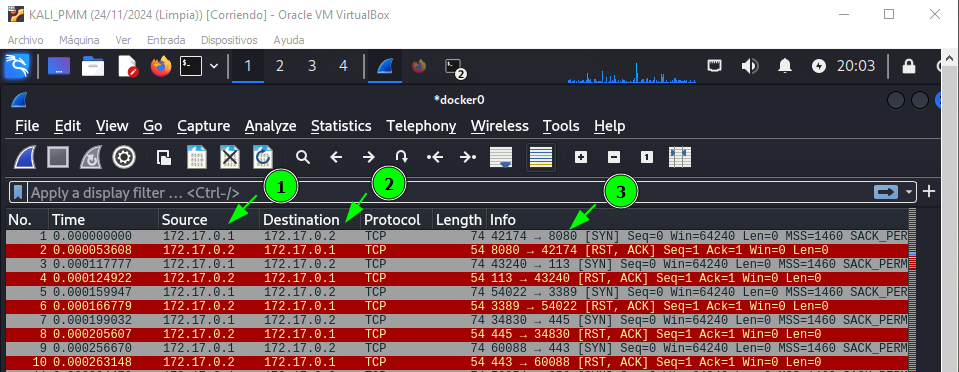
Entre otras cosas, el comando lanzado nos informará de los puertos abiertos, el protocolo de transporte que están utilizando así como el servicio que se ha encontrado corriendo detrás de ellos.



Encontramos los puertos 22 y 80 tcp abiertos y con los servicios ssh y http corriendo respectivamente, esto quiere decir que el equipo del contenedor admite conexiones por el protocolo ssh en el puerto 22 y que tiene levantado un servidor web detrás del puerto 80.

Antes de continuar con el siguiente paso vamos a echar un vistazo a que ha recogido Wireshark después de el lanzamiento de este escaneo de puertos.

Anteriormente a la ejecución de Nmap, pusimos a escuchar Wireshark por la interfaz de red adjudicada a la máquina vulnerable (docker0). En la siguiente imagen podemos observar la captura del tráfico de red mientras se ejecutaba Nmap sobre ella:



* Señalado con el número 1 tenemos la IP de origen del comando Nmap, en este caso nuestra IP privada 172.17.0.1
* Señalado con el número 2 tenemos la IP de destino del comando Nmap, en este cso es la IP asignada a la máquina vulnerable 172.17.0.2
* Señalado con el número 3 tenemos el puerto sobre el cual Nmap ejecuta ping para averiguar su estado.

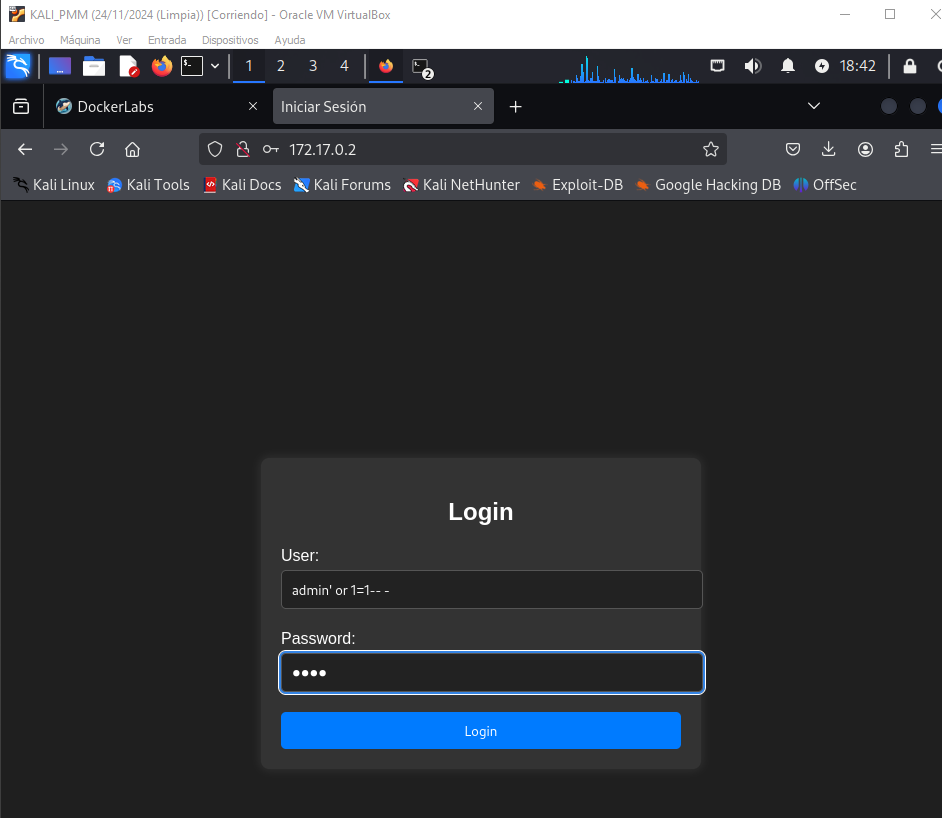
Debajo de cada línea gris representada por la petición tenemos una línea roja representando la respuesta del equipo objetivo.

Continuando con el ataque, escribimos en nuestro navegador web la dirección IP y el puerto del servidor web que ya sabemos que esta corriendo en la máquina. Al estar detrás del puerto 80 el navegador lo puede interpretar sin necesidad de especificárselo ya que es el puerto por defecto asignado al protocolo http y solo será necesario escribir la dirección IP para acceder al index del servidor.

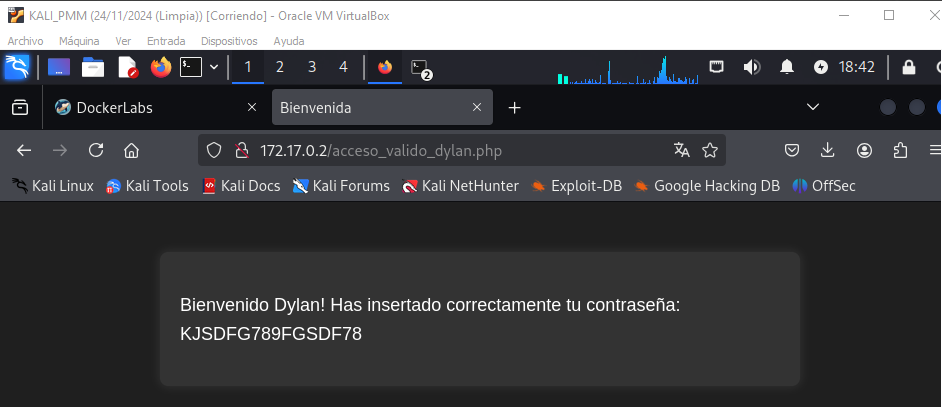
Pasamos a la segunda fase del ataque conocida como **“Explotación”**, podemos observar que el servidor nos devuelve el formulario de login para conectarnos al servicio web en el que deberemos introducir la secuencia típica de las inyecciones SQL explicadas un poco más arriba.

Campo User: **admin‘ OR 1=1-- -**

Campo Password: **1234** (o cualquier otra)



Al pulsar en el botón de Login observamos que hemos ganado acceso al servicio:

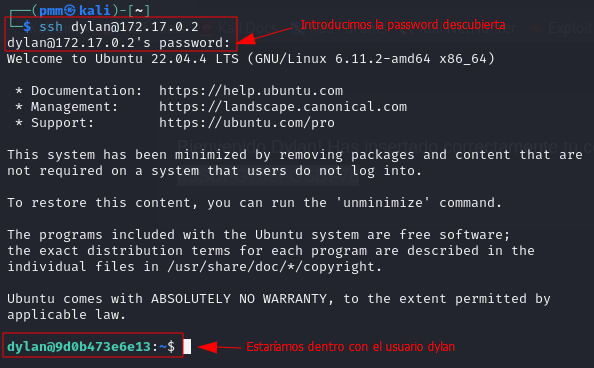


Una vez dentro se nos informa de nuestro nombre de usuario y contraseña que podemos contemplar como posibles datos válidos para la conexión por ssh.

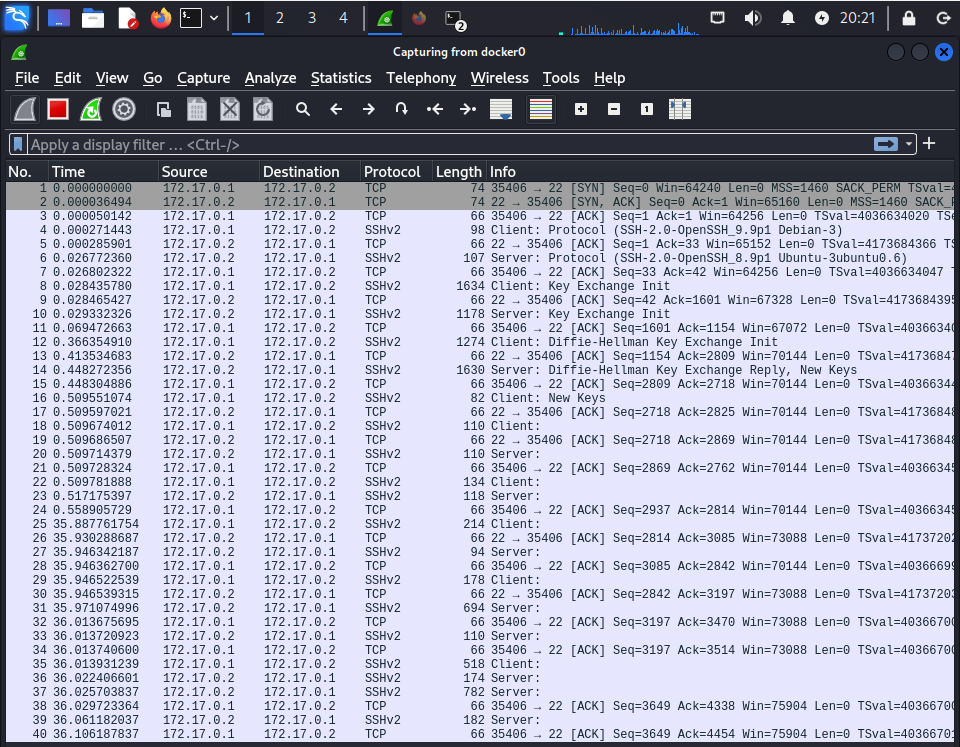
A este tipo de hallazgos se les conoce como banderas o flags en las máquinas de ciberseguridad, de ahí su nombre común CTF o Capture The Flag en inglés.

A continuación abrimos una nueva terminal y probamos la conexión ssh a la dirección IP de la máquina con los datos de usuario y contraseña que hemos obtenido del servidor web mediante la inyección SQL. Para ello debemos ejecutar el comando:

**Ssh** [**dylan@172.17.0.2**](mailto:dylan@172.17.0.2)e introducir después la clave obtenida



Observamos que hemos conseguido acceso al servidor con estas credenciales de Dylan pero vamos a ver que datos ha capturado Wireshark durante la conexión ssh:



Se aprecian de nuevo las trazas de la conversación típica de conexiones desde la IP local a la IP de la máquina mediante el protocolo ssh por el puerto 22.

Volviendo a la terminal recordamos que estamos dentro del servidor con el usuario Dylan, pero él es un usuario común, poco podemos hacer con sus permisos a si que es hora de pasar a la tercera fase del ataque conocida como **“Escalada de privilegios”**

Como primer paso vamos a buscar en el sistema operativo del servidor (Linux) si existe algún fichero binario con permisos SUID del usuario root con el siguiente comando:

**find / -perm -4000 -user root 2>/dev/null**

El comando find busca ficheros en el directorio indicado, al indicar que busque en el directorio raíz (/) simplemente buscará en todo el sistema.

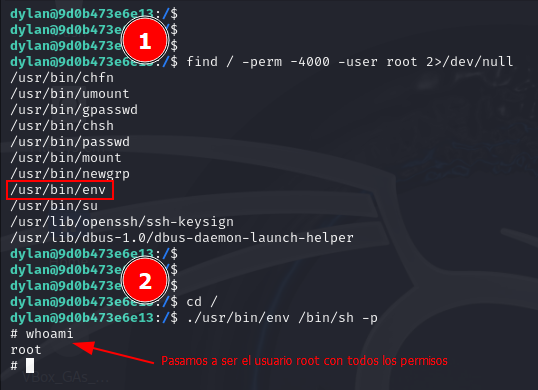
Por otro lado especificamos que busque ficheros con permisos SUID (4000) del usuario root y que redirija la salida de error al dev/null para no ensuciar el resultado por pantalla.

Los ficheros resultantes de esta búsqueda tienen una característica importante en común, los permisos SUID. (Set User ID) es un bit especial asociado a los archivos ejecutables en sistemas operativos tipo Unix, como Linux. Cuando se asigna este bit a un archivo, cualquier usuario que ejecute ese archivo lo hará con los privilegios del propietario del archivo, en lugar de los propios y como hemos indicado en el filtro del find que el propietario debe ser root, podremos ejecutarlos con permisos de root.

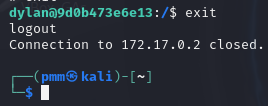
Sabiendo esto, elegimos por ejemplo el binario “env” y ejecutamos desde la raíz el siguiente comando:

**./usr/bin/env /bin/sh -p**

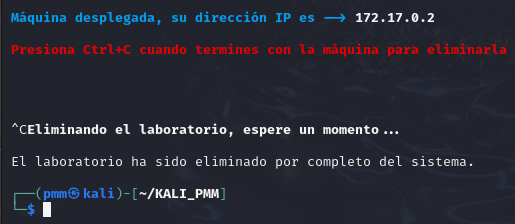
Para garantizar acceso a la Shell como el usuario root y comprobarlo ejecutando el comando Whoami



Por último podemos cerrar la conexión ssh con el servidor para volver a nuestro prompt de sistema:



Y volver a la primera terminal que abrimos, en la que desplegamos el contenedor, para cerrarlo y eliminarlo pulsando (Ctrl+c) ahora que ya hemos finalizado el laboratorio.



**¿Cómo podemos prevenir las inyecciones SQL?**

La inyección SQL es una vulnerabilidad común y peligrosa que puede comprometer la seguridad de cualquier aplicación web. Es fundamental implementar las medidas de seguridad necesarias para prevenir este tipo de ataques, algunas de estas son:

* **Validación de entradas:** Siempre valida y filtra los datos proporcionados por el usuario antes de incluirlos en las consultas SQL.
* **Uso de parámetros:** Utiliza parámetros en lugar de concatenar directamente los valores de entrada en las consultas SQL.
* **Minimización de privilegios:** Asigna a las cuentas de base de datos los mínimos privilegios necesarios.
* **Actualizaciones de software:** Mantén actualizados todos los componentes de tu aplicación y base de datos.
* **WAF (Web Application Firewall):** Implementa un WAF para detectar directamente escaneos de puertos y bloquear ataques de inyección SQL.

1. ***Segunda máquina (Obsession).***

Esta máquina aprovecha una vulnerabilidad típica en los servidores FTP como es la de mantener permisos totales para los usuarios conectados con el perfil “anónimo”.

**Writeup del ataque con la máquina de DockerLabs**

Comenzamos con el despliegue de la máquina, recordemos que previamente debemos descargarla de DockerLabs y tener instalado Docker en el sistema.



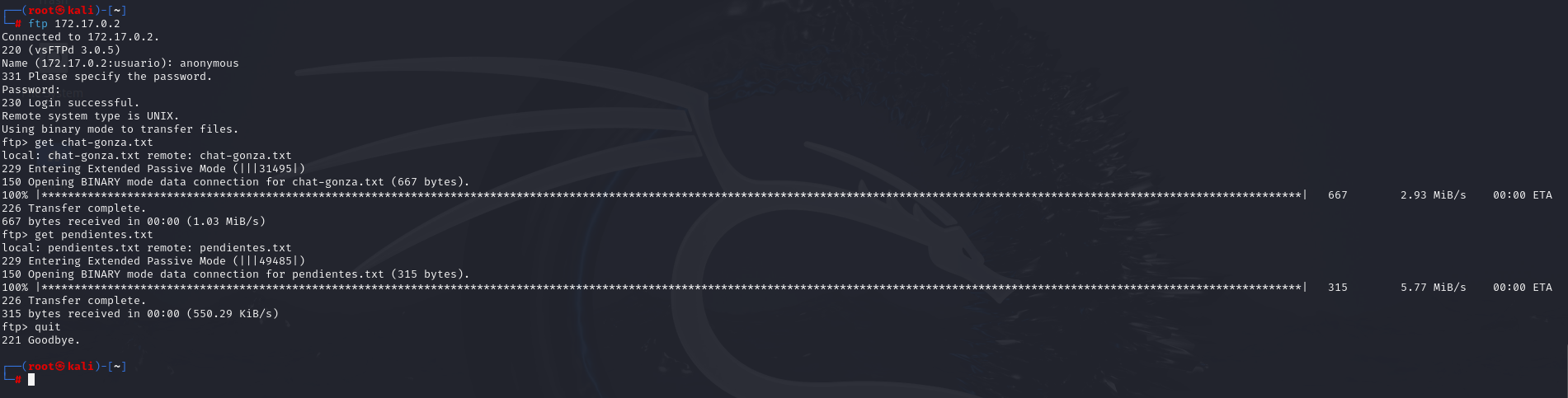
**Búsqueda de puertos**

Ahora veremos qué servicios y que puertos hay abiertos y que versión tiene

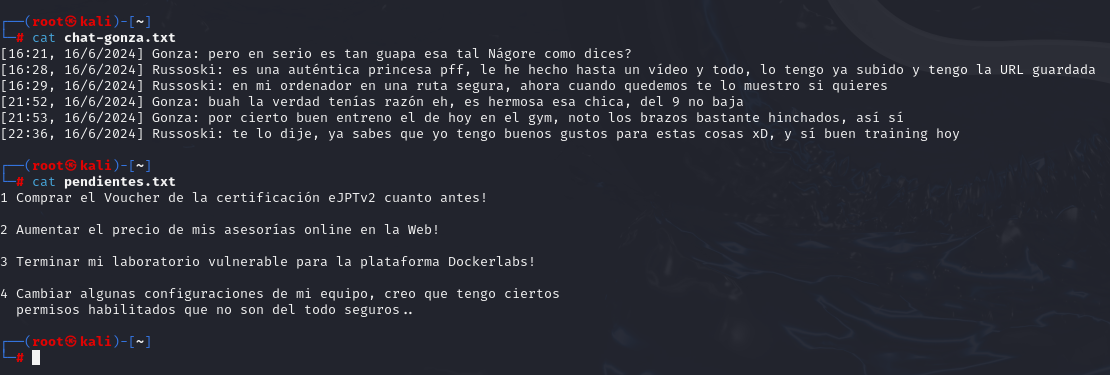


**Conexión mediante ftp**

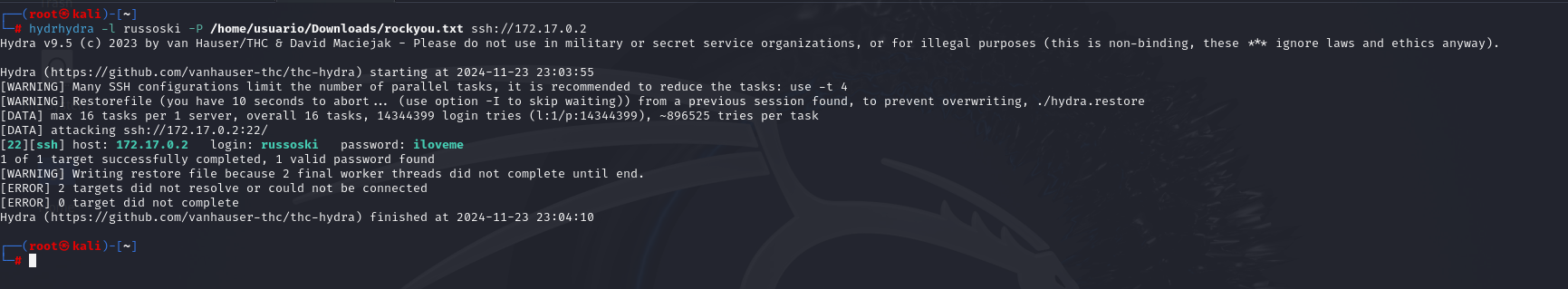
Vemos que hay un puerto ftp, vamos a probar si podemos conectarnos mediante usuario anónimo y descargarnos los archivos que nmap nos ha proporcionado



Una vez que hemos descargado los archivos, vemos que contienen los archivos



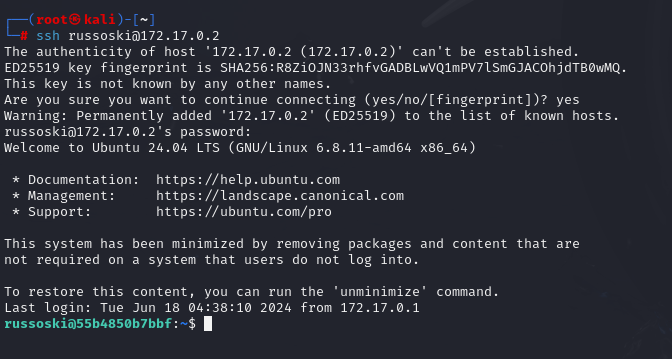
**Ataque con fuerza bruta**

Vemos que hay un usuario llamado russoski, vamos a probar con hydra un ataque de fuerza bruta mediante el archivo rockyou.txt que es un fichero de contraseñas.

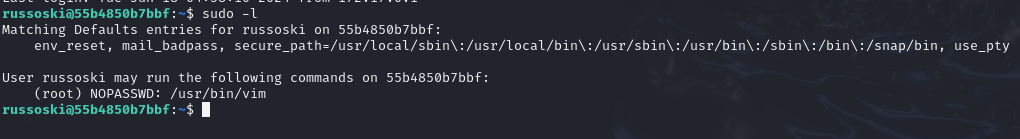
Vemos que hay un usuario llamado russoski y su contraseña es iloveme

**Conexión mediante ssh**

Como vimos en nmap que tiene un servicio ssh y un ftp, vamos a conectarnos mediante el usuario russoski y la contraseña iloveme.

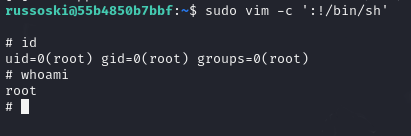


Una vez que nos hemos conectado miraremos como podemos escalar privilegios



**Escalada de privilegios.**

Mediante el comando vim cambiaremos a sudo ya que tiene una Shell distinta a la de bash



Finalmente comprobamos que somos usuario root con el comando whoami

**¿Cómo podemos prevenir este ataque por FTP?**

Deshabilitando el usuario anónimo, y que tenga archivos que no sean muy importantes, ya que viendo los archivos vemos los usuarios que están habilitados en la máquina, y podemos hacer un ataque mediante el comando hydra para saber la contraseña del usuario.

1. ***Tercera máquina (Breakmyssh).***

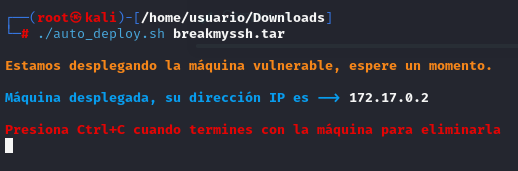
En esta máquina vamos a baipasear el protocolo ssh

**Writeup del ataque con la máquina de DockerLabs**

Como en los ejemplos anteriores debemos descargar la máquina de DockerLabs

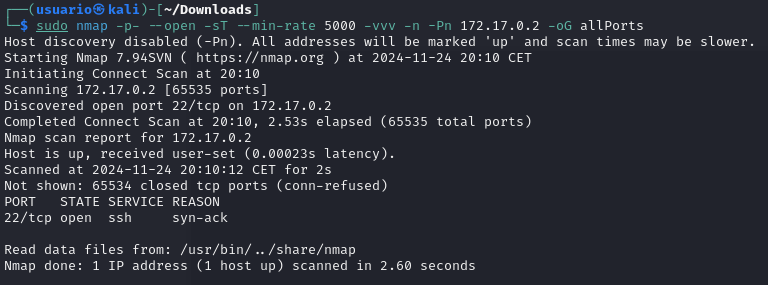
**Despliegue de la máquina**

Vamos a usar la maquina break my ssh, desplegaremos la máquina, importante hacer esto con root, si no nos daría fallo



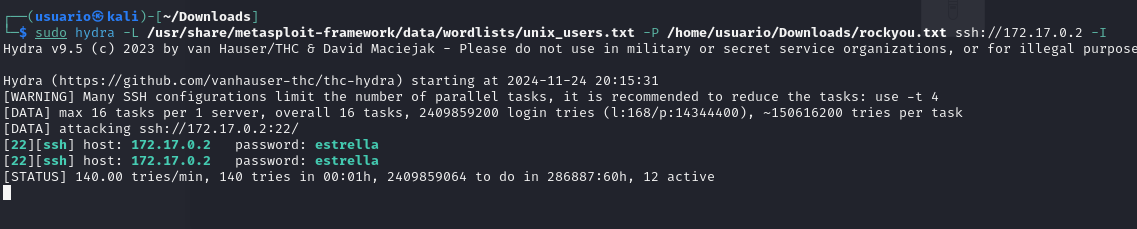
**Búsqueda de puertos**

Mediante el comando nmap haremos un escaneo general de los puertos habilitados

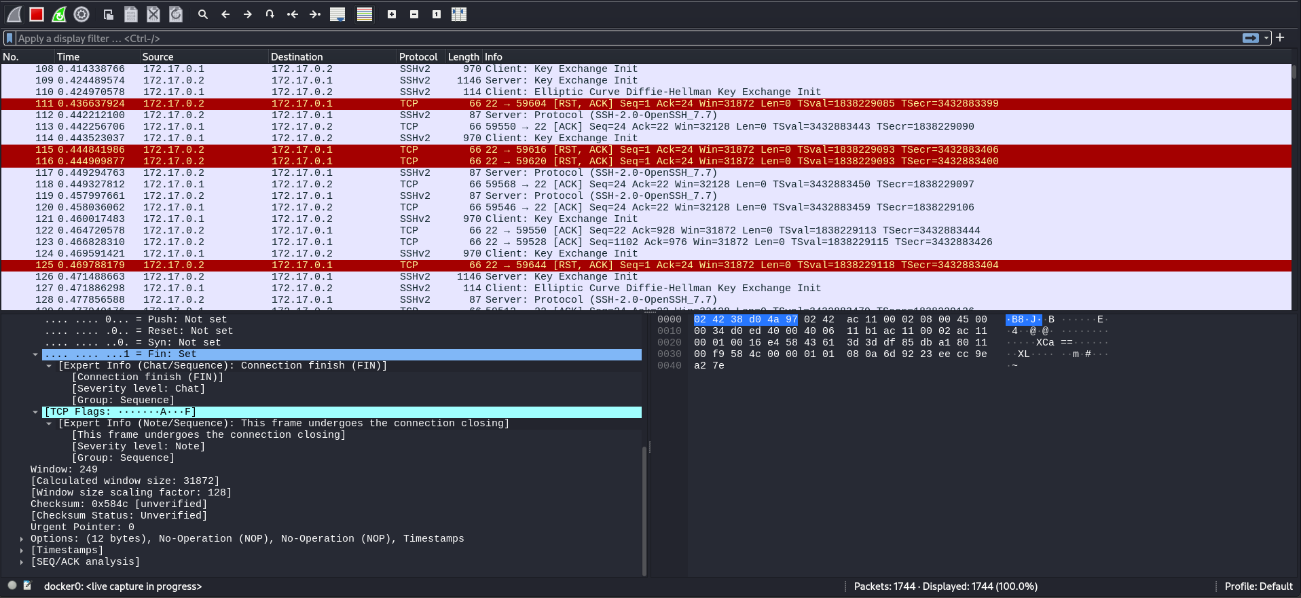


**Conexión ssh**

Vemos que hay un puerto ssh habilitado, usaremos el comando hydra y la base de datos de contraseñas de rockyou.txt para romper la contraseña y poder conectarnos

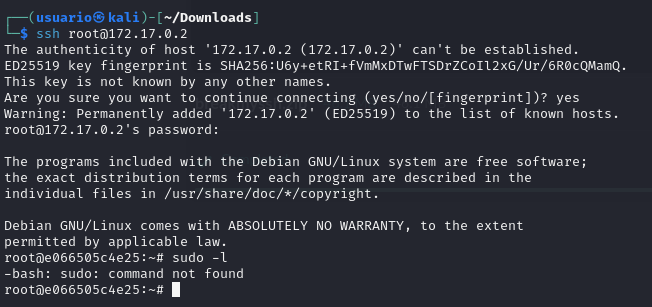


Vamos a usar wireshark mientras hacemos el ataque de fuerza bruta de ssh



**Conexión**

Una vez que tenemos el usuario y contraseña, iniciaremos sesión con ssh, como no nos ha indicado el comando hydra el usuario probaremos con el usuario root, que es el usuario con mayor privilegios del un sistema linux.



Aquí no hace falta escalada de privilegios, ya que hemos entrado mediante el usuario root.

**¿Cómo podemos prevenir este ataque por SSH?**

En esta maquina nos conectamos mediante root, esto es una falla muy grande, ya que tenemos permisos totales para hacer lo que queramos en la máquina. Podemos deshabilitar la conexión mediante root a una maquina por ssh, que por defecto esta denegado el acceso, también podemos denegar el acceso a los usuarios y conectarnos mediante certificados, comparando los certificados que tiene el servidor y verificar la autenticación del usuario.

1. ***Conclusión.***

A través de la resolución de las máquinas DockerLabs y el uso de herramientas como Docker, Nmap y Wireshark, hemos podido ver de manera práctica los conceptos fundamentales de la seguridad informática.

Cada máquina ha representado un desafío y metodología única, permitiéndonos aplicar diferentes técnicas de hacking ético y análisis de vulnerabilidades.

Podemos sacar tres conclusiones claras del trabajo:

* La seguridad informática es un campo en constante evolución y es fundamental mantenerse actualizado sobre las últimas amenazas y técnicas de ataque.
* La prevención es siempre mejor que la cura. Implementar medidas de seguridad desde el diseño de un sistema es crucial para reducir el riesgo de sufrir ataques.
* La colaboración y el intercambio de conocimientos son fundamentales para mejorar la seguridad de los sistemas informáticos.

1. ***Webgrafía***

* **DockerLabs**:

<https://dockerlabs.es/>

* **Chat GPT**:

<https://openai.com/index/chatgpt/>

* **Gemini**:

<https://gemini.google.com/>

* **Google**:

<https://google.es/>