Міністерство освіти і науки України
Державний університет «Житомирська політехніка»
Факультет інформаційно-комп'ютерних технологій
Кафедра комп'ютерної інженерії та кібербезпеки

ПОЯСНЮВАЛЬНА ЗАПИСКА

до випускної кваліфікаційної роботи бакалавра

на тему: «Проєкт платформи для проведення навчальних кібербезпекових досліджень за напрямом Penetration Testing у форматі CTF»

Виконав студент 4-го курсу групи КБ-20-1
спеціальності 125 «Кіберберзпека»
_____ Вадим НЕСТЕРОВ
Керівник: завідувач кафедри КІ та КБ, к.т.н., доцент,
_____ Андрій ЄФІМЕНКО
Рецензент: доцент кафедри КІ та КБ, к.т.н, доцент,
_____ Юрій РОССІНСЬКИЙ

Житомир – 2024

Державний університет «Житомирська політехніка»
Факультет інформаційно-комп'ютерних технологій
Кафедра комп'ютерної інженерії та кібербезпеки
Спеціальність 125 «Кібербезпека»
Освітня програма «Кібербезпека»

ЗАТВЕРДЖУЮ
Завідувач
кафедри комп'ютерної
інженерії та кібербезпеки
Андрій ЄФІМЕНКО
5 березня 2024 р.

ЗАВДАННЯ

на випускну кваліфікаційну роботу бакалавра

Студент: Нестеров Вадим Юрійович

Тема роботи: «Проєкт платформи для проведення навчальних кібербезпекових досліджень за напрямом Penetration Testing у форматі CTF»

Затверджена Наказом університету від 5 березня 2024 р. № 98

Термін здачі студентом закінченої роботи 10 червня 2024 р.

Вихідні дані роботи (зазначається об'єкт і предмет дослідження):

– Об'єктом дослідження є процес проєктування платформи для проведення навчальних кібербезпекових досліджень за напрямом Penetration Testing у форматі CTF;

– Предметом дослідження є методи та технології проєктування платформи для проведення навчальних кібербезпекових досліджень за напрямом Penetration Testing у форматі CTF.

Обладнання та програмне забезпечення:

1. Oracle VM VirtualBox, GNS3;
2. Маршрутизатор Cisco 7206VXR/NPE-G2;
3. Комутатор Cisco WS-C2960-24TT-L;
4. Операційні системи Ubuntu, Kali Linux;
5. Програмне забезпечення Docker, OpenVPN;
6. CTFd.

Консультанти з випускної кваліфікаційної роботи із зазначенням розділів, що їх стосуються:

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв
1	Єфіменко А.А.	20.02.2024	20.02.2024
2	Єфіменко А.А.	01.04.2024	01.04.2024
3	Єфіменко А.А.	26.04.2024	26.04.2024

Студент _____ Вадим НЕСТЕРОВ

Керівник _____ Андрій ЄФІМЕНКО

Календарний план

№ з/п	Назви етапів випускної кваліфікаційної роботи	Термін виконання етапів роботи	Примітка
1	Постановка завдання	20.02.2024	Виконано
2	Розробка плану роботи	25.02.2024	Виконано
3	Аналіз розгортання платформ, Penetration Testing та формату CTF	01.03.2024 – 31.03.2024	Виконано
4	Проєктування мережевої інфраструктури для розгортання платформи та розробка структури завдань	01.04.2024 – 25.04.2024	Виконано
5	Створення мережевої інфраструктури та налаштування обладнання	26.04.2024 – 29.04.2024	Виконано
6	Розгортання та налаштування платформи	30.04.2024 – 09.05.2024	Виконано
7	Наповнення платформи завданнями	10.05.2024 – 17.05.2024	Виконано
8	Тестування роботи мережі та функціоналу платформи	18.05.2024 – 20.05.2024	Виконано
9	Написання пояснювальної записки	21.05.2024 – 08.06.2024	Виконано
10	Отримання відгуку та рецензій		
11	Захист кваліфікаційної роботи		

Студент _____ Вадим НЕСТЕРОВ

Керівник _____ Андрій ЄФІМЕНКО

АНОТАЦІЯ

Випускна кваліфікаційна робота бакалавра являє собою проєкт платформи для проведення навчальних кібербезпекових досліджень за напрямом Penetration Testing у форматі CTF. Пояснювльна записка складається з переліку умовних скорочень, вступу, трьох розділів, висновків, списку використаної літератури та додатків. Текстова частина викладена на 134 сторінки друкованого тексту. Пояснювальна записка має 4 сторінки додатків. Список використаних джерел містить 22 найменування і займає 2 сторінки. В роботі наведено 212 рисунків та 3 таблиці. Загальний обсяг роботи склав 148 сторінок.

Метою роботи є розгортання інтерактивної платформи для проведення навчальних кібербезпекових досліджень за напрямом Penetration Testing у форматі CTF.

У роботі проаналізовано теоретичні засади та наявні рішення для розгортання платформ. Спроєктовано мережеву інфраструктуру, розроблено схему адресації для всіх пристроїв в мережі з урахуванням подальшого масштабування, здійснено вибір відповідного мережевого обладнання та проведено налаштування мережі. На віртуальному сервері встановлено операційну систему Ubuntu та ряд програмного забезпечення, зокрема OpenVPN та Docker. Обрано п'ять VM-жертв. Проведено розгортання, налаштування та наповнення платформи на базі готового рішення CTFd. Протестовано коректність роботи мережі та функціоналу платформи.

Ключові слова: ПЛАТФОРМА, PENETRATION TESTING, КІБЕРБЕЗПЕКА, НАВЧАННЯ, ДОСЛІДЖЕННЯ, CTF.

					ККІКБ.КР.Б – 125 – 13 – ПЗ		
Змн.	Арк.	№ докум.	Підпис	Дата			
Розроб.		Нестеров В.Ю.		15.06.24	Проєкт платформи для проведення навчальних кібербезпекових досліджень за напрямом Penetration Testing у форматі CTF	Літ.	Арк. 4 / Аркушів 148
Керівник		Єфіменко А.А.		15.06.24			
Реценз.		Россінський Ю.М.		15.06.24		ФІКТ Гр. КБ-20-1	
Н. контр.		Єфіменко А.А.		15.06.24			
Зав. каф.		Єфіменко А.А.		15.06.24			

ANNOTATION

The bachelor's thesis is a project of a platform for conducting educational cybersecurity research in the field of Penetration Testing in the CTF format. The explanatory note consists of a list of abbreviations, an introduction, three chapters, conclusions, a list of references, and appendices. The text part is set out on 134 pages of printed text. The explanatory note includes 4 pages of appendices. The list of references includes 22 titles and occupies 2 pages. The paper contains 212 figures and 3 tables. The total volume of the paper is 148 pages.

The purpose of this paper is to develop an interactive platform for conducting cybersecurity training research in the area of Penetration Testing in the CTF format.

The paper analyses the theoretical foundations and existing solutions for deploying platforms. The network infrastructure was designed, an addressing scheme was developed for all devices in the network, taking into account further scaling, the appropriate network equipment was selected, and the network was configured. The virtual server is equipped with the Ubuntu operating system and a number of software, including OpenVPN and Docker. Five VM victims are selected. The platform was deployed, configured, and populated based on the off-the-shelf CTFd solution. The correctness of the network and platform functionality is tested.

Keywords: PLATFORM, PENETRATION TESTING, CYBERSECURITY, TRAINING, RESEARCH, CTF.

ЗМІСТ

ВСТУП 8

РОЗДІЛ 1 ТЕОРЕТИЧНІ АСПЕКТИ РОЗГОРТАННЯ ПЛАТФОРМ, PENETRATION TESTING ТА ФОРМАТУ CTF 9

1.1 Сервери та їх типи 9

1.2 Огляд та аналіз гіпервізорів 18

1.3 Поняття Penetration Testing 24

1.4 Концепція та типи CTF 30

1.5 Огляд наявних рішень для розгортання платформ у форматі CTF 32

1.6 Ресурс готових VM-жерт VulnHub 35

РОЗДІЛ 2. ПРОЄКТУВАННЯ МЕРЕЖЕВОЇ ІНФРАСТРУКТУРИ ДЛЯ РОЗГОРТАННЯ ПЛАТФОРМИ ТА РОЗРОБКА СТРУКТУРИ ЗАВДАНЬ 37

2.1. Проєктування мережевої інфраструктури 37

2.2. Розробка схеми адресації мережі 40

2.3 Вибір мережевого обладнання 41

2.4 Вибір операційних систем, програмного забезпечення та VM-жертв 43

2.5 Розробка структури завдань 45

РОЗДІЛ 3 РОЗГОРТАННЯ, НАЛАШТУВАННЯ, ТЕСТУВАННЯ РОБОТИ МЕРЕЖІ ТА ФУНКЦІОНАЛУ ПЛАТФОРМИ 53

3.1 Налаштування мережевого обладнання 53

3.2 Розгортання та налаштування платформи 55

3.3 Встановлення та налаштування OpenVPN 61

3.4 Наповнення платформи завданнями 65

3.5 Тестування роботи мережі та функціоналу платформи 73

ВИСНОВКИ 141

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 142

ДОДАТКИ 144

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

IP – Internet Protocol

IPv4 – Internet Protocol version 4

UDP – User Datagram Protocol

TCP – Transmission Control Protocol

NAT – Network Address Translation

OSPF – Open Shortest Path First

DNS – Domain Name System

DHCP – Dynamic Host Configuration Protocol

MAC – Media Access Control

SSH – Secure Shell

SMTP – Simple Mail Transfer Protocol

POP3 – Post Office Protocol Version 3

LFI – Local File Inclusion

RCE – Remote Code Execution

CTF – Capture The Flag

ОС – операційна система

ПЗ – програмне забезпечення

ВСТУП

Актуальність теми. В сучасних умовах, коли кількість кібератак та загроз кібербезпеці зростає з кожним роком, важливість ефективного захисту інформації набуває особливого значення. У цьому контексті розгортання платформ для проведення навчальних кібербезпекових досліджень за напрямом Penetration Testing у форматі CTF стає важливим інструментом для навчання та перевірки навичок виявлення вразливостей з подальшим їх усуненням, що робить дану тему актуальною та важливою для дослідження.

Метою кваліфікаційної роботи є розгортання інтерактивної платформи для проведення навчальних кібербезпекових досліджень за напрямом Penetration Testing у форматі CTF.

Визначена мета обумовлює наступні завдання:

– Аналіз теоретичних аспектів та наявних рішення для розгортання платформ у форматі CTF;

– Розробка проєкту мережевої інфраструктури;

– Розробка структури завдань, підказок та нарахування балів за виконання;

– Створення та налалштування мережевої інфраструктури;

– Розгортання, налаштування та наповнення платформи;

– Тестування роботи мережі та функціоналу платформи.

Об’єктом дослідження є процес проєктування платформи для проведення навчальних кібербезпекових досліджень за напрямом Penetration Testing у форматі CTF.

Предметом дослідження є методи та технології проєктування платформи для проведення навчальних кібербезпекових досліджень за напрямом Penetration Testing у форматі CTF.

У процесі виконання кваліфікаційної роботи було розгорнуто платформу, яка надає можливість проводити навчальні кібербезпекові дослідження за напрямом Penetration Testing у форматі CTF.

РОЗДІЛ 1 ТЕОРЕТИЧНІ АСПЕКТИ РОЗГОРТАННЯ ПЛАТФОРМ, PENETRATION TESTING ТА ФОРМАТУ CTF

1.1 Сервери та їх типи

Сервери є невід'ємною частиною для розгортання будь-якої платформи. Вони забезпечують необхідну інфраструктуру для обробки даних, зберігання інформації та безперебійного доступу. Без надійних серверів неможливо уявити стабільну та ефективну роботу сучасних технологічних систем.

Сервери – це апаратні або програмні пристрої, які надають ресурси іншим комп'ютерам (клієнтам) через мережу. Ці високопродуктивні, відмовостійкі комп'ютери призначені для безперервної роботи протягом тривалого часу [10].

Сервери складаються з різних компонентів:

– Материнська плата;
– Центральний процесор (ЦП);
– Оперативна пам'ять (ОЗП);
– Блок живлення (БЖ);
– Графічний процесор (GPU);
– Накопичувачі;
– Мережа;
– Шасі (корпус);
– Програмні компоненти.

Материнська плата – це головна плата в комп'ютері. Її завдання полягає в тому, щоб зібрати всі компоненти сервера в одному місці і забезпечити зв'язок між ними.

Це такі компоненти, як чіпсет, слоти PCIe, роз'єми RDIMM (зареєстровані модулі пам'яті) та процесор. У порівнянні зі стандартним комп'ютером, серверні материнські плати зазвичай мають два процесори, кілька ядер і, як наслідок, більше потоків для обробки даних.

Центральний процесор – це мозок комп'ютера. Він відповідає за обробку вхідних даних, зберігання даних і виведення результатів.

Серверні процесори набагато потужніші, ніж процесори в звичайних ПК. Вони повинні бути здатні підтримувати складні робочі навантаження, від транзакцій з базами даних і обміну електронною поштою до маршрутизації мережевого трафіку і складних обчислювальних завдань.

Максимальна потужність будь-якого сервера визначається процесором. Чим більше ядер у вашому сервері, тим більше процесор може обчислювати кілька кластерів даних одночасно. Однак, чим більше ядер, тим, як правило, нижча продуктивність кожного з них. Тому важливо знайти правильний баланс між масштабом і продуктивністю.

Оперативна пам'ять – це короткочасна пам'ять комп'ютера. У ній тимчасово зберігаються дані з програми, запущеної на сервері. Це дозволяє центральному процесору обробляти дані швидше, ніж якби він отримував доступ до них безпосередньо.

Оперативна пам'ять має вигляд планки (відомої як DIMM), що вставляється в материнську плату. Доступні різні покоління оперативної пам'яті. Останні покоління включають DDR3, DDR4 і DDR5, і кожне з них має більш розширені функції та можливості продуктивності. Варто зазначити, що не всі покоління оперативної пам'яті сумісні з кожною материнською платою, наприклад, DDR3 не працюватиме з материнською платою, яка підтримує DDR4.

Існує два типи оперативної пам'яті:

– Оперативна пам'ять з кодом корекції помилок (ECC-пам'ять);

– Оперативна пам'ять без коду корекції помилок (не ECC-пам'ять).

Більшість серверів використовують ECC-пам'ять, бо вона поставляється з додатковим чіпом пам'яті, який допомагає запобігти пошкодженню даних, автоматично виявляючи та виправляючи помилки пам'яті. І навпаки, пам'ять без коду корекції помилок зазвичай використовується в ноутбуках і настільних комп'ютерах споживчого класу [16].

Блок живлення сервера не подає живлення безпосередньо на сервер, а отримує його з електричної розетки і регулює потужність різними способами, залежно від моделі сервера.

Блок живлення критично важливий для функціонування сервера, тому його резервування є важливим фактором. Сервер може мати один або декілька БЖ. Один ДБЖ здатен забезпечити достатню потужність за нормальних умов, але у випадку виходу його з ладу сервер залишиться без електроживлення. Додавання додаткових модулів БЖ забезпечує надмірність. Якщо один модуль вийде з ладу, другий (або третій) буде в режимі очікування, щоб замінити його.

Графічний процесор – це тип процесора, який прискорює рендеринг графіки та відео, обробляючи багато даних одночасно. Графічні процесори можуть бути інтегровані в центральний процесор сервера, але також можуть бути окремим апаратним блоком.

Більшість серверів потребують принаймні однієї форми сховища з прямим підключенням.

Існує два основних типи накопичувачів:

– Жорсткі диски (HDD);

– Твердотільні накопичувачі (SDD).

Жорсткий диск – це частина серверного обладнання, що складається з декількох дисків, розташованих навколо центрального шпинделя. Жорсткий диск зберігає дані, включаючи операційну систему, файли та програми навіть за відсутності електроживлення. Жорсткі диски складаються з двох центральних елементів магнітної пластини для зберігання даних і приводу для зчитування і запису даних.

Твердотільні накопичувачі використовують енергонезалежну пам'ять, яку можна перезаписувати у стандартному інтерфейсі дискового накопичувача. Вони не мають рухомих частин, що робить їх набагато швидшими за жорсткі диски та ідеальними для робочих навантажень з інтенсивним використанням даних.

Усім серверам потрібне принаймні одне мережеве підключення. І воно створюється за допомогою мережевого адаптера.

Мережевий адаптер – це фізичний порт на материнській платі, який дозволяє серверу спілкуватися через локальну мережу (LAN), підключаючись до інших пристроїв та інтернету.

Шасі – це металевий корпус, в якому розміщується сервер. Шасі призначені для економії простору за рахунок розміщення декількох серверів в одній фізичній структурі. Тип шасі визначається типом серверного обладнання (стійка, блейд або мейнфрейм).

Що стосується ПЗ, то всі сервери матимуть щонайменше два програмні компоненти:

– Операційну систему;

– Програмний додаток.

Комп'ютерна програма виконує завдання від імені клієнта. Саме ці програми перетворюють окреме серверне обладнання на веб-сервер, сервери додатків, сервер баз даних тощо. Операційна система виступає в ролі інтерфейсу між серверним обладнанням та серверним додатком.

Сервери зберігають величезні обсяги даних і роблять ці дані доступними для користувачів через мережу. Сервер працює разом з програмою, яка називається ОС, що керує апаратними ресурсами сервера. Функціональність можна розширити за допомогою додаткового програмного забезпечення.

Разом сервер і ОС працюють над тим, щоб отримувати запити користувачів і відповідати на них, витягуючи відповідні файли з джерел даних на сервері.

Ці відносини між сервером (постачальником ресурсів) і користувачем, який запитує послуги (клієнтом), називаються клієнт-серверною моделлю. В архітектурі клієнт-сервер клієнтський комп'ютер надсилає серверу запит на отримання даних. Сервер обробляє цей запит і надає запитувані дані клієнту [16].

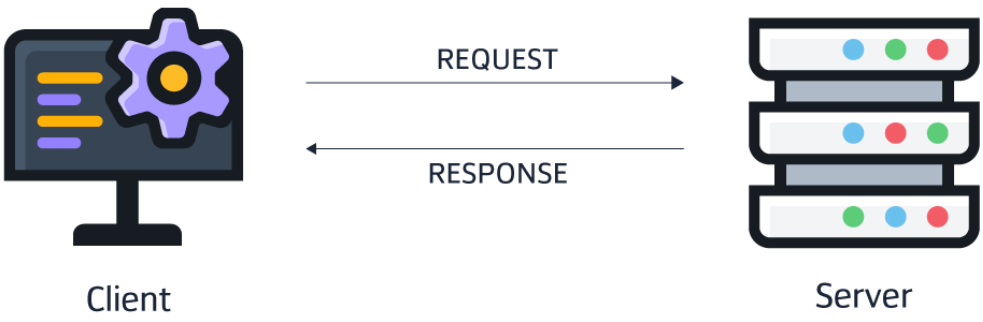


Рисунок 1.1.1 – Клієнт-серверна модель

Сервери мають різні функціональні можливості, які задовольняють окремі потреби клієнтів, уподобання щодо масштабування, обмеження простору та вимоги до управління.

Типи серверів:

– Фізичні сервери;

– Віртуальні сервери;

– Хмарні сервери.

Фізичні сервери – це фізичні машини або обладнання, які призначені для виконання різних обчислювальних завдань безпосередньо на апаратному рівні.

Розглянемо три типи фізичних серверів:

– Стоєчні сервери;

– Блейд-сервери;

– Мейнфрейм сервери.

Стоєчні сервери – це машини загального призначення, які (як випливає з назви) монтуються в серверну стійку разом з іншими стоєчними серверами та/або мережевими пристроями і пристроями зберігання даних. Стоєчні сервери мають розмір стандартного комп'ютера, тому займають мінімум місця і можуть підтримувати широкий спектр робочих навантажень. Їх легко замінювати та модернізувати за допомогою додаткових компонентів, і не обов'язково, щоб усі стоєчні сервери в одній стійці були однієї моделі. Кабелі для живлення, підключення до мережі та зберігання даних підключаються до задньої частини стека.

Блейд-сервери – це високопродуктивні сервери, щільно сконфігуровані в корпусі. Вони займають ще менше місця, ніж стоєчні сервери. Модульні компоненти (блейди) встановлюються в шасі разом з іншими блейдами.

Кожен окремий блейд – це сервер з власним процесором, пам'яттю та мережевими контролерами. Шасі забезпечує живлення, охолодження та мережу, спільну для всіх блейд-серверів у шасі, а диски можна замінювати в гарячому режимі (можна замінити без вимкнення системи). Інформацією, помилками та/або несправностями можна керувати з центральної консолі.

Мейнфрейм сервери – це великі, високопродуктивні комп'ютери. Сучасні мейнфрейми набагато компактніші, ніж старі моделі (які могли легко зайняти цілу кімнату). Але вони все ще значно громіздкіші, ніж стоєчні сервери або блейди. Мейнфрейми розраховані на потужність і високу пропускну здатність. Вони підтримують великі обсяги робочих навантажень з інтенсивним використанням даних і масштабні одночасні транзакції. Вони також мають високу конфігурацію з можливістю гарячої заміни обладнання, а загальносистемні рівні резервування серверів роблять мейнфрейми надзвичайно надійними.

Віртуальні сервери – це розділені простори, які прозоро існують у віддалених дата-центрах або хмарних середовищах. Оскільки вони розташовані всередині фізичних серверів та відтворюють функціональність виділених фізичних серверів, які можуть бути спільними для кількох користувачів у різних місцях, кожен з яких має різні рівні контролю над сервером [17].

Створення віртуального сервера починається з гіпервізора, програмного забезпечення для віртуалізації, яке ізолює (або абстрагує) операційні системи від основного комп'ютерного обладнання. Процес віртуалізації дозволяє апаратному забезпеченню хост-машини керувати гостьовими віртуальними машинами, які спільно використовують ресурси фізичного сервера, включаючи простір пам'яті та пропускну здатність мережі.

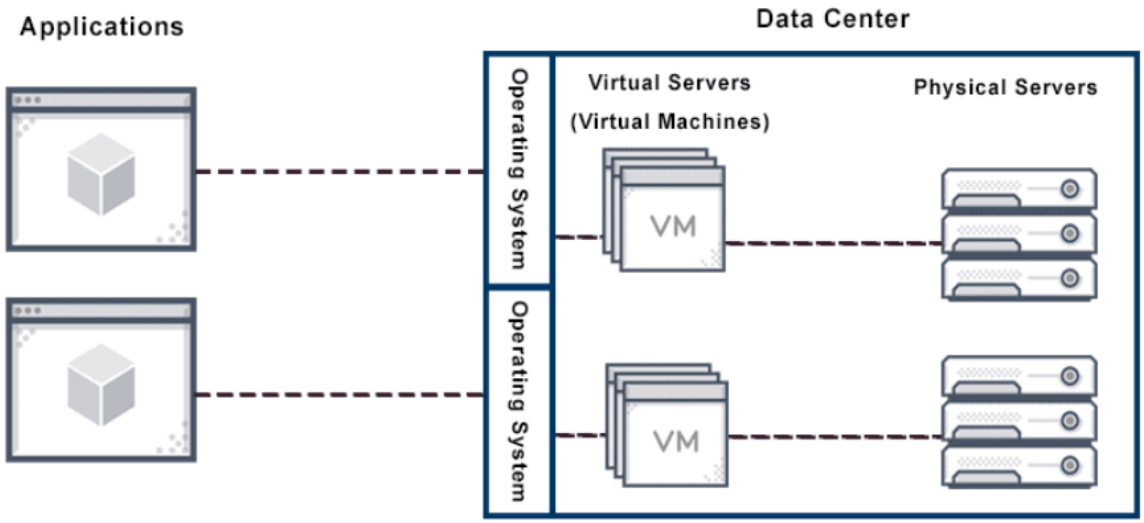


Рисунок 1.1.2 – Принцип роботи віртуальних серверів

Віртуалізація фізичного сервера забезпечує користувачам безпечний доступ до даних, дозволяючи їм швидше використовувати його потужність, підвищуючи ефективність та швидкодію. Віртуалізація серверів також централізує управління серверами, усуває надмірне резервування та зберігає IT-ресурси.

Розглянемо три типи віртуалізації серверів.

– Повна віртуалізація;

– Паравіртуалізація;

– Контейнеризація (віртуалізація на рівні ОС).

Повна віртуалізація, запроваджена у 1966 році компанією IBM, використовує двійкову трансляцію та методи прямого підходу, щоб повністю ізолювати гостьову операційну систему від рівня віртуалізації та апаратного забезпечення комп'ютера.

У цьому середовищі будь-яка ОС, що підтримується базовим обладнанням, може працювати в кожній окремій віртуальній машині, і будь-яке ПЗ, яке може виконуватися на фізичному обладнанні, також може працювати у віртуальній машині.

Паравіртуалізація – це технологія, яка дозволяє віртуальним машинам мати інтерфейс, подібний до інтерфейсу основного обладнання. Для цього гостьова ОС модифікується для реалізації API, який дозволяє їй безпосередньо спілкуватися з гіпервізором. Ця модифікація дозволяє ОС швидко виконувати трудомісткі завдання за допомогою команд (гіпервикликів).

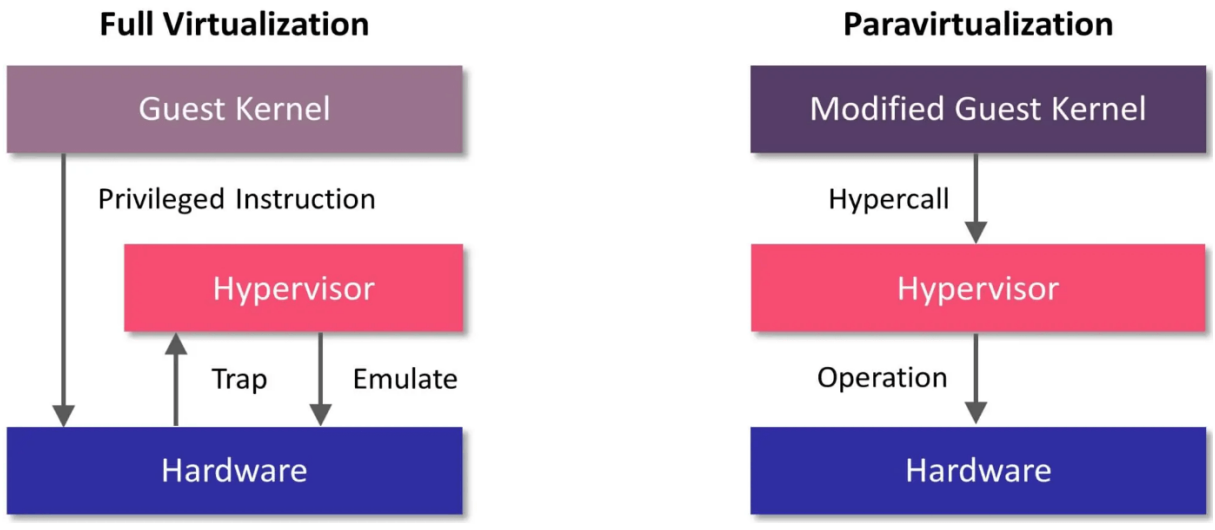


Рисунок 1.1.3 – Принцип роботи повної віртуалізації та паравіртуалізації

Контейнеризація – це технологія, яка розділяє операційну систему для створення декількох ізольованих віртуальних машин (за допомогою програмного забезпечення для контейнеризації, такого як Docker або Kubernetes). Таким чином, ОС може запускати різні програми на одному комп'ютері одночасно, не заважаючи один одному.

Віртуалізація на рівні ОС підвищує доступність, керованість та безпеку складних програмних середовищ, вимагаючи мінімальних змін в існуючій обчислювальній інфраструктурі та незначних витрат ресурсів.

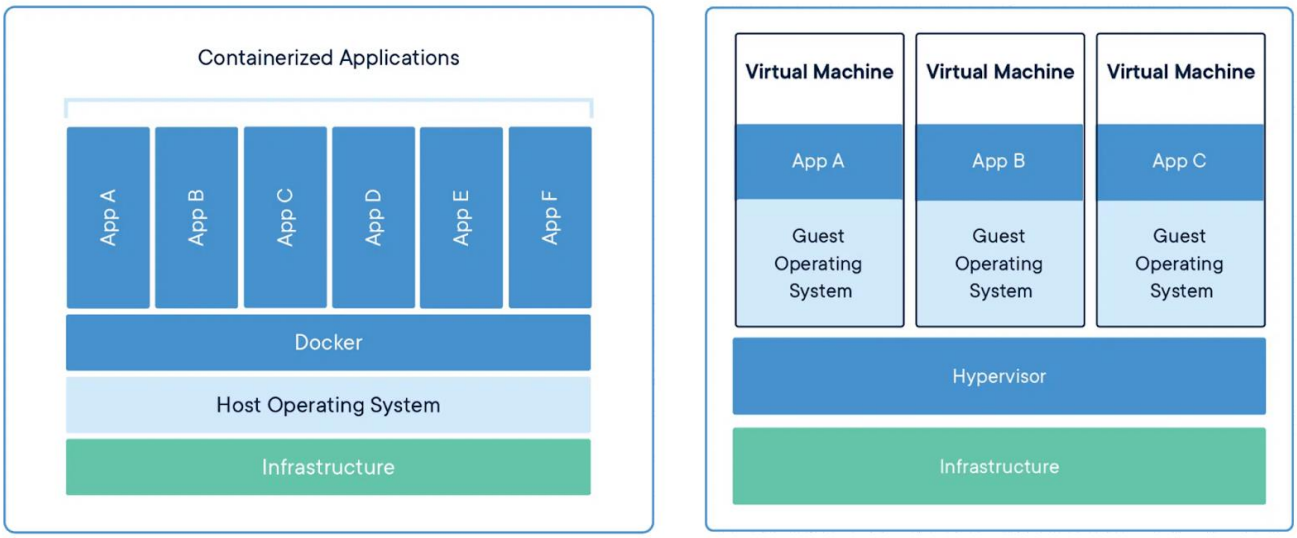


Рисунок 1.1.4 – Принцип роботи контейнеризації

Хмарні сервери – це об'єднані, централізовані серверні ресурси, які розміщуються та надаються через мережу (зазвичай Інтернет). До них на вимогу можуть звертатися різні користувачі. Хмарні сервери можуть виконувати всі ті ж функції, що й традиційні фізичні сервери, надаючи обчислювальну потужність, сховище та додатки [12].

Хмарні сервери можуть бути розташовані в будь-якій точці світу і надавати послуги віддалено через хмарне обчислювальне середовище.

Хмарні сервери стали можливими завдяки віртуалізації. ПЗ для управління, яке називається гіпервізором, встановлюється на фізичних серверах для їх підключення та віртуалізації, абстрагування їх спільних ресурсів та об'єднання їх разом для створення віртуальних серверів. Ці віртуальні ресурси можуть бути автоматизовані та доставлені через хмару для спільного використання.

Цей підхід відомий як модель «інфраструктура як послуга» (IaaS). При використані IaaS, не потрібно володіти та керувати власним обладнанням, його можна отримати від третіх сторін, які надають ресурси на вимогу через публічну хмару. Поширеним прикладом хмарного сервера є використання публічної хмари для тимчасових, сезонних або змінних робочих навантажень, які необхідно швидко масштабувати в міру необхідності [12].

У деяких випадках хмарні сервери також можуть бути налаштовані хмарним провайдером як виділені сервери. У цьому випадку, їх іноді називають «голими» серверами, провайдер виділяє фізичні хмарні сервери одному клієнту, який може мати специфічні вимоги до продуктивності або сховища даних.

Хмарні сервери можуть бути розгорнуті в трьох основних типах хмар:

– Публічні хмари;

– Приватні хмари;

– Гібридні хмари.

Публічні хмари. Хмарні сервери загалом розгортаються в публічних хмарах. У цьому випадку сторонній провайдер володіє та керує серверами, а також надає своїм клієнтам доступ до обчислювальних послуг на вимогу.

Приватні хмари. Можна розміщувати власні хмарні сервери в приватному секторі та зберігати контроль над їхнім управлінням і обслуговуванням. Ці серверні ресурси не використовуються спільно з іншими, але оскільки вони знаходяться в хмарі, до них може отримати віддалений доступ будь-який клієнт через VPN.

Гібридні хмари. Публічні та приватні хмари можуть поєднуватися з локальними хмарними серверами та хмарними серверами, що працюють поза межами підприємства. Таке гібридне хмарне середовище дає більше можливостей і гнучкості для забезпечення контролю та безпеки, коли це необхідно.

1.2 Огляд та аналіз гіпервізорів

Гіпервізор – це програма, яка контролює фізичні ресурси обчислювальної машини і розподіляє ці ресурси між кількома різними операційними системами, що дозволяє їм працювати одночасно [15].

Іншими словами, гіпервізор створює копії або клони апаратних ресурсів одного фізичного комп'ютера.

Кожен клон представляється як окремий пристрій. Користувачі можуть встановлювати гостьові операційні системи на кожну з віртуальних машин без прив'язки до основного обладнання.

Гіпервізор ізолює операційні системи одна від одної таким чином, щоб кожна з них отримувала ексклюзивне використання виділених для неї ресурсів. За потреби гіпервізор може дозволити операційним системам окремих віртуальних машин взаємодіяти.

Таке з'єднання можна використовувати для спільного доступу до певних файлів або обміну даними через локальну мережу.

Існує два основних типи гіпервізорів, які широко використовуються:

– Гіпервізори типу 1;

– Гіпервізори типу 2.

Гіпервізори типу 1 також називають мікроядрами, тонкими гіпервізорами або автономними гіпервізорами, що працюють на «голому металі». Гіпервізори 1-го типу найкраще уявляти собі як компактну операційну систему зі специфічними функціями, яка встановлюється безпосередньо на «голий» сервер і має основні ознаки ОС:

– Представляє абстрактний набір ресурсів для додатків замість невпорядкованого набору апаратного забезпечення;

– Керує пулом ресурсів, розподіляючи процесорний час, пам'ять та пристрої вводу/виводу між програмами, які запитують використання ресурсів комп'ютера, забезпечуючи ефективну роботу.

Гіпервізори типу 1 надає гостьовим операційним системам, що запускаються під його управлінням на верхньому рівні, абстракцію віртуальної машини.

В результаті кожна гостьова ОС отримує ілюзію повного контролю над усіма ресурсами внизу. Вона функціонує так само, як операційна система, що працює на реальному обладнанні у привілейованому режимі, як супервізор, надаючи їй доступ до всіх системних функцій і ресурсів, необхідних для її нормальної роботи.

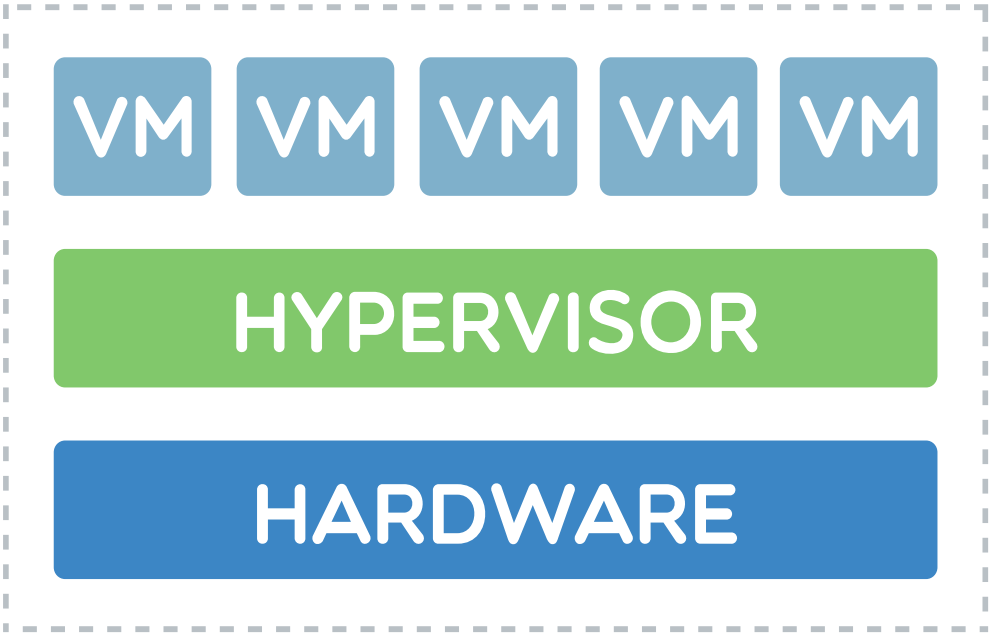


Рисунок 1.2.1 – Принцип роботи гіпервізора типу 1

Супервізор – це центральний модуль керування, ядро операційної системи. Він може складатися з декількох частин: диспетчера програм, диспетчера завдань, диспетчера вводу/виводу та інших [14].

Більшість сучасних процесорів Intel і AMD для серверів і настільних комп'ютерів підтримують віртуалізацію на апаратному рівні. Операційна система може використовуватися у двох режимах: привілейованому та користувацькому. Права доступу програми, яка керує ресурсами комп'ютера, суворо обмежені.

Гіпервізори 1-го типу мають дуже важливу якість, розмір їхніх кодів у сотні разів менший, ніж у більшості сучасних операційних систем. Це також призводить до меншої кількості помилок, які можуть призвести до зависання всієї системи. Помилка в одній з операційних систем на одній з віртуальних машин користувача не повинна впливати на інші системи на тому ж фізичному обладнанні.

Безпека є однією з основних вимог до гіпервізорів, оскільки вони отримують повний контроль над апаратними ресурсами, на яких виконується віртуалізація.

Отже, завданням гіпервізора є безпечне виконання машинного коду, не дозволяючи при цьому гостьовій ОС:

– Блокувати переривання;

– Модифікувати таблиці сторінок віртуальної пам'яті у фізичну пам'ять всієї машини;

– Змінювати дані в комірках пам'яті, призначених для інших запущених процесів (за винятком випадків, коли цього вимагає завдання). Системні виклики також перехоплюються і виконуються всередині гіпервізора, але кожна гостьова ОС виконує код у власному привілейованому режимі. Іншими словами, гіпервізор створює ілюзію, що код гостьової ОС виконується на апаратному рівні, у привілейованому режимі, тоді як насправді він виконується у гостьовому режимі. Якщо одна з гостьових систем вийде з ладу, інші продовжать працювати.

Гіпервізор виявляється єдиним програмним забезпеченням, що працює в режимі повних привілеїв. Ця якість гіпервізора називається еквівалентністю, поведінка користувацьких програм не змінюється при роботі на віртуальній машині або на фізичному обладнанні, за винятком часових характеристик.

Істотно відрізняється час виконання коду, гіпервізор забирає частину процесорного часу на перехоплення та аналіз інструкцій гостьової операційної системи, а також на емуляцію виконання. Крім того, ресурси фізичного обладнання зазвичай розподіляються між кількома віртуальними машинами, і кожна з них отримує лише частину процесорного часу на вимогу. Однак цього достатньо для безперебійного запуску більшості процесів, адже не всі вони використовуються з однаковою інтенсивністю. Деякі процеси можуть перебувати в режимі очікування на введення даних користувачем або на вимкнення повільного периферійного обладнання. Цей час використовується ефективно, оскільки система розподіляє його між іншими активними процесами в багатозадачному режимі.

Найпопулярніші гіпервізори типу 1:

– Xen;

– VMware ESXi;

– Hyper-V.

Тонкий гіпервізор Xen був розроблений в рамках дослідницького проекту лабораторії Кембриджського університету. Перший публічний реліз відбувся у 2003 році, коли компанія XenSource випустила гіпервізор на ринок. У 2007 році Citrix поглинула XenSource, в результаті чого деякі продукти були перейменовані. Під керівництвом Citrix гіпервізор Xen отримав значний комерційний успіх.

Xen – це кросплатформенний гіпервізор, який підтримує апаратну віртуалізацію та паравіртуалізацію. Більшість компонентів розташовані за межами гіпервізора, тому Xen містить мінімальну кількість коду. Xen є повністю відкритим гіпервізором з ліцензією GNU GPL 2 з відкритим вихідним кодом. Це означає, що продукт можна модифікувати так, як потрібно користувачеві. Дехто вважає Xen гібридним гіпервізором через його підтримку паравіртуалізації та апаратної віртуалізації.

Автономний гіпервізор VMware ESXi – це рішення для віртуалізації класу Enterprise, розроблене компанією VMware. Як і інші продукти VMware, ESXi доступний безкоштовно з обмеженою функціональністю. Платна повна версія підтримує такі функції, як централізований контроль над усіма віртуальними машинами на кожному хості проекту за допомогою платформи vCenter. Втім, навіть безкоштовна версія справляється з усіма основними завданнями гіпервізора. За відгуками користувачів, продукт відрізняється високою стабільністю, простотою в управлінні, використовує мінімальну кількість коду і підтримує широкий спектр гостьових систем, широко використовуваних в корпоративному секторі.

Hyper-V – це рішення апаратної віртуалізації для систем x64 від компанії Microsoft. Доступне у двох варіантах: як роль у серверних ОС сімейства Windows (Windows Server 2008, Windows Server 2012 та ін.), а також x64 Pro та Enterprise версіях Windows 8, Windows 8.1, Windows 10, і як окремий продукт сервер Microsoft Hyper-V. Багато користувачів, які звикли до рішень Microsoft, вважають Hyper-V одним з найзручніших гіпервізорів, коли мова йде про віртуалізацію. Хмара Azure повністю побудована на нативних продуктах Microsoft.

Гіпервізори типу 2 також називають розміщеними гіпервізорами. Цей тип гіпервізорів є додатковим програмним шаром, який знаходиться над основною ОС.

Ці гіпервізори працюють як один з процесів, що виконуються основною ОС, найчастіше Linux. У цьому випадку права гіпервізора значно обмежені. Він керує гостьовими операційними системами, в той час як емуляцією та контролем фізичних ресурсів займається хостова ОС.

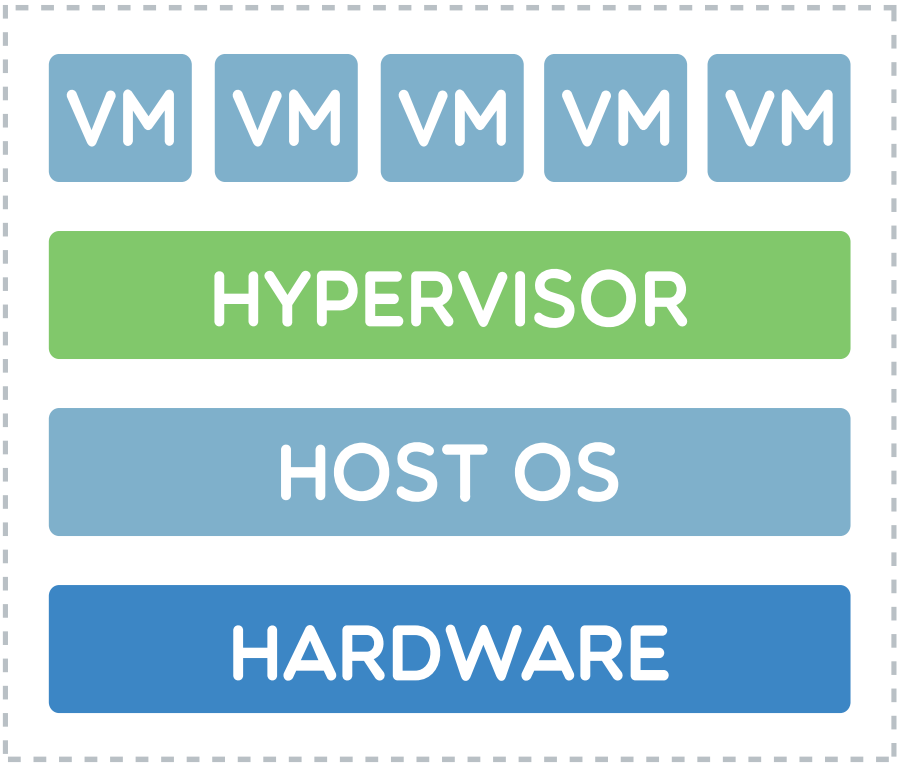


Рисунок 1.2.2 – Принцип роботи гіпервізора типу 2

Найпопулярніші гіпервізори типу 2:

– Oracle VM VirtualBox;

– VMware Workstation;

– KVM.

Oracle VM VirtualBox – це модульний кросплатформний гіпервізор для Linux, macOS, Microsoft Windows, FreeBSD, Solaris/OpenSolaris, ReactOS, DOS та інших систем Oracle. Гіпервізор був створений у 2007 році компанією Sun Microsystem, після поглинання компанії Oracle робота над гіпервізором продовжилася. Вихідний код базової версії був відкритий за ліцензією GNU GPL, гіпервізор популярний завдяки необмеженим можливостям модифікації. VirtualBox може підтримувати x64 гостьові операційні системи, навіть якщо хостова ОС 32-розрядна.

Перша версія гіпервізора VMware Workstation, розроблена компанією VMware, була випущена в 1999 році. Це власне програмне забезпечення працює з хостовими операційними системами x86 і x64, такими як Microsoft Windows, Linux, Ubuntu, CentOS. Він підтримує понад 200 гостьових операційних систем. Для тестування програмного забезпечення можна використовувати безкоштовну версію Workstation Player.

Kernel-based Virtual Machine, KVM – це гіпервізор, створений у жовтні 2006 року. Він був швидко інтегрований в основну гілку ядра Linux 2.6.20, випущену на початку 2007 року. Пізніше KVM було адаптовано як модуль ядра у FreeBSD. До складу KVM входить завантажуваний модуль ядра kvm.ko, який відповідає за віртуалізацію, завантажуваний модуль для процесорів AMD або Intel - kvm-amd.kp або kvm-intel.ko, а також компоненти користувацького режиму QEMU. KVM має повністю відкритий вихідний код за ліцензіями GNU GPL та GNU LGPL. KVM QEMU було використано для створення загальнодоступної хмари SIM-Cloud від SIM-Networks.

Гібридні гіпервізори поєднують в собі деякі характеристики гіпервізорів типу 1 і типу 2 (поєднання тонкого гіпервізора і спеціалізованої сервісної ОС, що функціонує на апаратному рівні під його управлінням). Гіпервізор безпосередньо керує процесором і пам'яттю, в той час як сервісна ОС надає гостям доступ до пристроїв вводу/виводу [14].

Ця технологія активно розвивається, постійно створюються більш гнучкі та інтегровані версії. Останнім часом Xen і Hyper-V класифікують як гібридні гіпервізори, а не як гіпервізори типу 1, що частково відповідає дійсності. Сучасні версії цих гіпервізорів поєднують в собі значну кількість характеристик обох типів.

1.3 Поняття Penetration Testing

Тестування на проникнення – це імітація кібератаки на комп'ютерну систему для перевірки наявності вразливостей, які можна використати [1].

Тестувальники використовують ті ж інструменти, методи і процеси, що і зловмисники, щоб знайти і продемонструвати вплив слабких місць в системі на бізнес.

Тести на проникнення зазвичай імітують різноманітні атаки, які можуть загрожувати бізнесу. Вони можуть перевірити, чи достатньо надійна система, щоб протистояти атакам з авторизованих і неавторизованих позицій, а також з різних системних ролей. При правильному підході, пентест може зануритися в будь-який аспект системи [19].

Усі тести на проникнення передбачають імітацію кібератаки на комп'ютерні системи. Однак різні типи тестів націлені на різні типи активів підприємства:

– Тестування додатків;

– Тестування мережі;

– Апаратне тестування.

Тестування додатків шукає вразливості в додатках і пов'язаних з ними системах, включаючи веб-додатки, веб-сайти, мобільні додатки, додатки Інтернету речей, хмарні додатки та інтерфейси прикладного програмування (API).

Пентестери часто починають з пошуку вразливостей, перелічених у Топ-10 проекту Open Web Application Security Project (OWASP).

OWASP Top 10 – це список найбільш критичних вразливостей у веб-додатках. Список періодично оновлюється, щоб відображати мінливий ландшафт кібербезпеки, але найпоширеніші вразливості включають ін'єкції шкідливого коду, неправильні конфігурації та збої автентифікації. Окрім OWASP Top 10 також виконуються ручні тести додатків, пошук менш поширених недоліків безпеки та вразливості, які можуть бути унікальними для конкретного додатку [20].

Під час тестування мережі атакують всю комп'ютерну мережу компанії. Існує два основних типи мережевих тестів:

– Зовнішні;

– Внутрішні.

У зовнішніх тестах імітується поведінка зовнішніх хакерів, щоб знайти проблеми з безпекою в інтернет-активах, таких як сервери, маршрутизатори, веб-сайти та комп'ютери співробітників.

У внутрішніх тестах пентестери імітують поведінку зловмисників або хакерів з викраденими обліковими даними.

Апаратне тестування шукає вразливості в пристроях, підключених до мережі, таких як ноутбуки, мобільні пристрої, пристрої Інтернету речей тощо.

Пентестери можуть шукати недоліки програмного забезпечення, наприклад, вразливість операційної системи, яка дозволяє хакерам отримати віддалений доступ до кінцевої точки.

Вони можуть шукати фізичні вразливості, наприклад, неналежно захищений центр обробки даних, в який можна проникнути. Команда тестувальників також може оцінити, як хакери можуть переміститися зі скомпрометованого пристрою в інші частини мережі.

Перед початком тестування команда тестувальників і компанія визначають обсяг тесту. Обсяг визначає, які системи будуть протестовані, час проведення тестування, методи та скільки інформації тестувальники матимуть заздалегідь:

– Під час тестування «чорної скриньки» (black-box) пентестери не мають жодної інформації про цільову систему. Вони повинні покладатися на власні дослідження, щоб розробити план атаки;

– Під час тестування «білої скриньки» (white-box) пентестери мають повну прозорість щодо цільової системи. Компанія ділиться такими деталями, як мережеві схеми, вихідні коди, облікові дані тощо;

– Під час тестування «сірої скриньки» (gray-box) пентестери отримують певну інформацію, але небагато. Наприклад, компанія може поділитися діапазонами IP-адрес мережевих пристроїв, але тестувальники повинні самостійно дослідити ці діапазони на наявність вразливостей.

Після того, як діапазон визначено, починається тестування. Пентестери можуть використовувати декілька методологій тестування на проникнення.

Найпоширеніші з них включають:

– Керівні принципи тестування безпеки додатків OWASP;

– Стандарт виконання тестування на проникнення (PTES);

– Національний інститут стандартів і технологій (NIST) SP 800-115.

Незалежно від того, яку методологію використовує команда тестувальників, процес зазвичай складається з одних і тих самих етапів [5]:

– Розвідка;

– Виявлення та розробка цілей;

– Отримання доступу;

– Підтримання доступу;

– Прибирання та звітність.

Розвідка. Команда тестувальників збирає інформацію про цільову систему. Залежно від цілі, пентестери використовують різні методи розвідки. Наприклад, якщо об'єктом тестування є додаток, пентестери можуть вивчати його вихідний код. Якщо метою є ціла мережа, пентестери можуть використовувати аналізатор пакетів для перевірки потоків мережевого трафіку.

Пентестери також часто використовують розвідувальні дані з відкритих джерел (OSINT). Читаючи публічну документацію, новини, акаунти співробітників у соціальних мережах і на GitHub, пентестери можуть отримати цінну інформацію про свої цілі.

Виявлення та розробка цілей. Пентестери використовують інформацію, отриману на етапі розвідки, для виявлення вразливостей в системі, які можна використати. Наприклад, пентестери можуть використовувати сканер портів Nmap, для пошуку відкритих портів, через які можна відправити шкідливе програмне забезпечення. Для тестування на соціальну інженерію команда тестувальників може розробити фейкову історію, яку вони використають у фішинговому електронному листі для викрадення облікових даних співробітників.

На цьому етапі тестувальники можуть перевірити, як функції безпеки реагують на вторгнення. Наприклад, вони можуть надіслати підозрілий трафік на

брандмауер компанії, щоб подивитися, що станеться. Тестувальники будуть використовувати отриману інформацію, щоб уникнути виявлення.

Отримання доступу. Тестувальники можуть спробувати різноманітні атаки в залежності від цільової системи, вразливостей, які вони знайшли та обсягу тесту. Деякі з найбільш поширених атак включають в себе:

– SQL-ін'єкції (пентестери намагаються змусити веб-сторінку або додаток розкрити конфіденційні дані, вводячи ін'єкції у поля введення);

– Міжсайтовий скриптинг (пентестери намагаються впровадити шкідливий код на веб-сайт);

– Атаки на відмову в обслуговуванні, відомі також як DDoS атаки (пентестери намагаються вивести з ладу сервери, додатки та інші мережеві ресурси, переповнюючи їх трафіком);

– Соціальна інженерія (пентестери використовують фішинг, приманки, вигадки та інші тактики, щоб обманом змусити співробітників порушити мережеву безпеку);

– Атаки грубої сили (пентестери намагаються зламати систему, запускаючи скрипти, які генерують і тестують потенційні паролі, поки один з них не спрацює);

– Атаки «людина посередині» (пентестери перехоплюють трафік між двома пристроями або користувачами, щоб викрасти конфіденційну інформацію або встановити шкідливе програмне забезпечення).

Підтримання доступу. Після того, як пентестери скористалися вразливістю, щоб закріпитися в системі, вони намагаються пересуватися і отримати доступ до ще більшої її частини. Цей етап називають «ланцюжком вразливостей», оскільки тестувальники переходять від вразливості до вразливості, щоб проникнути в мережу глибше. Наприклад, вони можуть почати з процесу встановлення кейлоггера на комп'ютер співробітника. Використовуючи цей кейлоггер, вони можуть перехопити облікові дані працівника та отримати доступ до конфіденційної бази даних.

На цьому етапі метою тестувальника є збереження доступу та підвищення своїх привілеїв в обхід заходів безпеки. Пен-тестери роблять все це, щоб імітувати

сучасні постійні загрози (APT), які можуть ховатися в системі тижнями, місяцями або роками, перш ніж їх викриють.

Прибирання та звітність. Наприкінці симульованої атаки тестувальники прибирають усі сліди, які вони залишили після себе, наприклад, троянські програми, які вони встановили або конфігурації, які вони змінили. Це робиться для того, щоб система повернулася до нормального режиму роботи.

Потім тестувальники готують звіт про атаку. У звіті, як правило, описуються знайдені вразливості, використані експлойти, деталі того, як вони обходили засоби захисту, а також опис того, що вони робили, перебуваючи всередині системи. Звіт також може містити конкретні рекомендації щодо усунення вразливостей. Внутрішня команда безпеки може використовувати цю інформацію для посилення захисту від реальних атак [11].

Пентестери використовують різні інструменти для проведення розвідки, виявлення вразливостей та автоматизації ключових частин процесу тестування на проникнення. Деякі з найпоширеніших інструментів включають:

– Спеціалізовані операційні системи;
– Інструменти для злому облікових даних;
– Сканери портів;
– Сканери вразливостей;
– Аналізатори пакетів;
– Metasploit.

Спеціалізовані операційні системи. Більшість пентестерів використовують операційні системи, призначені для тестування на проникнення та етичного злому. Найпопулярнішим є Kali Linux, дистрибутив Linux з відкритим вихідним кодом, який постачається з попередньо встановленими інструментами.

Інструменти для злому облікових даних. Це програми, які можуть розкривати паролі, запускаючи атаки грубої сили, які використовують ботів або скрипти для автоматичної генерації та тестування потенційних паролів, поки один з них не буде підібраний. Прикладами таких програм є Medusa, Hyrda, Hashcat, John the Ripper.

Сканери портів. Це програми, які дозволяють тестувальникам віддалено тестувати пристрої на наявність відкритих і доступних портів, які вони можуть використовувати для злому мережі. Прикладами таких програм є Nmap, masscan, Zmap.

Сканери вразливостей. Інструменти сканування вразливостей шукають в системах відомі вразливості, що дозволяє тестувальникам швидко знаходити потенційні входи в ціль. Приклади включають Nessus, Core Impact та Netsparker.

Веб-сканери вразливостей – це підгрупа сканерів вразливостей, які оцінюють веб-додатки та веб-сайти. Приклади включають Burp Suite і Zed Attack Proxy (ZAP) від OWASP.

Аналізатори пакетів. Це програми, які також називають сніфферами пакетів, вони дозволяють тестувальникам аналізувати мережевий трафік, перехоплюючи та перевіряючи пакети.

Пентестери можуть з'ясувати, звідки надходить трафік, куди він прямує і в деяких випадках, які дані в ньому містяться. Wireshark і tcpdump є одними з найбільш часто використовуваних аналізаторів пакетів.

Metasploit – це фреймворк для тестування на проникнення. Найголовніше, що Metasploit дозволяє тестувальникам автоматизувати кібератаки. Metasploit має вбудовану бібліотеку попередньо написаних кодів експлойтів та корисного навантаження. Тестувальники можуть вибрати експлойт, надати йому корисне навантаження для доставки в цільову систему і дозволити Metasploit впоратися з рештою.

1.4 Концепція та типи CTF

CTF – це змагання в сфері кібербезпеки, де команди або окремі учасники вирішують завдання, пов'язані з безпекою комп'ютерних систем, для того щоб знайти прапор (flag) заздалегідь визначений рядок символів, який слугує доказом вирішення завдання.

CTF-змагання стали популярним інструментом для навчання, тренувань та перевірки навичок у сфері інформаційної безпеки.

Типи CTF:

– Jeopardy;

– Attack/Defense (Атака/оборона);

– King Of the Hill (Король гори);

– Linear (Лінійний);

– Mixed (Змішаний).

Jeopardy. Найпоширеніший тип CTF який нагадує телевізійну гру Jeopardy, учасники отримують набір завдань згрупованих за категоріями.

За кожне завдання можна отримати бали, для того щоб вирішити завдання та отримати певну кількість балів, потрібно знайти та ввести прапор. Здебільшого завдання розділені за рівнями складності, у переважній більшості випадків чим складніше завдання, тим більше балів можна отримати за його виконання, учасники можуть самостійно обирати завдання на яких вони хочуть зосередитися.

Категорії завдань в Jeopardy можуть включати:

– Cryptography (Криптографія). Завдання, які пов'язані з розшифруванням зашифрованих повідомлень, аналізом криптографічних алгоритмів та інших аспектів криптографії;

– Forensics (Криміналістика). Завдання, що передбачають аналіз цифрових слідів, відновлення даних, аналіз пам'яті, мережевий аналіз тощо;

– Web Exploitation (Веб-експлуатація). Завдання, що включають пошук вразливостей на веб-сайтах та їх експлуатацію;

– Reverse Engineering (Зворотна розробка). Завдання, де учасники мають аналізувати виконувані файли або коди для розуміння їх функціональності та пошуку прихованої інформації;

– Binary Exploitation (Бінарна експлуатація). Завдання, що передбачають знаходження та експлуатацію вразливостей у виконуваних файлах чи системних додатках.

Attack/Defense (Атака/оборона). Цей тип CTF поділяється червону та синю команди, кожна з яких має власний сервер або мережеву інфраструктуру з вразливостями та прихованими прапорами.

Синя команда виконує лише оборонну роль, вони захищають системи або сервери з прихованим прапором і виправляють будь-які вразливості, які вони знаходять на місці. Ця робота повинна бути виконана вчасно, щоб інша команда не змогла атакувати або використати їхній ослаблений сервер. Також вони не можуть виключити свої атаки за допомогою брандмауера, тому що це суперечить правилам, їм доведеться вручну знаходити або сканувати вразливості і працювати над рішенням, щоб запобігти будь-якій атаці [9].

Червона команда виконує наступальну роль, щоб атакувати іншу команду і захопити їхні прапори, вони можуть потрапити в системи інших команд, і їхня єдина мета порушити або експлуатувати сервери.

Фіолетові команди призначені для покращення обміну інформацією між червоними та синіми командами, щоб максимізувати їхню індивідуальну та спільну ефективність.

King Of the Hill (Король гори). Цей тип CTF є найцікавішим, оскільки в ньому беруть участь декілька команд, які одночасно змагаються на декількох вразливих серверах. Команда, яка зможе першою завоює центральний сервер, нагороджується балами, а її завдання полягає в тому, щоб ніхто не отримав сервер від них, тому вони спочатку атакують, а потім захищають один і той же сервер. На змагання дається час, і команда, яка утримує доступ до центрального сервера до закінчення часу, отримує більше балів.

Спеціальний закодований механізм перевіряє кожен сервер через певні проміжки часу для нарахування балів командам.

King of the Hill зазвичай мають завдання «чорної скриньки» (black-box), коли учасники не мають жодних знань чи інформації про систему, яку вони намагаються зламати, а потім захистити [7].

Linear (Лінійний). Цей тип CTF фокусується на певній категорії і представляє серію завдань з головоломками та підказками. У лінійному CTF питання мають бути вирішені в обов'язковому порядку, який є обов'язковим.

Це схоже на гру-головоломку та ідеальний приклад карти скарбів, де послідовність має значення, одне питання веде до іншого.

Mixed (Змішаний). Цей тип CTF включає в себе Jeopardy та Атаку/Оборону, де команди виконують окремі завдання і водночас можуть атакувати іншу команду, захищати свої власні машини.

1.5 Огляд наявних рішень для розгортання платформ у форматі CTF

Коли мова йде про вибір платформ на яких можна проводити навчання за певним напрямком у форматі CTF, кожна платформа матиме власні критерії, такі як складність процесу розгортання, налаштування, зручність інтерфейсу, наявність навчальних матеріалів, підтримка різних форматів завдань тощо. Тому розглянемо популярні рішення для розгортання платформ у форматі CTF:

– Facebook CTF;

– CTFd;

– Root the Box.

Facebook CTF – це платформа, яка розроблена компанією Facebook для проведення змагань та навчань з кібербезпеки у форматі CTF.

Основні характеристики Facebook CTF:

– Платформа має відкритий код, що дозволяє будь-кому розгорнути її на власному сервері та налаштувати, модифікувати відповідно до своїх потреб;

– Інтерфейс платформи виконано в цікаво стилі, він максимально інтуїтивно зрозумілий для користувача. Всі завдання та прогрес відображаються в одному місці, що дуже сильно полегшує навігацію;

– Платформа підтримує різноманітні типи завдань, включаючи Web Exploitation, Binary Exploitation, Reverse Engineering, Forensics, Crypto та інші. Це дозволяє учасникам розвивати свої навички у різних напрямках кібербезпеки;

– Facebook CTF підтримує командні змагання, що дозволяє учасникам працювати разом над вирішенням завдань, обмінюватися ідеями та знаннями;

– Для підвищення конкурентного духу платформа має рейтингові таблиці, де відображаються результати команд та учасників у реальному часі;

– Платформа забезпечується докладною документацією, що допомагає організаторам налаштувати та запустити змагання.



Рисунок 1.5.1 – Інтерфейс Facebook CTF

CTFd – це платформа для проведення змагань та навчань з кібербезпеки у форматі CTF.

Основні характеристики CTFd:

– Платформа має відкритий код, що дозволяє будь-кому розгорнути її на власному сервері та налаштувати, модифікувати відповідно до своїх потреб;

– CTFd має інтуїтивно зрозумілий інтерфейс, що полегшує створення завдань. Інтерфейс призначений як для адміністраторів так і для учасників;

– Платформа підтримує різноманітні типи завдань, такі як Web Exploitation, Binary Exploitation, Reverse Engineering, Forensics, Crypto та інші. Підтримує виклики з можливістю розблокування, підказки, що розблоковуються, обмеження спроб для викликів тощо;

– CTFd дозволяє проводити командні та індивідуальні змагання, що забезпечує гнучкість у форматах проведення;

– Платформа автоматично оновлює рейтингові таблиці в реальному часі, відображаючи результати команд та учасників.

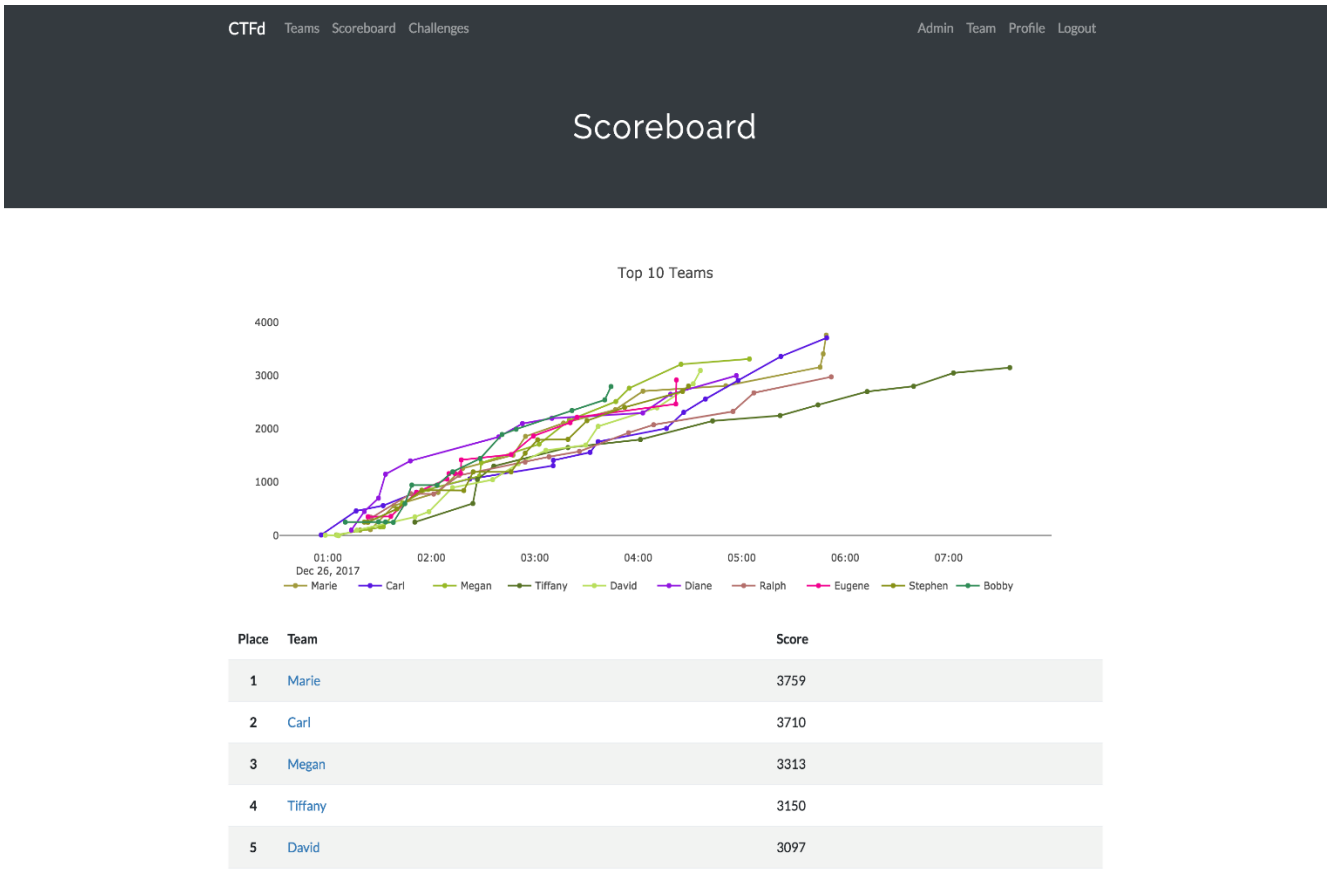


Рисунок 1.5.2 – Інтерфейс CTFd

Root The Box – це платформа для проведення змагань та навчань з кібербезпеки у форматі CTF, яка орієнтована на забезпечення комплексного і гнучкого середовища.

Основні характеристики Root The Box:

– Платформа має відкритий код, що дозволяє користувачам безкоштовно завантажувати, використовувати і модифікувати його відповідно до своїх потреб;

– Платформа має інтуїтивно зрозумілий інтерфейс, який підтримує різні функції для адміністраторів та учасників змагань. Усі основні дії легко доступні через веб-інтерфейс;

– Платформа підтримує різноманітні типи завдань, включаючи Web Exploitation, Binary Exploitation, Reverse Engineering, Forensics, Crypto та інші. Завдання можуть бути різної складності та формату;

– Root The Box має вбудовану систему економіки гри, де учасники можуть заробляти і витрачати бали на підказки, купівлю прапорів та інші ігрові елементи. Це додає додаткову стратегічну складову до змагань;

– Платформа автоматично оновлює рейтингові таблиці в реальному часі, відображаючи результати команд та учасників;

– Root The Box підтримує як командні, так і індивідуальні змагання, що дозволяє організаторам вибирати найбільш підходящий формат;

– Платформа надає API для інтеграції з іншими інструментами та розширення її функціональності.

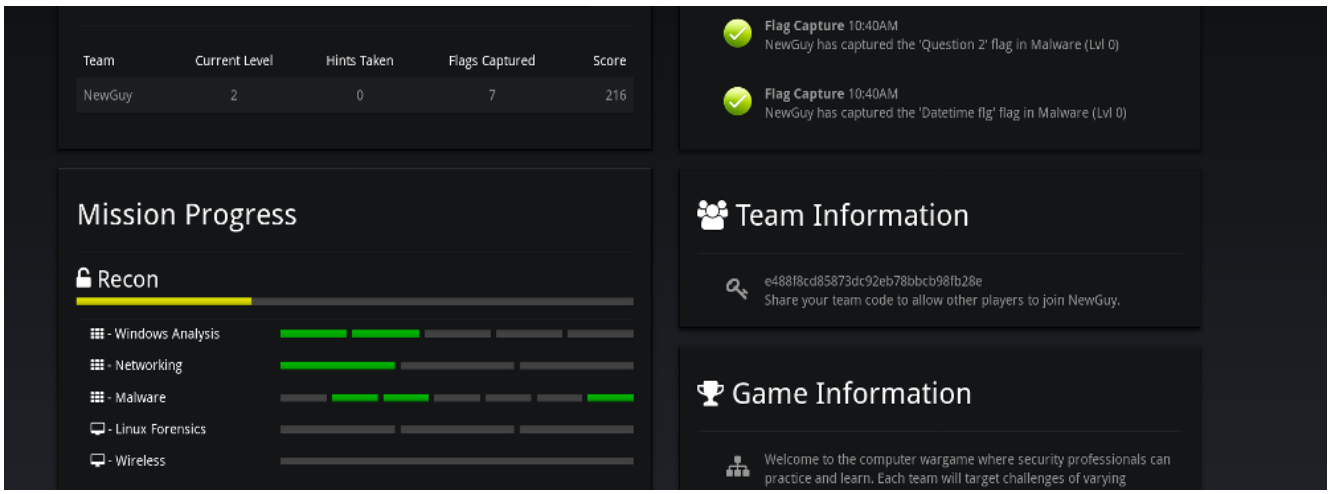


Рисунок 1.5.3 – Інтерфейс Root The Box

1.6 Ресурс готових VM-жерт VulnHub

VulnHub – це онлайн-ресурс, який надає користувачам віртуальні машини для практичного вивчення кібербезпеки та Penetration Testing. Він створений для

того, щоб допомогти людям покращити свої навички у галузі інформаційної безпеки через практичний досвід.

Основні характеристики VulnHub:

– VulnHub надає великий вибір віртуальних машин, створених для імітації різних типів вразливостей та конфігурацій систем. Користувачі можуть завантажувати дані віртуальні машини і запускати їх у своїх локальних середовищах для тестування;

– Віртуальні машини охоплюють широкий спектр сценаріїв з різними рівнями складності. Це дозволяє користувачам поступово нарощувати свої навички;

– VulnHub надає безкоштовний доступ до своїх ресурсів, що робить його доступним для всіх охочих навчатися кібербезпеці без фінансових витрат;

– VulnHub має активну спільноту користувачів, які діляться своїм досвідом та підказками. Це допомагає новачкам навчатися та знаходити підтримку від більш досвідчених користувачів.

РОЗДІЛ 2. ПРОЄКТУВАННЯ МЕРЕЖЕВОЇ ІНФРАСТРУКТУРИ ДЛЯ РОЗГОРТАННЯ ПЛАТФОРМИ ТА РОЗРОБКА СТРУКТУРИ ЗАВДАНЬ

2.1. Проєктування мережевої інфраструктури

Для розгортання платформи найкраще використовувати глобальну мережу (WAN), оскільки платформа повинна мати доступ до Інтернету, щоб клієнти могли отримати доступ до платформи та вирішувати завдання.

Глобальна мережа (WAN) – це мережа, яка з'єднує кілька локальних мереж (LAN), забезпечуючи взаємодію між ними на великих відстанях.

Таким чином, визначившись з типом мережі була спроєктована мережева інфраструктура, яка представлена на рисунку 2.1.1.

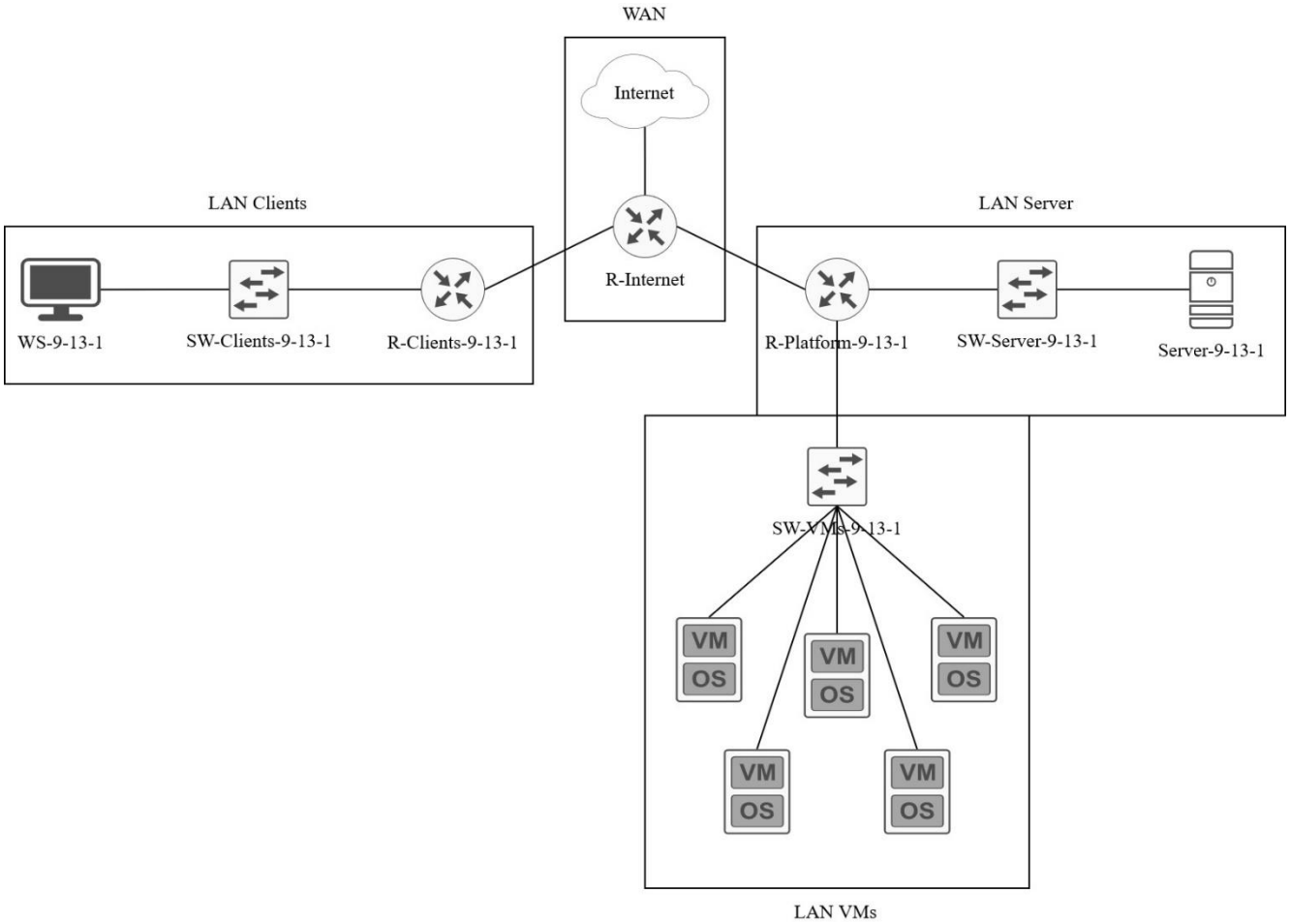


Рисунок 2.1.1 – Проєкт мережевої інфраструктури

Розглянемо проєкт мережевої інфраструктури більш детально.

Загалом мережа складається з підмереж WAN, LAN Clients, LAN Server та LAN VMs. Детально розглянемо кожну з підмереж.

Підмережа WAN включає в себе маршрутизатор R-Internet, який з'єднує між собою LAN Clients та LAN Server з WAN за допомогою NAT. Всі пристрої в підмережі з'єднанні за допомогою технології Fast Ethernet.

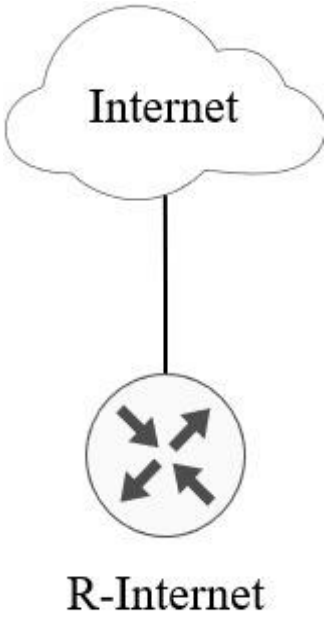


Рисунок 2.1.2 – Підмережа WAN

Підмережа LAN Clients включає в себе маршрутизатор R-Clients-9-13-1, комутатор 2-го рівня SW-Clients-9-13-1 та робочу станцію WS-9-13-1. Всі пристрої в підмережі з'єднанні за допомогою технологій Ethernet та Fast Ethernet.

Робоча станція WS-9-13-1 виступає у ролі клієнта, з даної робочої станції буде здійснюватись тестування функціоналу платформи.

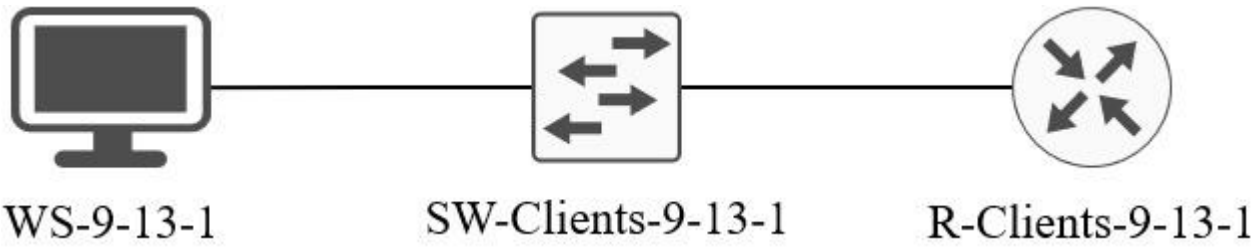


Рисунок 2.1.3 – Підмережа LAN Clients

Підмережа LAN Server включає в себе маршрутизатор R-Platform-9-13-1, комутатор 2-го рівня SW-Server-9-13-1 та віртуальний сервер Server-9-13-1. Всі пристрої в підмережі з'єднанні за допомогою технологій Ethernet та Fast Ethernet.

На віртуальному сервері Server-9-13-1 буде здійснюватися розгортання, налаштування платформи та встановлюватися VPN, який буде виконувати роль VPN-сервера.

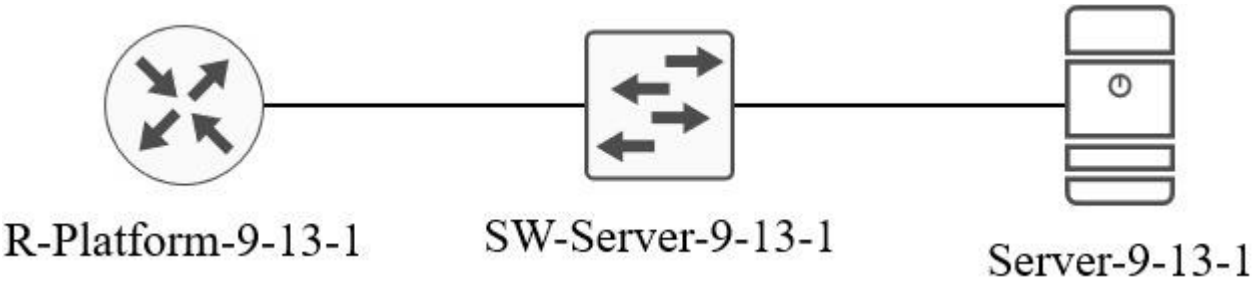


Рисунок 2.1.4 – Підмережа LAN Server

Підмережа LAN VMs включає в себе маршрутизатор R-Platform-9-13-1, комутатор 2-го рівня SW-VMs-9-13-1 та п'ять VM-жертв. Всі пристрої в підмережі з'єднанні за допомогою технологій Ethernet та Fast Ethernet.

Для кожної з VM-жертв будуть створюватися завдання на платформі.

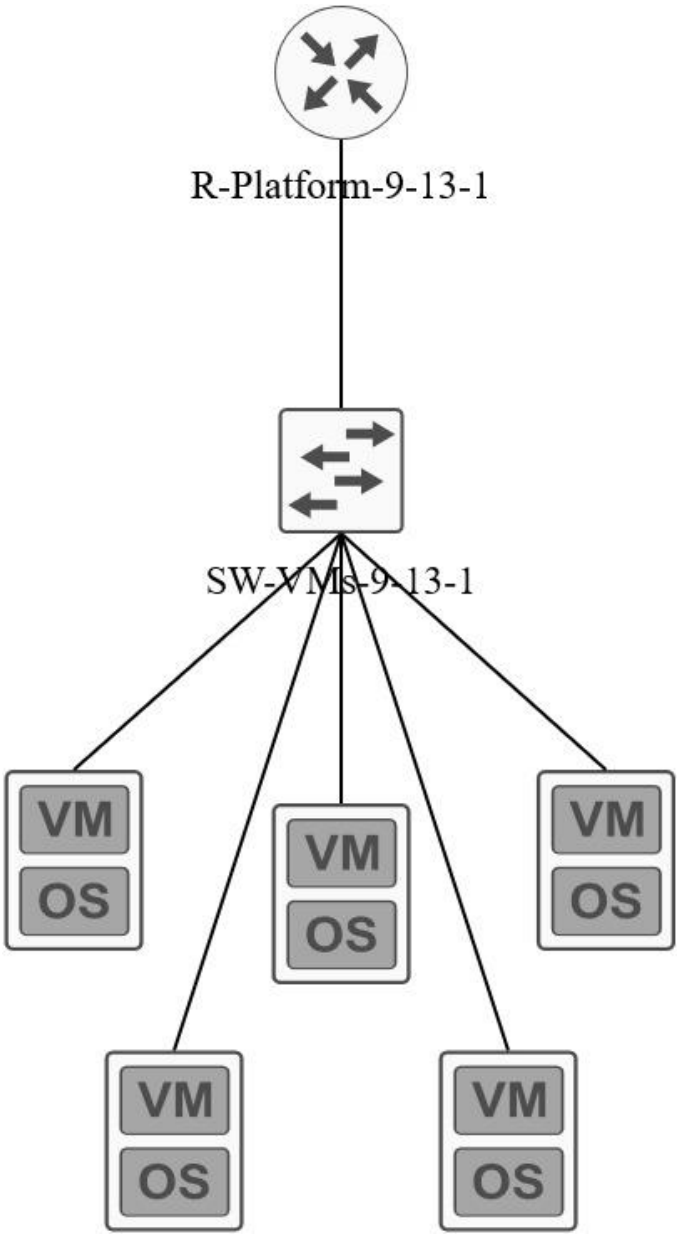


Рисунок 2.1.5 – Підмережа LAN VMs

2.2. Розробка схеми адресації мережі

Для того, щоб забезпечити наявність зв'язку між підмережами WAN, LAN Clients та LAN Server на маршрутизаторах будуть призначатися канали.

Канал A: R-Internet – WAN – 192.168.122.0/24

Канал B: R-Internet – R-Clients-9-13-1 – 193.9.13.0/24

Канал C: R-Internet – R-Platform-9-13-1 – 194.9.13.0/24

Розрахунок IP-адрес для підмереж наведений у таблицях 2.2.1 – 2.2.3.

Таблиця 2.2.1

Адресація підмережі Clients

Призначення	IP-адреса
Адреса мережі	195.9.13.0
Маска мережі	255.255.255.0/24
Інвертована маска	0.0.0.255
Мінімальна адреса	195.9.13.1
Максимальна адреса	195.9.13.254
Широкомовна адреса	195.9.13.255
Шлюз за замовчуванням	195.9.13.1
Адреса робочої станції (DHCP)	195.9.13.2
Діапазон вільних адрес	195.9.13.3 – 195.9.13.254

Таблиця 2.2.2

Адресація підмережі Server

Призначення	IP-адреса
Адреса мережі	200.9.13.0
Маска мережі	255.255.255.0/24
Інвертована маска	0.0.0.255
Мінімальна адреса	200.9.13.1
Максимальна адреса	200.9.13.254
Широкомовна адреса	200.9.13.255
Шлюз за замовчуванням	200.9.13.1
Адреса сервера	200.9.13.2
Діапазон вільних адрес	200.9.13.3 – 200.9.13.254

Таблиця 2.2.3

Адресація підмережі VMs

Призначення	IP-адреса
Адреса мережі	10.9.13.0
Маска мережі	255.255.255.0/24
Інвертована маска	0.0.0.255
Мінімальна адреса	10.9.13.1
Максимальна адреса	10.9.13.254
Широкомовна адреса	10.9.13.255
Шлюз за замовчуванням	10.9.13.1
Діапазон адрес VM-жертв (DHCP)	10.9.13.2 – 10.9.13.6
Діапазон вільних адрес	10.9.13.7 – 10.9.13.254

2.3 Вибір мережевого обладнання

При виборі моделі маршрутизатора потрібно враховувати як маршрутизатор маршрутизує пакети в мережі. Маршрутизатор повинен мати достатню кількість портів для функціонування мережі, забезпечувати можливість легкого розширення, необхідну швидкість передачі даних, швидкість їх обробки тощо.

Для коректного функціонування мережі потрібно обрати маршрутизатор, який підтримує технологію Fast Ethernet, протоколи динамічної маршрутизації та містить усі потрібні функції для налаштування: NAT, DHCP, Rate-Limit тощо.

Для реалізації вищеперелічених пунктів, найбільш оптимальним варіантом є маршрутизатор Cisco серії 7200.



Рисунок 2.3.1 – Маршрутизатор Cisco 7206VXR/NPE-G2

Маршрутизатори серії 7200 здобули світову популярність, багато компаній надійно використовують їх для своїх мережних потреб. Ці маршрутизатори мають високу продуктивність і можуть передавати дані у багатьох потоках, що розширює можливості обчислювальних центрів.

Маршрутизатори серії 7200 володіють процесорами, які розроблені Cisco, що позитивно впливає на їхню продуктивність. За допомогою чіпів Cisco NPE-G2 створюється мережева платформа, яка не лише забезпечує швидку передачу даних, але й ефективно справляється з високим навантаженням на каналах передачі. В таких ситуаціях активується магістральна передача інформації, що дозволяє швидко відправити надлишок пакетів за короткий період. При великій кількості пакетів, які надсилаються за секунду, цей продукт на повну виправдовує свої витрати.

Маршрутизатори серії 7200 мають нову функцію IP/MPLS, що дозволяє користувачам використовувати широкосмугову комутацію, постійний доступ та агрегацію підключень. Адміністратор може вибирати оптимальний інтерфейс відповідно до складності проекту. Початковим рівнем може бути інтерфейс DS 0, а з розвитком системи користувач сам може вибирати потрібний режим роботи. Сімейство маршрутизаторів серії 7200 підтримує термінацію PPP і L2TP, а також може обробляти до 16 тисяч абонентських сесій.

Серед комутаторів найбільш оптимальним варіантом є Cisco Catalyst серії 2960. Дані комутатори є лінійкою з фіксованою конфігурацією та портами Ethernet, Fast Ethernet, Gigabit Ethernet, що мають розширені LAN сервіси. Комутатори Cisco Catalyst серії 2960 підтримують передачу голосу, даних, відео, а також безпечний доступ.



Рисунок 2.3.2 – Комутатор Cisco WS-C2960-24TT-L

Комутатори Cisco Catalyst 2960 підтримують створення складних списків управління доступом, розширену безпеку, комбіновані гігабітні аплінки (мідний 10/100/1000BASE-T Ethernet або SFP-модуль).

Крім того, пристрої Catalyst 2960 підтримують QoS, обмеження швидкості, ACL (на основі MAC або IP-адреси, UDP/TCP-порту), багатоадресні служби, можливість регулювати швидкість передачі кожного порту з кроком 64 кбіт, агрегацію каналів для досягнення більшої швидкості. швидкості. Організуйте з'єднання між комутаторами та серверами, організуйте базові з'єднання на порт за допомогою тегів 802.1q, до 255 VLAN на комутатор, до 4000 ідентифікаторів VLAN і можливість керувати за допомогою Cisco Network Assistant.

Прошивка комутаторів Cisco Catalyst 2960 випускається в образах LAN Base та LAN Lite Image [22].

LAN Base має ширший набір функцій, включаючи покращену безпеку (ACL), відстеження DHCP, додаткові можливості контролю доступу, веб-автентифікацію, розширення 802.1x, додаткові можливості установок якості обслуговування QoS, підтримку резервного джерела живлення RPS і велику кількість портів SFP. Функції Flex Links, відстеження стану зв'язку, збільшена кількість підтримуваних VLAN (до 256), хости IPv6, MLD Snooping, LLDP-MED, RSPAN, MVR, DHCP Option 82, IP SLA (Responder) тощо.

2.4 Вибір операційних систем, програмного забезпечення та VM-жертв

Проєкт мережевої інфраструктури буде реалізовуватись за допомогою GNS3.

Graphical Network Simulator-3 (GNS3) – це емулятор мережевого програмного забезпечення, вперше випущений у 2008 році. Він дозволяє поєднувати віртуальні та реальні пристрої для моделювання мереж.

GNS3 підтримує емуляцію маршрутизаторів Cisco, за допомогою Dynamips, що дозволяє імітувати Cisco IOS для цих пристроїв.

Для емуляції комутаторів Cisco зазвичай використовуються IOU або L2IOU образи, що дозволяє створювати точні мережеві моделі з підтримкою комутації на другому рівні.

На робочій станції WS-9-13-1 буде використовуватися ОС Kali Linux для проведення тестування платформи.

Kali Linux – це дистрибутив Linux з відкритим вихідним кодом, заснований на Debian, призначений для розширеного тестування на проникнення та аудиту безпеки [18].

Kali Linux містить галузеві модифікації, а також кілька сотень інструментів, орієнтованих на різні завдання інформаційної безпеки, такі як тестування на проникнення, дослідження безпеки, комп'ютерна криміналістика тощо.

На віртуальному сервері Server-9-13-1 буде використовуватися ОС Ubuntu, в якості гіпервізора типу 2 виступає Oracle VM VirtualBox.

Ubuntu – це дистрибутив Linux, заснований на Debian GNU/Linux. Він є одним із найпопулярніших дистрибутивів Linux у світі, відомий своєю простотою використання, стабільністю та орієнтацією на користувача.

Для коректного та успішного розгортання платформи буде встановлюватися Docker, оскільки дане ПЗ може створювати локальні ізольовані екземпляри.

Docker – це відкрита платформа для розробки, доставки та запуску додатків. Docker дозволяє відокремити додатки від інфраструктури, щоб швидко доставляти програмне забезпечення.

В якості готового рішення для розгортання платформи на Server-9-13-1 буде використовуватися CTFd, бо дана платформа пропонує простий у використанні інтерфейс, широкий набір функцій для створення та адміністрування завдань.

Через те, що LAN VMs немає доступу в Інтернет на Server-9-13-1 буде встановлюватися OpenVPN, який буде виконувати роль VPN-сервера. VM-жертви будуть підключені до VPN-сервера.

В якості VM-жертв для LAN VMs серед величезного списку наявних машин на ресурсі VulnHub будуть використовуватися наступні:

– SolidState;
– WebDeveloper;
– Straylight;
– zico2;

– DC-1.

SolidState – це віртуальна машина на базі Linux, створена Ch33z_plz. Дана машина середня за складністю та підходить для користувачів, які мають певний досвід у тестуванні на проникнення.

WebDeveloper – це віртуальна машина на базі Linux, створена Fred Wemeijer. Дана машина початкового рівня та підходить для користувачів, які роблять свої перші кроки в тестуванні на проникнення.

Straylight – це віртуальна машина на базі Linux, створена creosote. Дана машина середньо-високого рівня складності та підходить для користувачів, які мають певний досвід у тестуванні на проникнення.

zico2 – це віртуальна машина на базі Linux, створена Rafael. Дана машина середньо-високого рівня складності та підходить для користувачів, які мають певний досвід у тестуванні на проникнення.

DC-1 – це віртуальна машина на базі Linux, створена DCAU. Дана машина початкового рівня складності та підходить для користувачів, які роблять свої перші кроки у тестуванні на проникнення.

2.5 Розробка структури завдань

Під кожну VM-жертву на платформі буде створюватися серія завдань, яка буде мати підказки для допомоги у вирішенні. Суть полягає в тому, що вирішення першого завдання в серії буде відкривати наступне завдання і так до тих пір, поки всі завдання не будуть виконані. Саме тому на платформі буде використовуватися Linear (Лінійний) тип CTF.

Розробка структури серії завдань для машини SolidState:

– Перше завдання, SolidState (Вступ), в даному завданні буде зазначатися опис, IP-адреса/посилання на веб-сайт VM-жертви, конфігураційний файл .ovpn для отримання доступу до машини та безкоштовна підказка в якій буде міститися прапор для виконання завдання. Після того, як користувач надішле даний прапор завдання автоматично зарахується та одразу стане доступним наступне завдання в серії. За виконання даного завдання не будуть нараховуватися бали;

– Друге завдання, SolidState (Перший прапор), в даному завданні будуть зазначатися дві безкоштовні підказки, в яких буде вказуватися, що потрібно зробити чи за допомогою яких інструментів можна наблизитися до прапора для виконання завдання. Після того, як користувач знайде і надішле даний прапор завдання автоматично зарахується та одразу стане доступним наступне завдання в серії. За виконання даного завдання буде нараховуватися 150 балів;

– Третє/Фінальне завдання, SolidState (Другий прапор), в даному завданні будуть зазначатися дві заблоковані підказки, для того щоб їх розблокувати потрібно витратити певну кількість балів, оскільки за виконання минулого завдання користувач може отримати 150 балів, а поточне завдання має дві заблоковані підказки, то розблокування кожної з них буде коштувати по 75 балів. Користувач сам вирішує, що краще зробити, розблокувати одну чи дві підказки або взагалі не витрачати бали та знайти прапор для виконання завдання самостійно. За виконання даного завдання буде нараховуватися 150 балів.

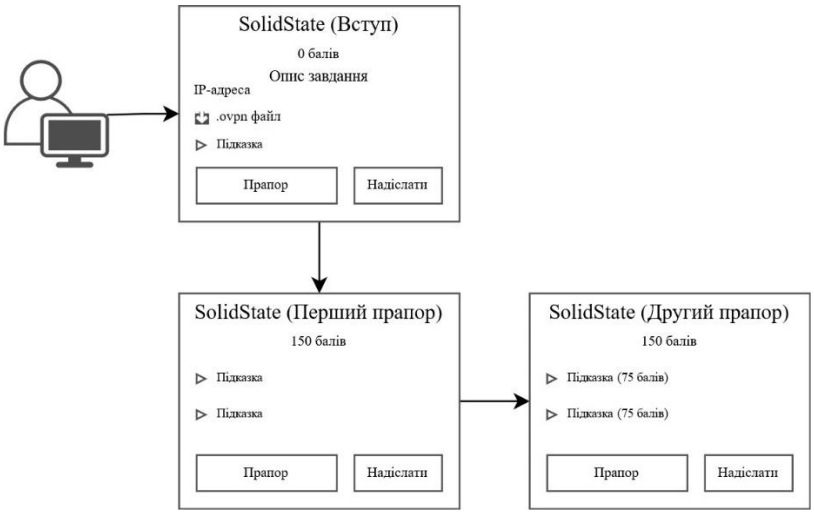


Рисунок 2.5.1 – Структура серії завдань для машини SolidState

Розробка структури серії завдань для машини WebDeveloper:

– Перше завдання, WebDeveloper (Вступ), в даному завданні буде зазначатися опис, IP-адреса/посилання на веб-сайт VM-жертви, конфігураційний файл .ovpn для отримання доступу до машини та безкоштовна підказка в якій буде міститися прапор для виконання завдання. Після того, як користувач надішле даний прапор завдання автоматично зарахується та одразу стане доступним наступне завдання в серії. За виконання даного завдання не будуть нараховуватися бали;

– Друге завдання, WebDeveloper (Перший прапор), в даному завданні будуть зазначатися три безкоштовні підказки, в яких буде вказуватися, що потрібно зробити чи за допомогою яких інструментів можна наблизитися до прапора для виконання завдання. Після того, як користувач знайде і надішле даний прапор завдання автоматично зарахується та одразу стане доступним наступне завдання в серії. За виконання даного завдання буде нараховуватися 50 балів;

– Третє завдання, WebDeveloper (Другий прапор), в даному завданні будуть зазначатися дві заблоковані підказки, для того щоб їх розблокувати потрібно витратити певну кількість балів, оскільки за виконання минулого завдання користувач може отримати 50 балів, а поточне завдання має дві заблоковані підказки, то розблокування кожної з них буде коштувати по 25 балів. Користувач сам вирішує, що краще зробити, розблокувати одну чи дві підказки або взагалі не витрачати бали та знайти прапор для виконання завдання самостійно. За виконання даного завдання буде нараховуватися 50 балів;

– Четверте/Фінальне завдання, WebDeveloper (Третій прапор), в даному завданні буде зазначатися одна заблокована підказка, для того щоб її розблокувати потрібно витратити певну кількість балів, оскільки за виконання минулого завдання користувач може отримати 50 балів, а поточне завдання має одну заблоковану підказку, то розблокування даної підказки буде коштувати 50 балів. Користувач сам вирішує, що краще зробити, розблокувати дану підказку або взагалі не витрачати бали та знайти прапор для виконання завдання самостійно. За виконання даного завдання буде нараховуватися 50 балів.

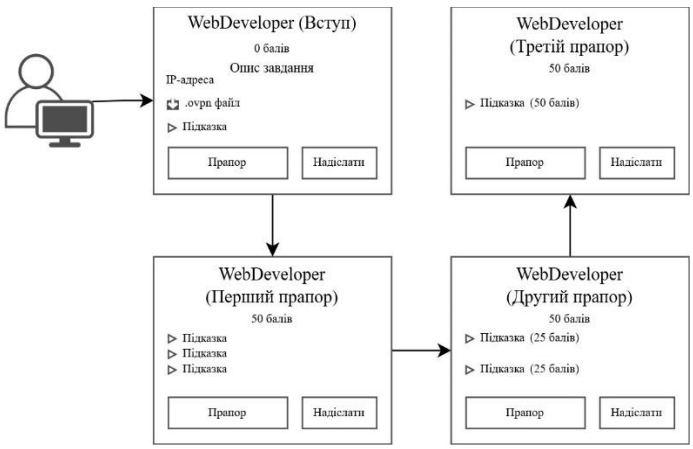


Рисунок 2.5.2 – Структура серії завдань для машини WebDeveloper

Розробка структури серії завдань для машини Straylight:

– Перше завдання, Straylight (Вступ), в даному завданні буде зазначатися опис, IP-адреса/посилання на веб-сайт VM-жертви, конфігураційний файл .ovpn для отримання доступу до машини та безкоштовна підказка в якій буде міститися прапор для виконання завдання. Після того, як користувач надішле даний прапор завдання автоматично зарахується та одразу стане доступним наступне завдання в серії. За виконання даного завдання не будуть нараховуватися бали;

– Друге завдання, Straylight (Перший прапор), в даному завданні будуть зазначатися дві безкоштовні підказки, в яких буде вказуватися, що потрібно зробити чи за допомогою яких інструментів можна наблизитися до прапора для виконання завдання. Після того, як користувач знайде і надішле даний прапор завдання автоматично зарахується та одразу стане доступним наступне завдання в серії. За виконання даного завдання буде нараховуватися 250 балів;

– Третє/Фінальне завдання, Straylight (Другий прапор), в даному завданні буде зазначатися одна заблокована підказка, для того щоб її розблокувати потрібно витратити певну кількість балів, оскільки за виконання минулого завдання користувач може отримати 250 балів, а поточне завдання має одну заблоковану підказку, то розблокування даної підказки буде коштувати 250 балів. Користувач сам вирішує, що краще зробити, розблокувати дану підказку або взагалі не витрачати бали та знайти прапор для виконання завдання самостійно. За виконання даного завдання буде нараховуватися 250 балів.

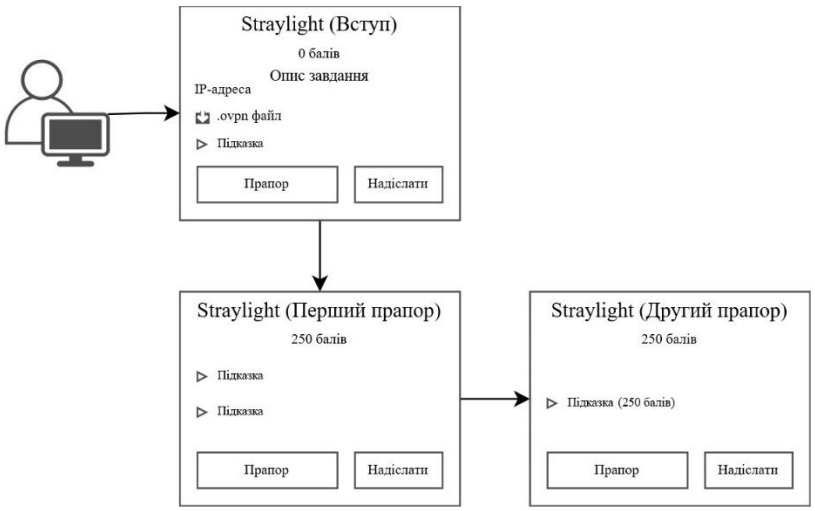


Рисунок 2.5.3 – Структура серії завдань для машини Straylight

Розробка структури серії завдань для машини zico2:

– Перше завдання, zico2 (Вступ), в даному завданні буде зазначатися опис, IP-адреса/посилання на веб-сайт VM-жертви, конфігураційний файл .ovpn для отримання доступу до машини та безкоштовна підказка в якій буде міститися прапор для виконання завдання. Після того, як користувач надішле даний прапор завдання автоматично зарахується та одразу стане доступним наступне завдання в серії. За виконання даного завдання не будуть нараховуватися бали;

– Друге завдання, zico2 (Перший прапор), в даному завданні будуть зазначатися дві безкоштовні підказки, в яких буде вказуватися, що потрібно зробити чи за допомогою яких інструментів можна наблизитися до прапора для виконання завдання. Після того, як користувач знайде і надішле даний прапор завдання автоматично зарахується та одразу стане доступним наступне завдання в серії. За виконання даного завдання буде нараховуватися 200 балів;

– Третє завдання, zico2 (Другий прапор), в даному завданні будуть зазначатися дві заблоковані підказки, для того щоб їх розблокувати потрібно витратити певну кількість балів, оскільки за виконання минулого завдання користувач може отримати 200 балів, а поточне завдання має дві заблоковані підказки, то розблокування кожної з них буде коштувати по 100 балів. Користувач сам вирішує, що краще зробити, розблокувати одну чи дві підказки або взагалі не витрачати бали та знайти прапор для виконання завдання самостійно. За виконання даного завдання буде нараховуватися 200 балів;

– Четверте/Фінальне завдання, zico2 (Третій прапор), в даному завданні буде зазначатися одна заблокована підказка, для того щоб її розблокувати потрібно витратити певну кількість балів, оскільки за виконання минулого завдання користувач може отримати 200 балів, а поточне завдання має одну заблоковану підказку, то розблокування даної підказки буде коштувати 200 балів. Користувач сам вирішує, що краще зробити, розблокувати дану підказку або взагалі не витрачати бали та знайти прапор для виконання завдання самостійно. За виконання даного завдання буде нараховуватися 200 балів.

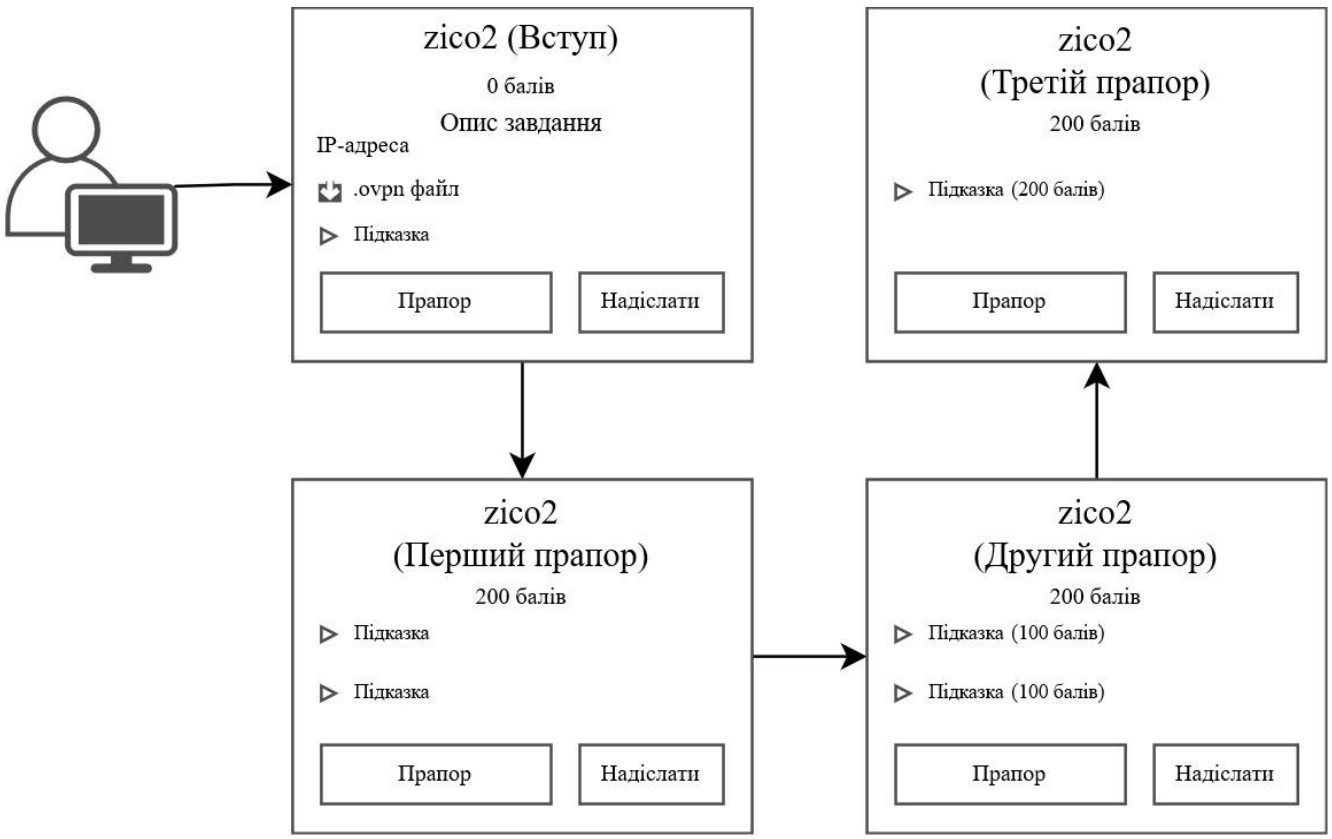


Рисунок 2.5.4 – Структура серії завдань для машини zico2

Розробка структури серії завдань для машини DC-1:

– Перше завдання, DC-1 (Вступ), в даному завданні буде зазначатися опис, IP-адреса/посилання на веб-сайт VM-жертви, конфігураційний файл .ovpn для отримання доступу до машини та безкоштовна підказка в якій буде міститися прапор для виконання завдання. Після того, як користувач надішле даний прапор завдання автоматично зарахується та одразу стане доступним наступне завдання в серії. За виконання даного завдання не будуть нараховуватися бали;

– Друге завдання, DC-1 (Перший прапор), в даному завданні буде зазначатися одна безкоштовна підказка, в якій буде вказуватися, що потрібно зробити чи за допомогою яких інструментів можна наблизитися до прапора для виконання завдання. Після того, як користувач знайде і надішле даний прапор завдання автоматично зарахується та одразу стане доступним наступне завдання в серії. За виконання даного завдання буде нараховуватися 90 балів;

– Третє завдання, DC-1 (Другий прапор), в даному завданні буде зазначатися одна заблокована підказка, для того щоб її розблокувати потрібно витратити певну кількість балів, оскільки за виконання минулого завдання користувач може

отримати 90 балів, а поточне завдання має одну заблоковану підказку, то розблокування даної підказки буде коштувати 90 балів. Користувач сам вирішує, що краще зробити, розблокувати дану підказку або взагалі не витрачати бали та знайти прапор для виконання завдання самостійно. За виконання даного завдання буде нараховуватися 90 балів;

– Четверте завдання, DC-1 (Третій прапор), в даному завданні будуть зазначатися дві заблоковані підказки, для того щоб їх розблокувати потрібно витратити певну кількість балів, оскільки за виконання минулого завдання користувач може отримати 90 балів, а поточне завдання має дві заблоковані підказки, то розблокування кожної з них буде коштувати по 45 балів. Користувач сам вирішує, що краще зробити, розблокувати одну чи дві підказки або взагалі не витрачати бали та знайти прапор для виконання завдання самостійно. За виконання даного завдання буде нараховуватися 90 балів;

– П'яте завдання, DC-1 (Четвертий прапор), в даному завданні буде зазначатися одна заблокована підказка, для того щоб її розблокувати потрібно витратити певну кількість балів, оскільки за виконання минулого завдання користувач може отримати 90 балів, а поточне завдання має одну заблоковану підказку, то розблокування даної підказки буде коштувати 90 балів. Користувач сам вирішує, що краще зробити, розблокувати дану підказку або взагалі не витрачати бали та знайти прапор для виконання завдання самостійно. За виконання даного завдання буде нараховуватися 90 балів;

– Шосте/Фінальне завдання, DC-1 (П'ятий прапор), в даному завданні буде зазначатися одна заблокована підказка, для того щоб її розблокувати потрібно витратити певну кількість балів, оскільки за виконання минулого завдання користувач може отримати 90 балів, а поточне завдання має одну заблоковану підказку, то розблокування даної підказки буде коштувати 90 балів. Користувач сам вирішує, що краще зробити, розблокувати дану підказку або взагалі не витрачати бали та знайти прапор для виконання завдання самостійно. За виконання даного завдання буде нараховуватися 90 балів.

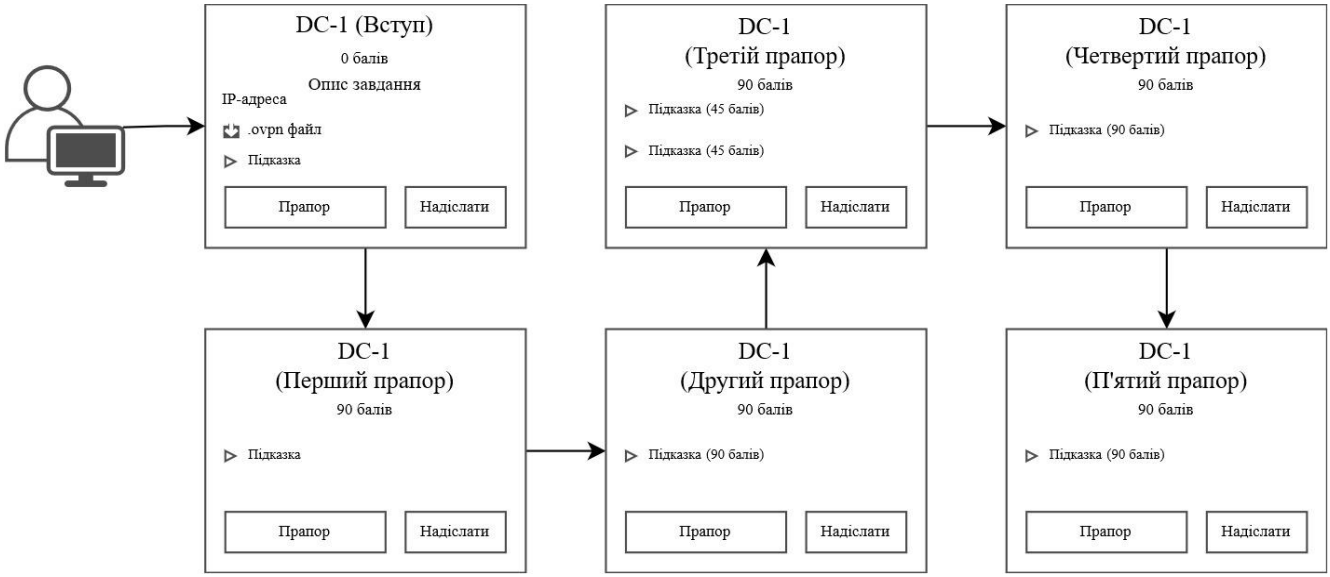


Рисунок 2.5.5 – Структура серії завдань для машини DC-1

Для того, щоб запобігти перебору прапора за допомогою методу «грубої сили» (Brute-force attack) для кожного завдання в серії, окрім першого завдання буде встановлюватися обмеження в максимальну кількість спроб (3 спроби).

РОЗДІЛ 3 РОЗГОРТАННЯ, НАЛАШТУВАННЯ, ТЕСТУВАННЯ РОБОТИ МЕРЕЖІ ТА ФУНКЦІОНАЛУ ПЛАТФОРМИ

3.1 Налаштування мережевого обладнання

Мережева інфраструктура створена відповідно до проєктованої схеми.

Перш ніж почати розгортання платформи, налаштуємо параметри адресації на мережевому обладнані.

```
R-Internet#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
R-Internet(config)#int fa0/0
R-Internet(config-if)#ip add dhcp
R-Internet(config-if)#no sh
R-Internet(config-if)#exit
R-Internet(config)#int fa1/0
R-Internet(config-if)#ip add 193.9.13.1 255.255.255.0
R-Internet(config-if)#no sh
R-Internet(config-if)#exit
R-Internet(config)#int fa2/0
R-Internet(config-if)#ip add 194.9.13.1 255.255.255.0
R-Internet(config-if)#no sh
R-Internet(config-if)#exit
R-Internet(config)#exit
```

Рисунок 3.1.1 – Налаштування параметрів адресації на R-Internet

Налаштування параметрів адресації на інших маршрутизаторах мережі проводяться аналогічно.

Налаштуємо R-Internet як DNS-сервера.

```
R-Internet#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
R-Internet(config)#ip dns server
R-Internet(config)#ip domain-lookup
R-Internet(config)#exit
```

Рисунок 3.1.2 – Налаштування R-Internet як DNS-сервера

Налаштуємо параметри адресації на Server-9-13-1.

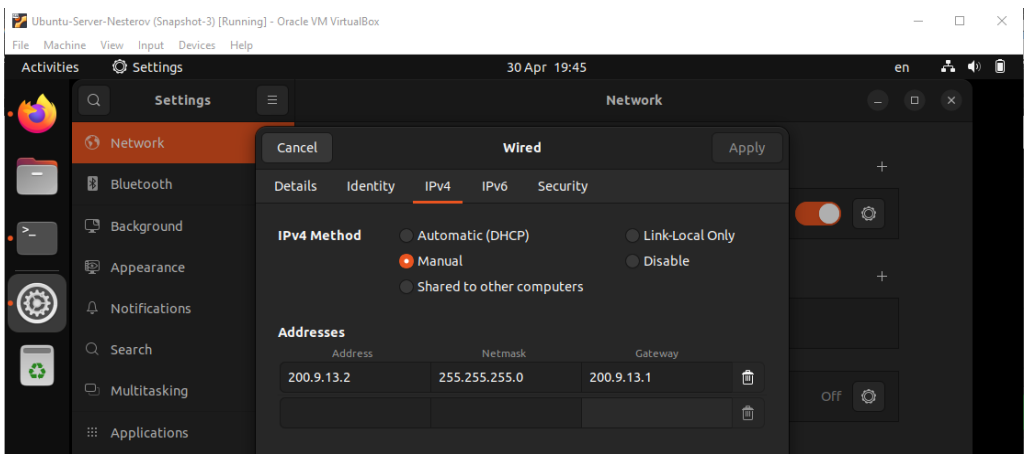


Рисунок 3.1.3 – Налаштування параметрів адресації на Server-9-13-1

Наступним кроком є налаштування функціонування маршрутизаторів мережі як DHCP-серверів.

```
R-Clients-9-13-1#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
R-Clients-9-13-1(config)#ip dhcp excluded-address 195.9.13.1
R-Clients-9-13-1(config)#
R-Clients-9-13-1(config)#ip dhcp pool Clients
R-Clients-9-13-1(dhcp-config)#network 195.9.13.0 255.255.255.0
R-Clients-9-13-1(dhcp-config)#default-router 195.9.13.1
R-Clients-9-13-1(dhcp-config)#dns-server 192.168.122.23
R-Clients-9-13-1(dhcp-config)#exit
R-Clients-9-13-1(config)#exit
```

Рисунок 3.1.4 – Налаштування R-Clients-9-13-1 як DHCP-сервера

```
R-Platform-9-13-1#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
R-Platform-9-13-1(config)#ip dhcp excluded-address 10.9.13.1
R-Platform-9-13-1(config)#
R-Platform-9-13-1(config)#ip dhcp pool VMs
R-Platform-9-13-1(dhcp-config)#network 10.9.13.0 255.255.255.0
R-Platform-9-13-1(dhcp-config)#default-router 10.9.13.1
R-Platform-9-13-1(dhcp-config)#dns-server 192.168.122.23
R-Platform-9-13-1(dhcp-config)#exit
R-Platform-9-13-1(config)#exit
```

Рисунок 3.1.5 – Налаштування R-Platform-9-13-1 як DHCP-сервера

Налаштуємо Static DHCP для VM-жертв.

```
R-Platform-9-13-1#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
R-Platform-9-13-1(config)#ip dhcp pool SolidState
R-Platform-9-13-1(dhcp-config)#host 10.9.13.2 255.255.255.0
R-Platform-9-13-1(dhcp-config)#hardware-address 0800.2729.fb2b
R-Platform-9-13-1(dhcp-config)#exit
R-Platform-9-13-1(config)#exit
```

Рисунок 3.1.6 – Налаштування Static DHCP для SolidState на R-Platform-9-13-1

Налаштування Static DHCP для інших VM-жертв проводяться аналогічно.

Для того, щоб з'єднати між собою підмережі LAN Clients і LAN Server з WAN налаштуємо NAT, щоб забезпечити зв'язок між усіма підмережами налаштуємо протокол динамічної маршрутизації OSPF.

```
R-Internet#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
R-Internet(config)#int fa0/0
R-Internet(config-if)#ip nat outside
R-Internet(config-if)#exit
R-Internet(config)#int range fa 1/0 , fa 2/0
R-Internet(config-if-range)#ip nat inside
R-Internet(config-if-range)#exit
R-Internet(config)#ip nat inside source list 1 int fa 0/0 overload

R-Internet(config)#access-list 1 permit any
R-Internet(config)#exit
```

Рисунок 3.1.7 – Налаштування NAT на R-Internet

```
R-Internet#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
R-Internet(config)#router ospf 913
R-Internet(config-router)#network 192.168.122.0 0.0.0.255 area 0
R-Internet(config-router)#network 193.9.13.0 0.0.0.255 area 0
R-Internet(config-router)#network 194.9.13.0 0.0.0.255 area 0
R-Internet(config-router)#exit
R-Internet(config)#exit
```

Рисунок 3.1.8 – Налаштування динамічної маршрутизації OSPF на R-Internet

Налаштування протоколу динамічної маршрутизації OSPF на інших маршрутизаторах мережі проводиться аналогічно.

Налаштуємо Rate-Limit для захисту від DoS (DDoS) атак.

```
R-Internet#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
R-Internet(config)#access-list 101 permit udp any any
R-Internet(config)#access-list 101 permit icmp any any echo
R-Internet(config)#access-list 101 permit icmp any any echo-reply
R-Internet(config)#
R-Internet(config)#int range fa 1/0 , fa 2/0 , fa 3/0
R-Internet(config-if-range)#ip verify unicast reverse-path
R-Internet(config-if-range)#$ conform-action transmit exceed-action drop
R-Internet(config-if-range)#$0 conform-action transmit exceed-action drop
R-Internet(config-if-range)#exit
R-Internet(config)#ip tcp synwait-time 30
R-Internet(config)#exit
```

Рисунок 3.1.9 – Налаштування Rate-Limit на R-Internet

Налаштування Rate-Limit на інших маршрутизаторах мережі проводиться аналогічно.

3.2 Розгортання та налаштування платформи

Перед розгортанням та налаштуванням платформи на Server-9-13-1 потрібно оновити список пакетів та встановлені пакети до їхніх нових версій, щоб уникнути проблем під час подальших налаштувань.

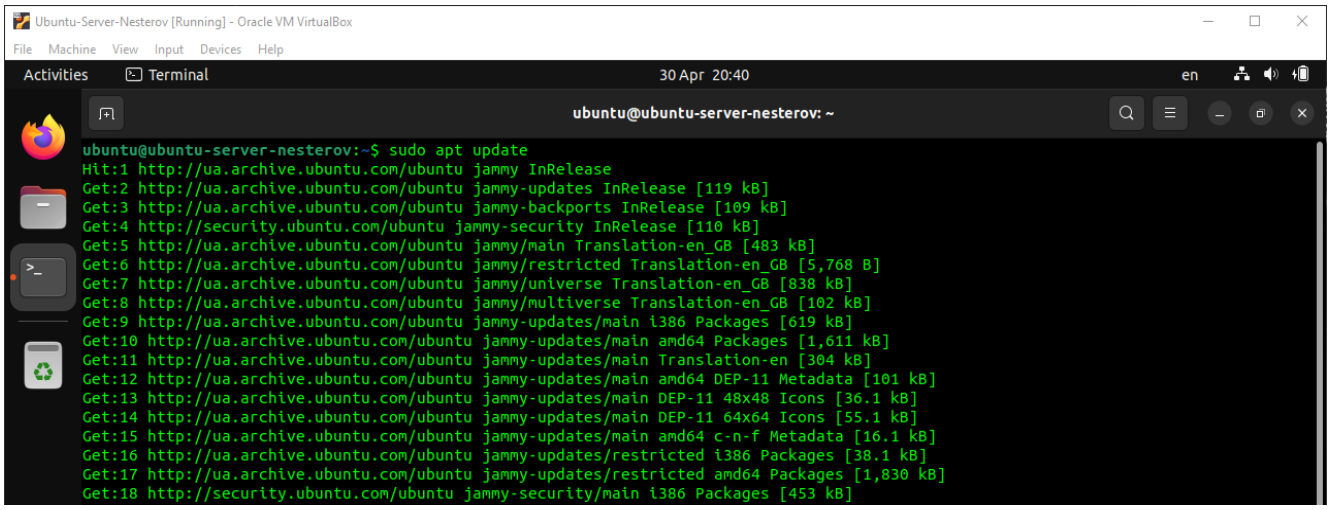


Рисунок 3.2.1 – Оновлення списку пакетів за допомогою команди sudo apt update

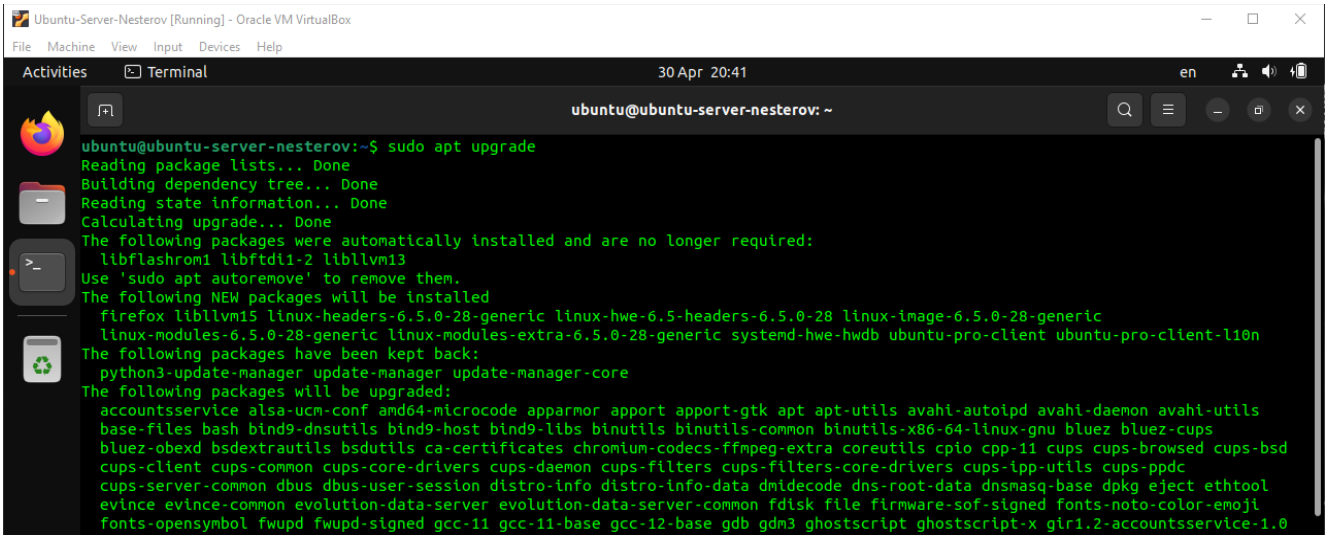


Рисунок 3.2.2 – Оновлення встановлених пакетів до їхніх нових версій за допомогою команди sudo apt upgrade

Наступним кроком є встановлення Docker.

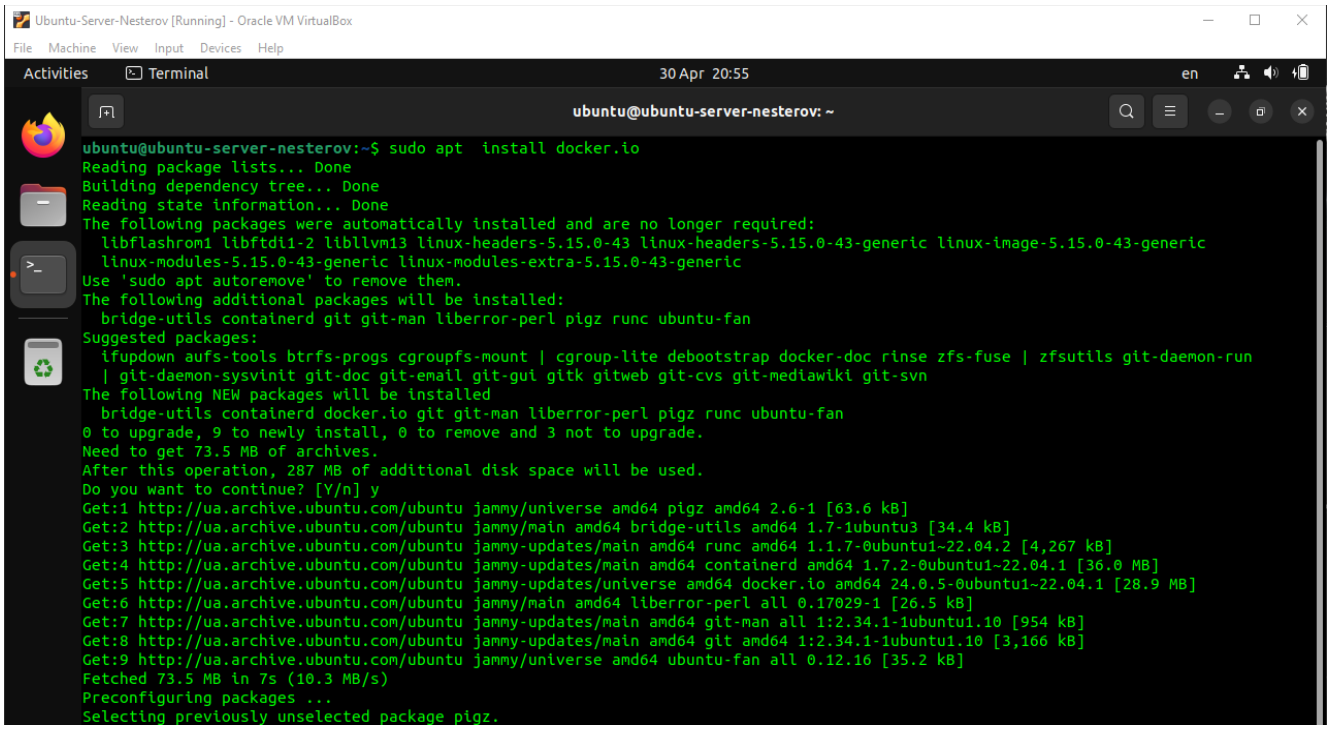


Рисунок 3.2.3 – Встановлення Docker за допомогою команди sudo apt install docker.io

Docker успішно встановлено, наступним кроком є встановлення зміної DOCKER_CONFIG використовуючи значення за замовчуванням $HOME/.docker, після створюємо директорію cli-plugins в яку завантажуємо Docker Compose та встановлюємо права для виконання, після перевіряємо правильності встановлення за допомогою команди docker compose version.

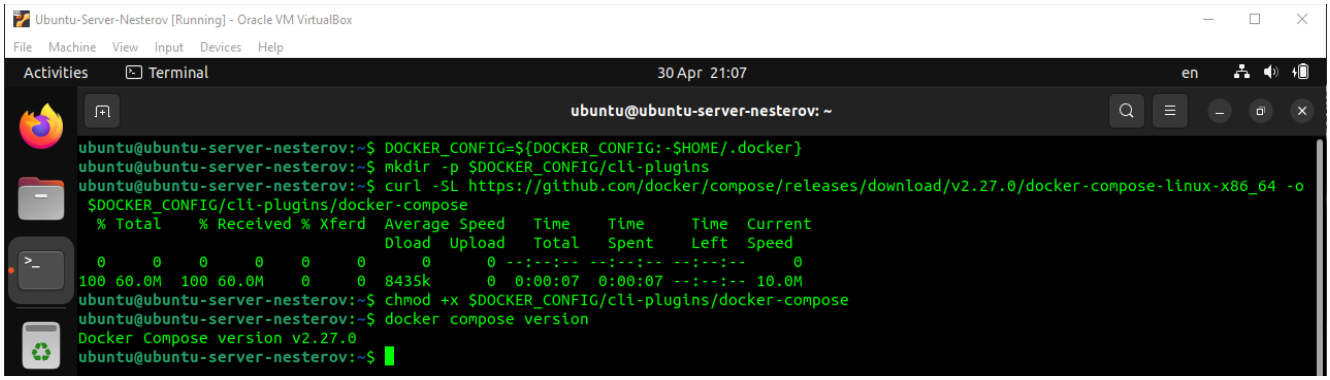


Рисунок 3.2.4 – Встановлення Docker Compose

Наступним кроком є розгортання платформи на базі рішення CTFd. Спочатку завантажуємо ресурси CTFd з репозиторію, після переходимо в директорію CTFd та розгортаємо платформу за допомогою команди docker compose up.

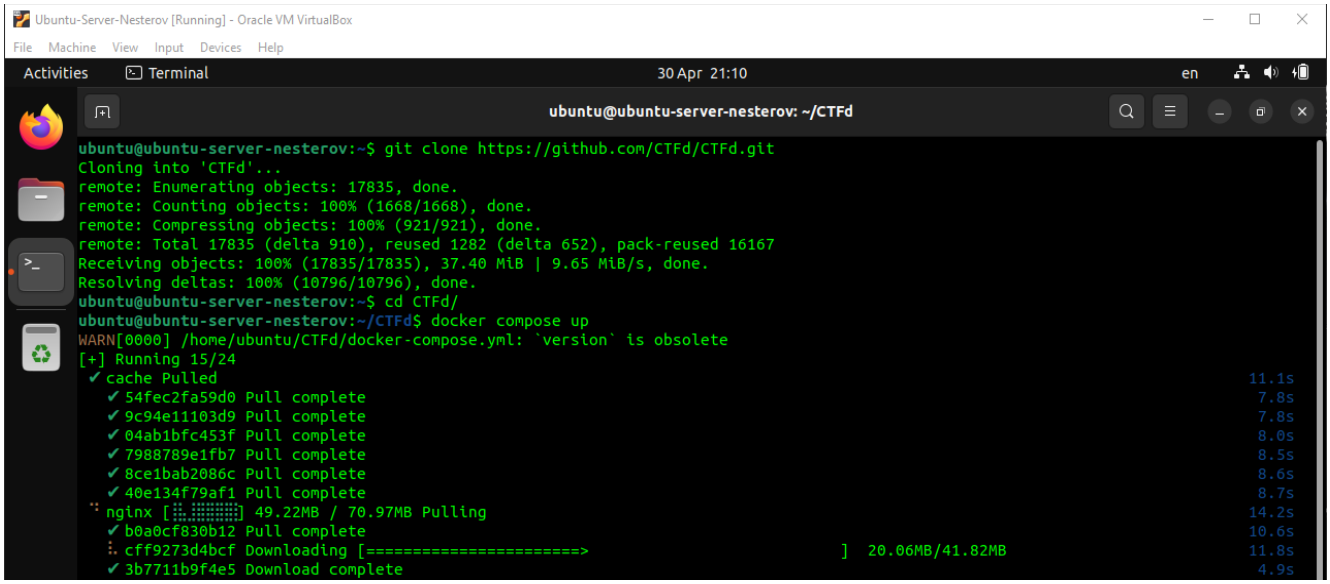


Рисунок 3.2.5 – Розгортання платформи на базі рішення CTFd

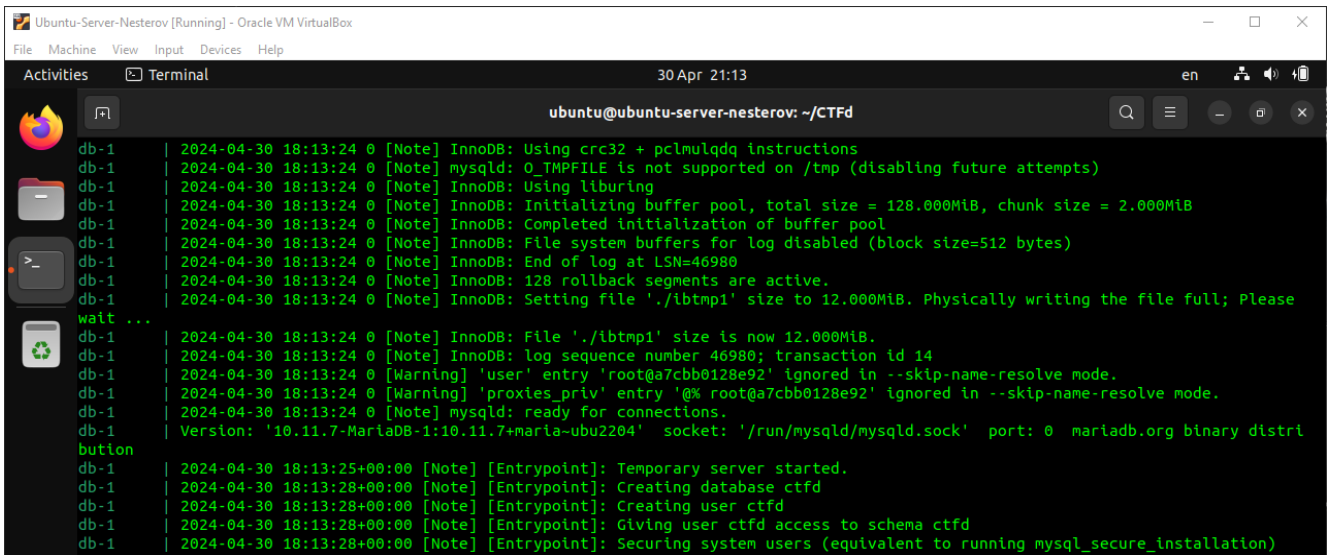


Рисунок 3.2.6 – Розгортання платформи на базі рішення CTFd

Платформу на базі рішення CTFd успішно розгорнуто, наступним кроком є її налаштування.

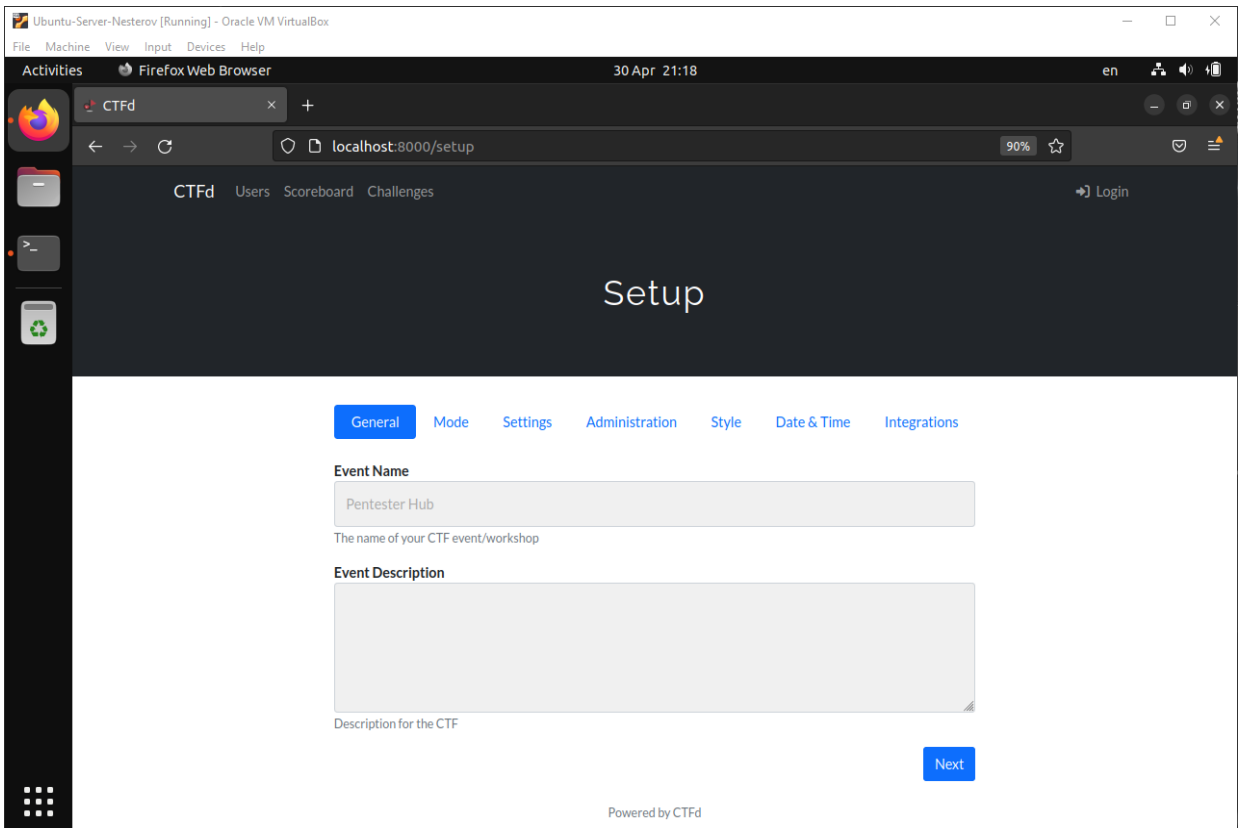


Рисунок 3.2.7 – Налаштування платформи на базі CTFd, задання назви

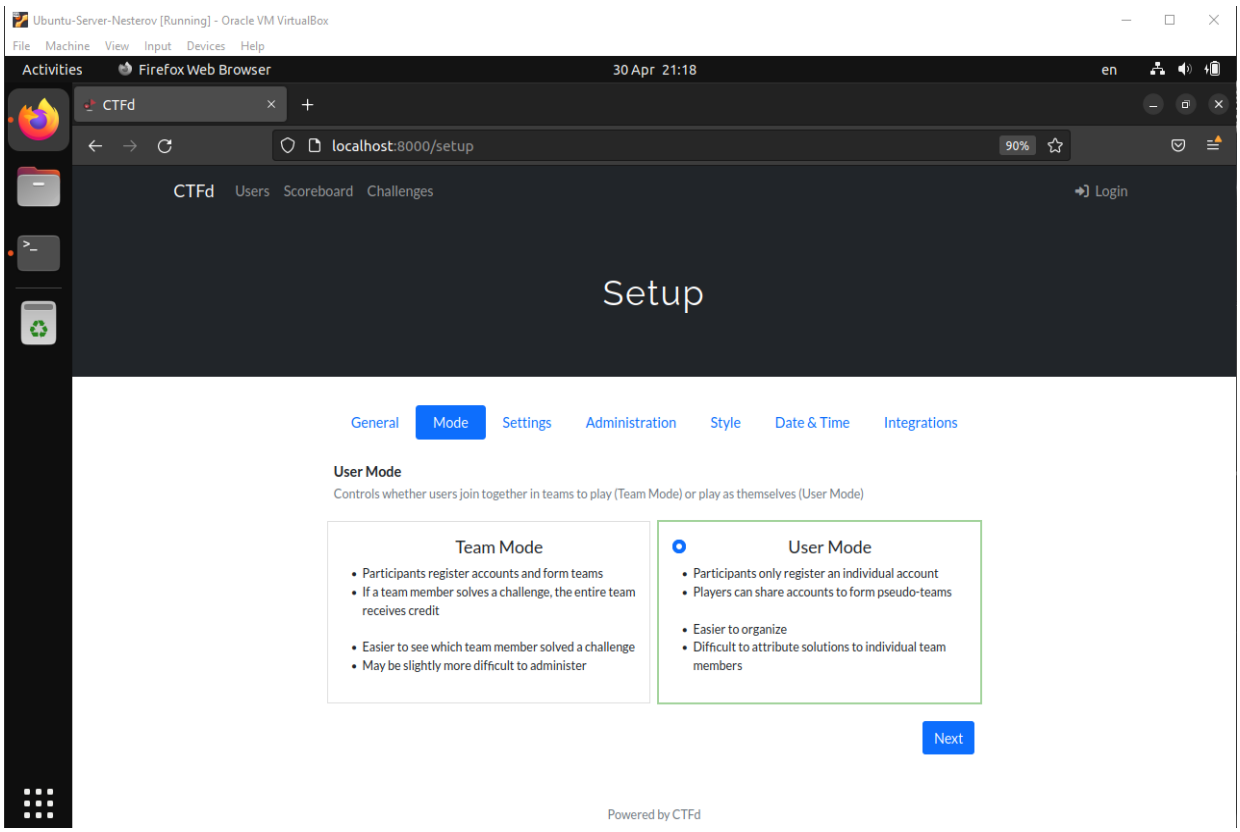


Рисунок 3.2.8 – Налаштування платформи на базі CTFd, вибір режиму User Mode

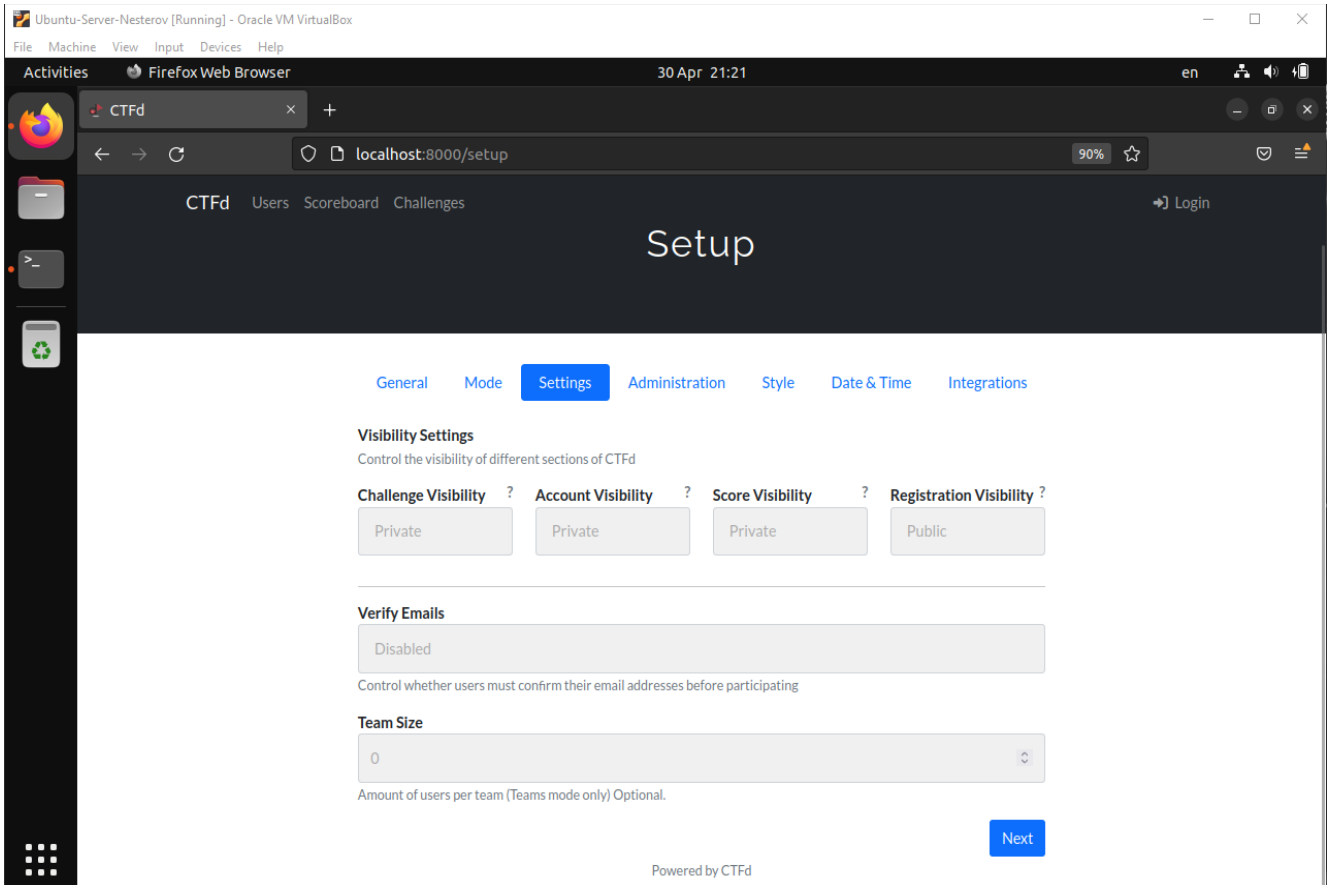


Рисунок 3.2.9 – Налаштування платформи на базі CTFd, налаштування видимості

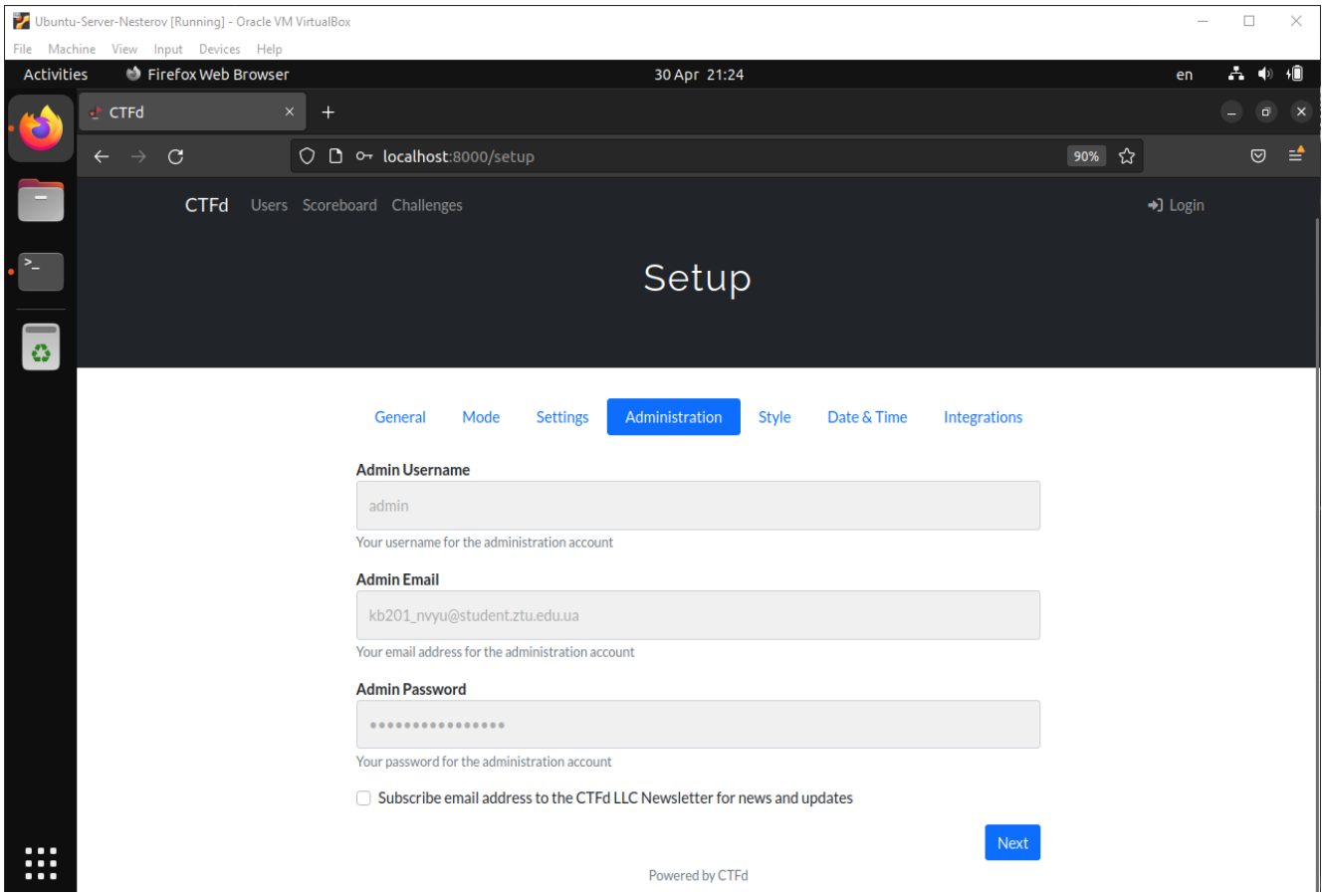


Рисунок 3.2.10 – Налаштування платформи на базі CTFd, створення адміна

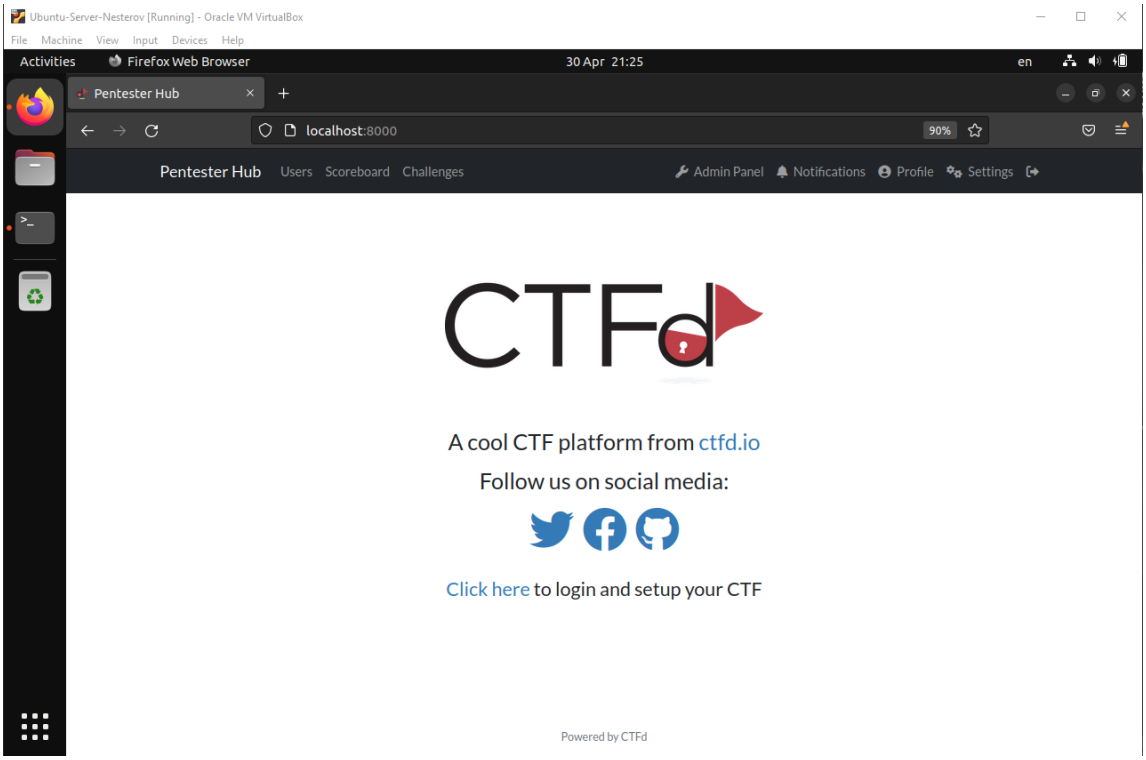


Рисунок 3.2.11 – Розгортання та налаштування платформи на базі CTFd успішно завершено

```
R-Internet#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
R-Internet(config)#ip host pentesterhub.net 200.9.13.2
R-Internet(config)#exit
```

Рисунок 3.2.12 – Створення DNS-запису для розгорнутої платформи на R-Internet

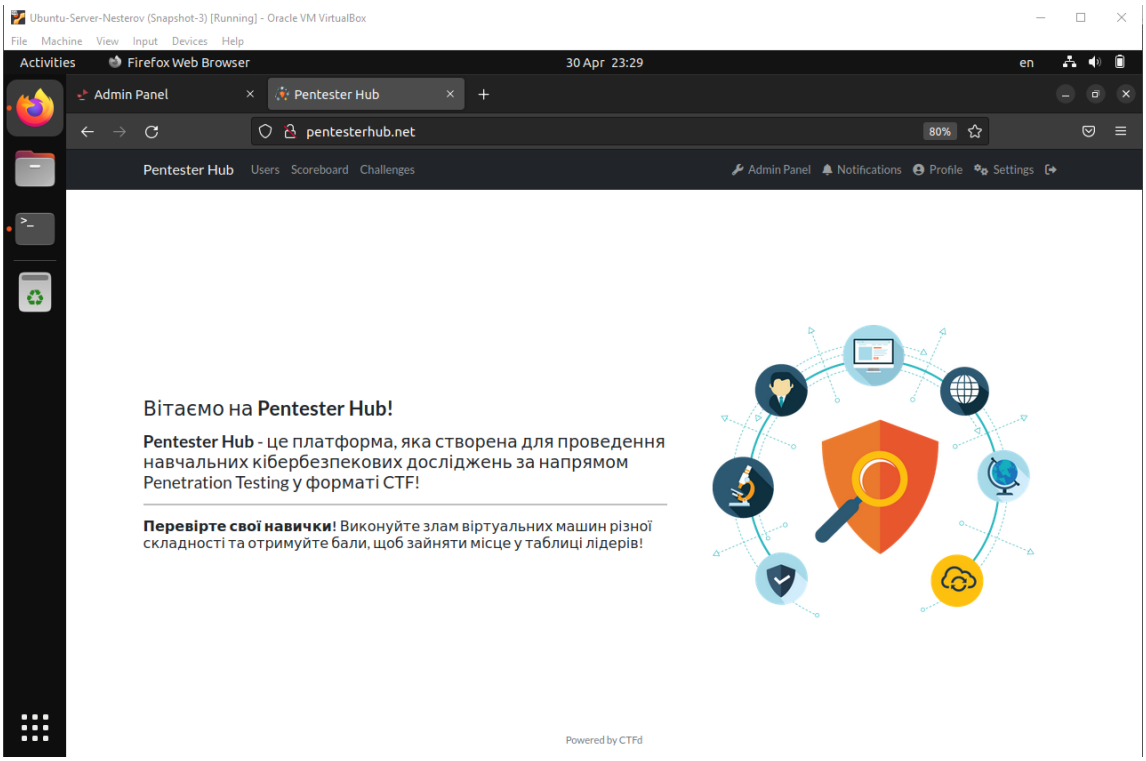


Рисунок 3.2.13 – Домене ім'я pentesterhub.net успішно функціонує

3.3 Встановлення та налаштування OpenVPN

Виконаємо встановлення OpenVPN на Server-9-13-1. Спочатку завантажуємо скрипт для встановлення OpenVPN з репозиторію.

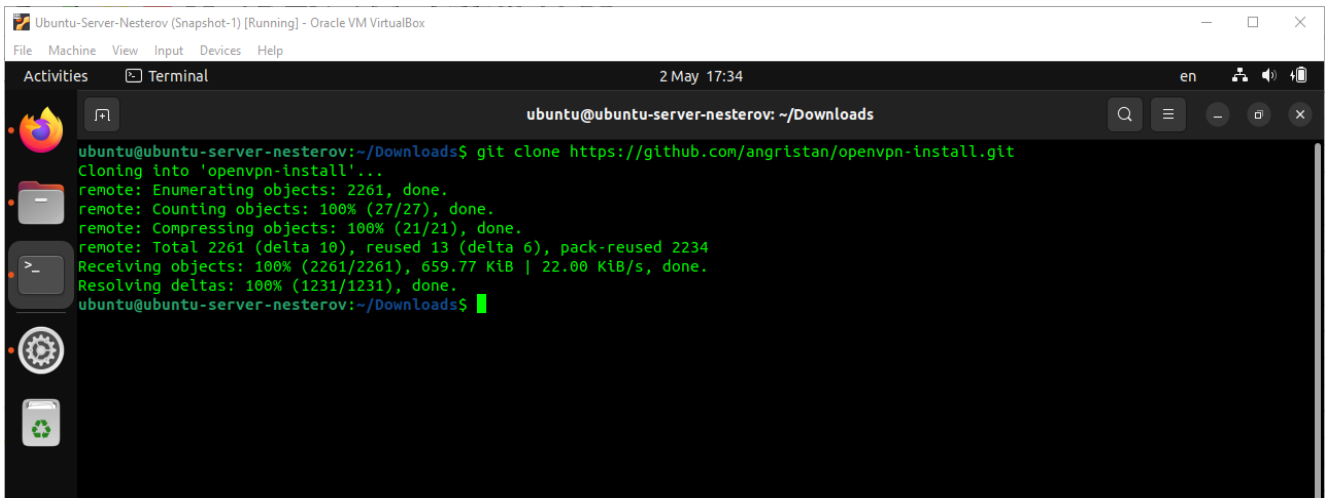


Рисунок 3.3.1 – Завантаження скрипта openvpn-install.sh

Скрипт для встановлення OpenVPN успішно завантажено, наступним кроком є перехід в директорію openvpn-install та виконання скрипта за допомогою команди sudo bash openvpn-install.sh.

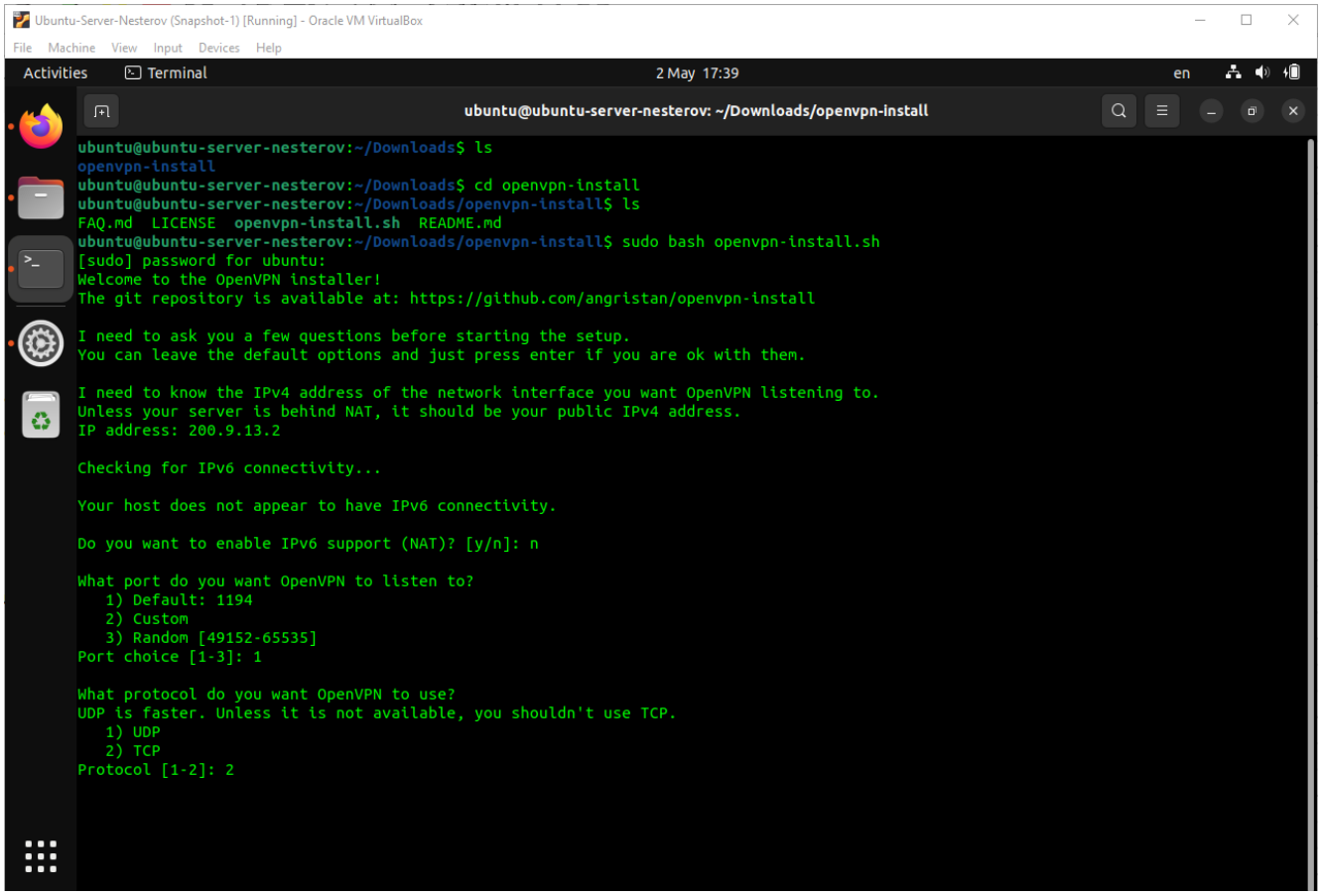


Рисунок 3.3.2 – Встановлення OpenVPN

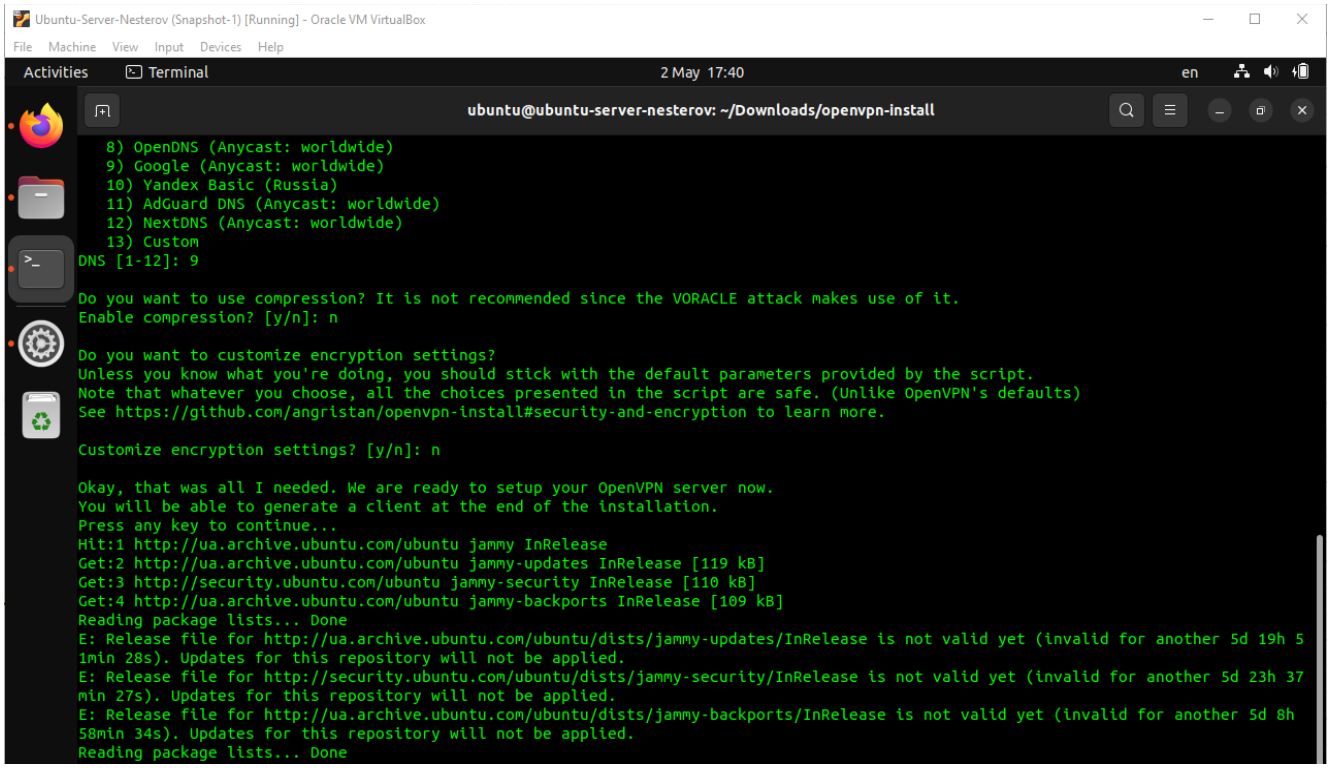


Рисунок 3.3.3 – Встановлення OpenVPN

OpenVPN успішно встановлено, виконаємо створення конфігураційного файлу .ovpn.

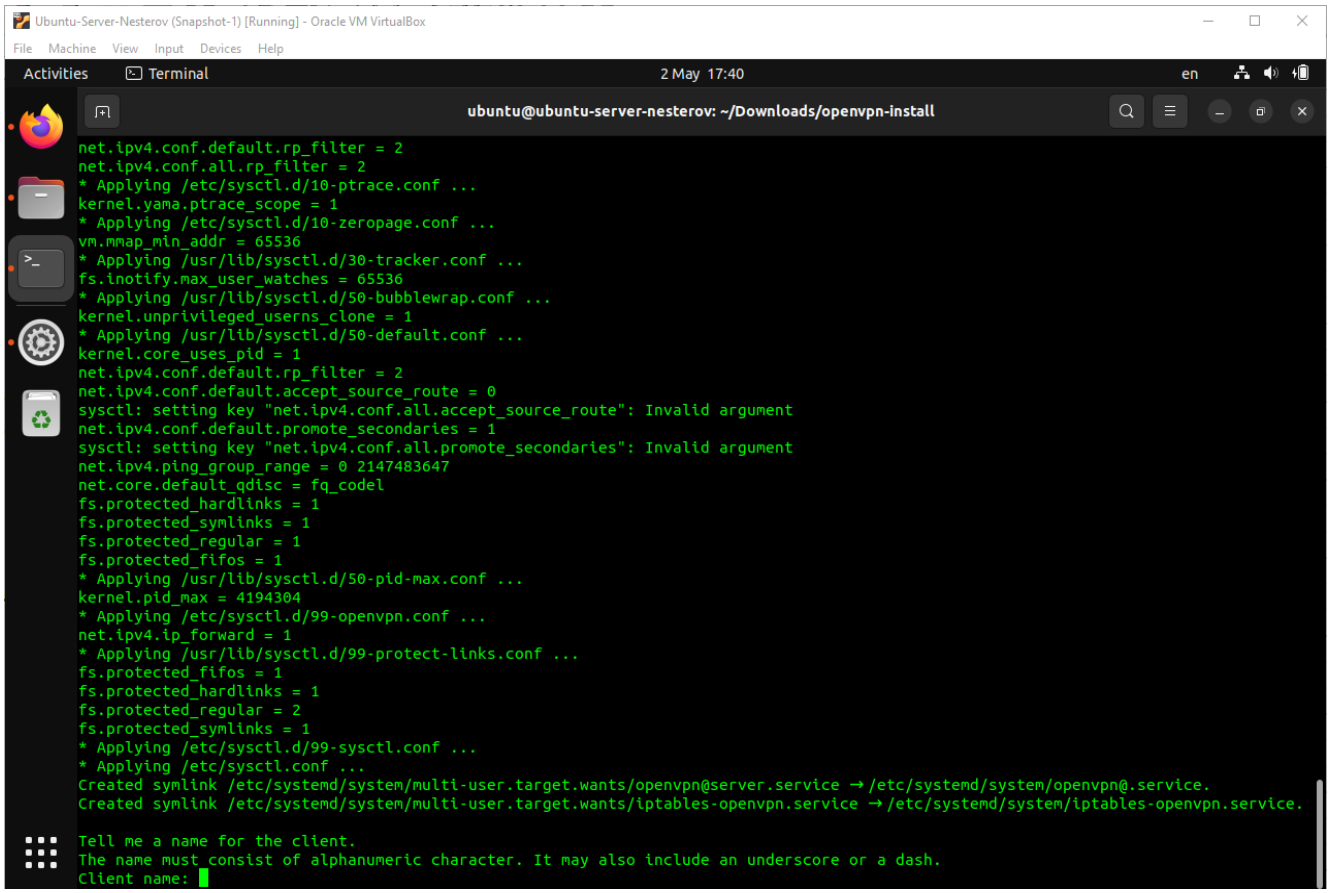


Рисунок 3.3.4 – Створення конфігураційного файлу .ovpn

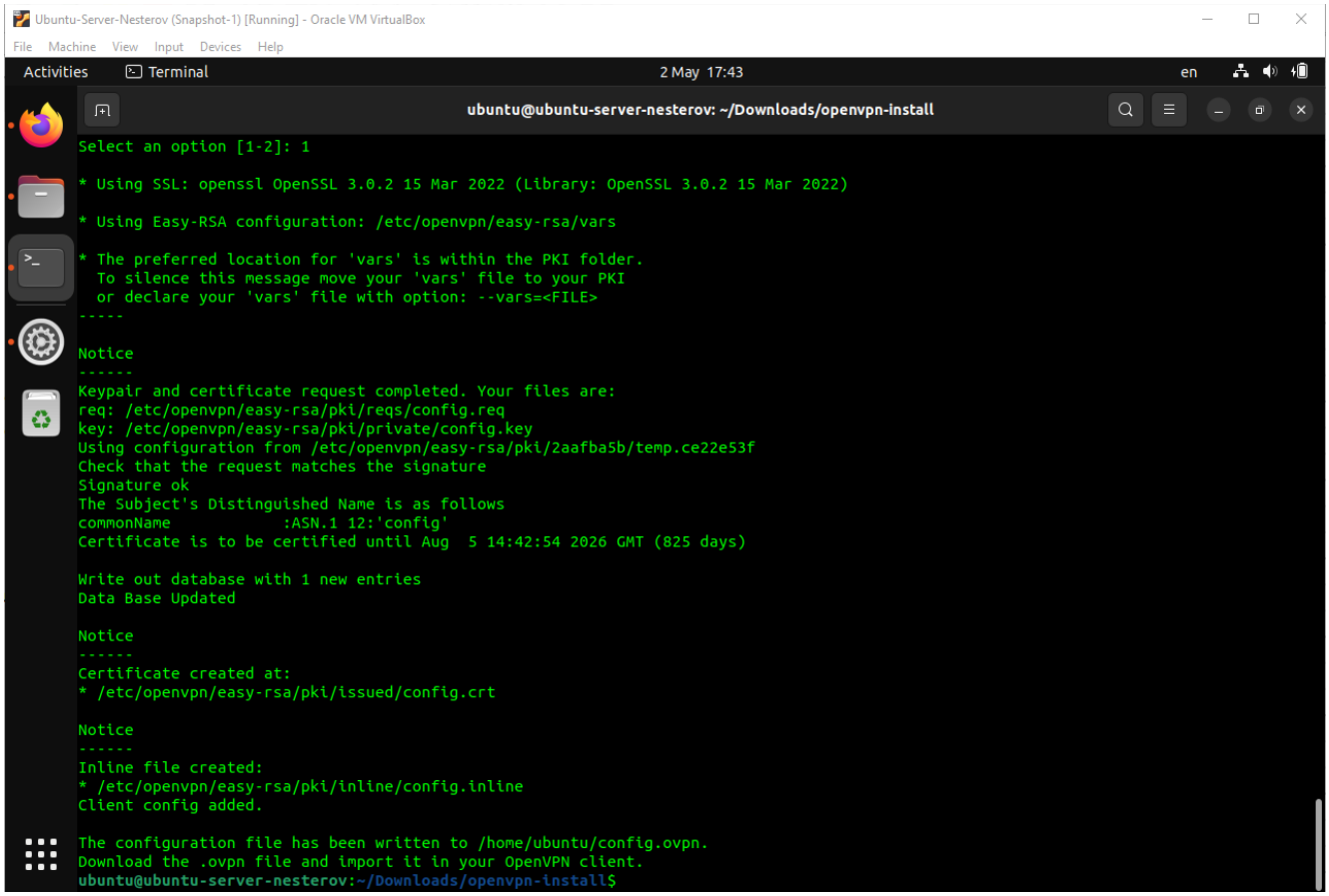


Рисунок 3.3.5 – Створення конфігураційного файлу .ovpn

Конфігураційний файл .ovpn успішно створено, наступним кроком є налаштування OpenVPN як VPN-сервера, переходимо в директорію /etc/openvpn та виконуємо редагування конфігураційного файлу server.conf.

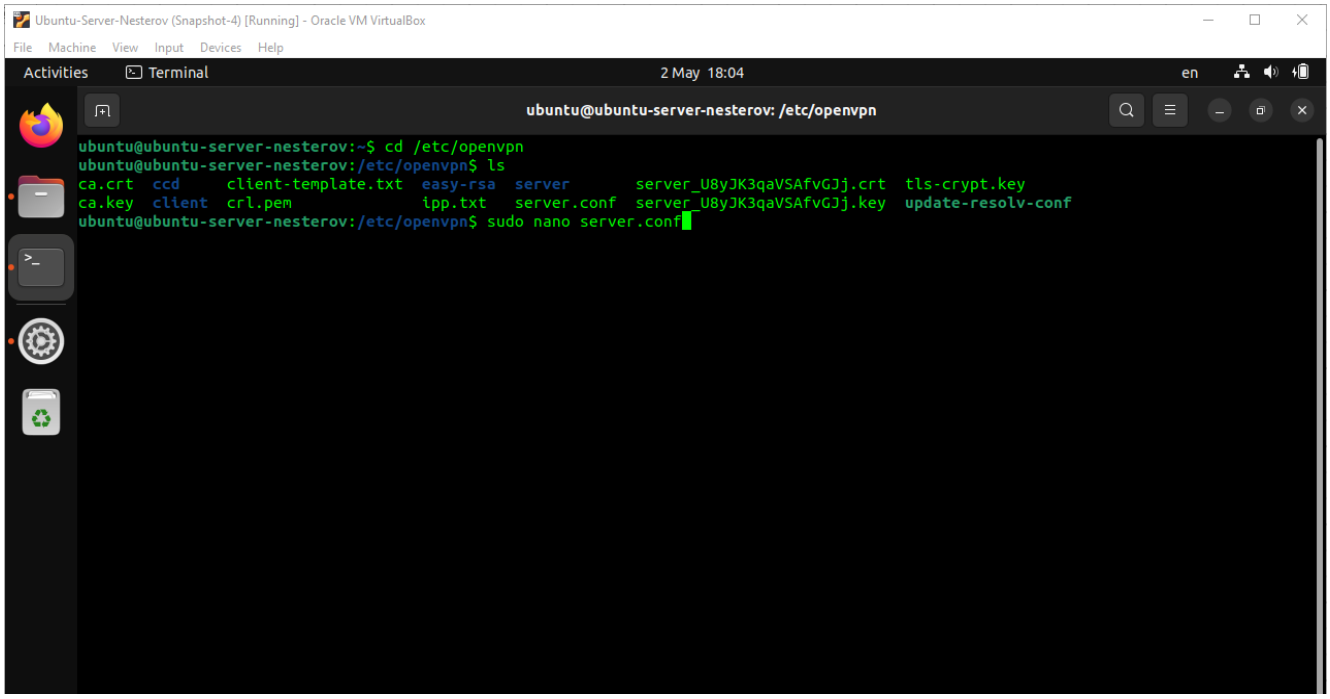


Рисунок 3.3.6 – Виконання команди sudo nano server.conf

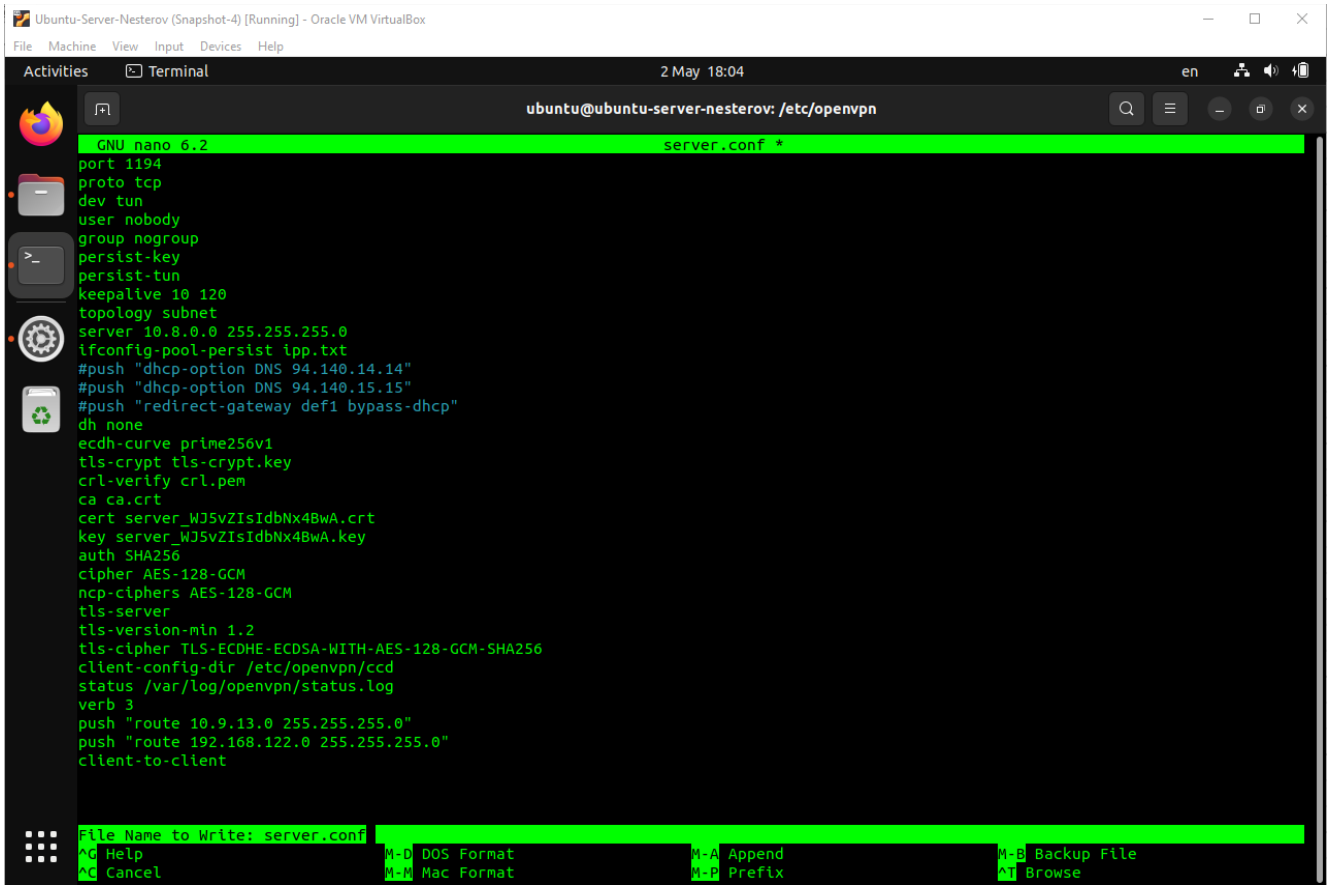


Рисунок 3.3.7 – Редагування конфігураційного файлу server.conf

Після встановлення та налаштування OpenVPN на Server-9-13-1 повинен створитися інтерфейс tun0.

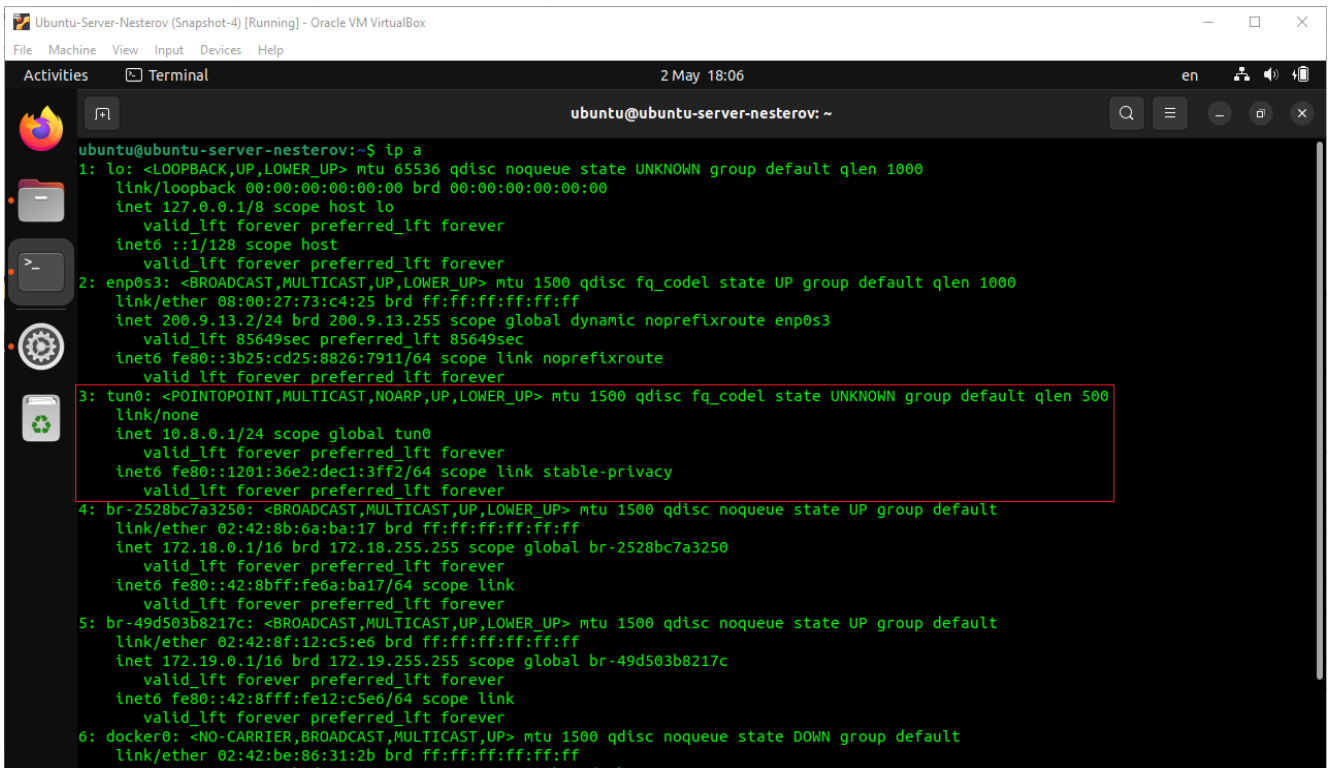


Рисунок 3.3.8 – Інтерфейс tun0 успішно створено

Дозволимо Forwarding на Server-9-13-1, щоб перенаправляти трафік з клієнтів на VM-жертви використовуючи OpenVPN сервер за допомогою команди sudo ufw default allow FORWARD.

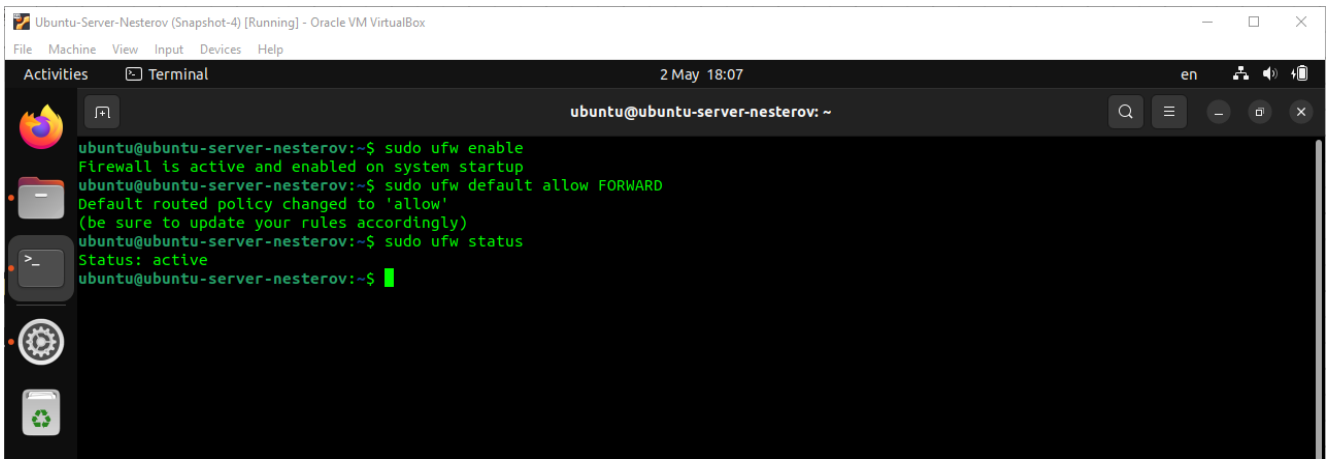


Рисунок 3.3.9 – Дозволяємо перенаправляти трафік з клієнтів на VM-жертви

3.4 Наповнення платформи завданнями

Наступним кроком є створення серій завдань для кожної VM-жертви.

Машина SolidState.

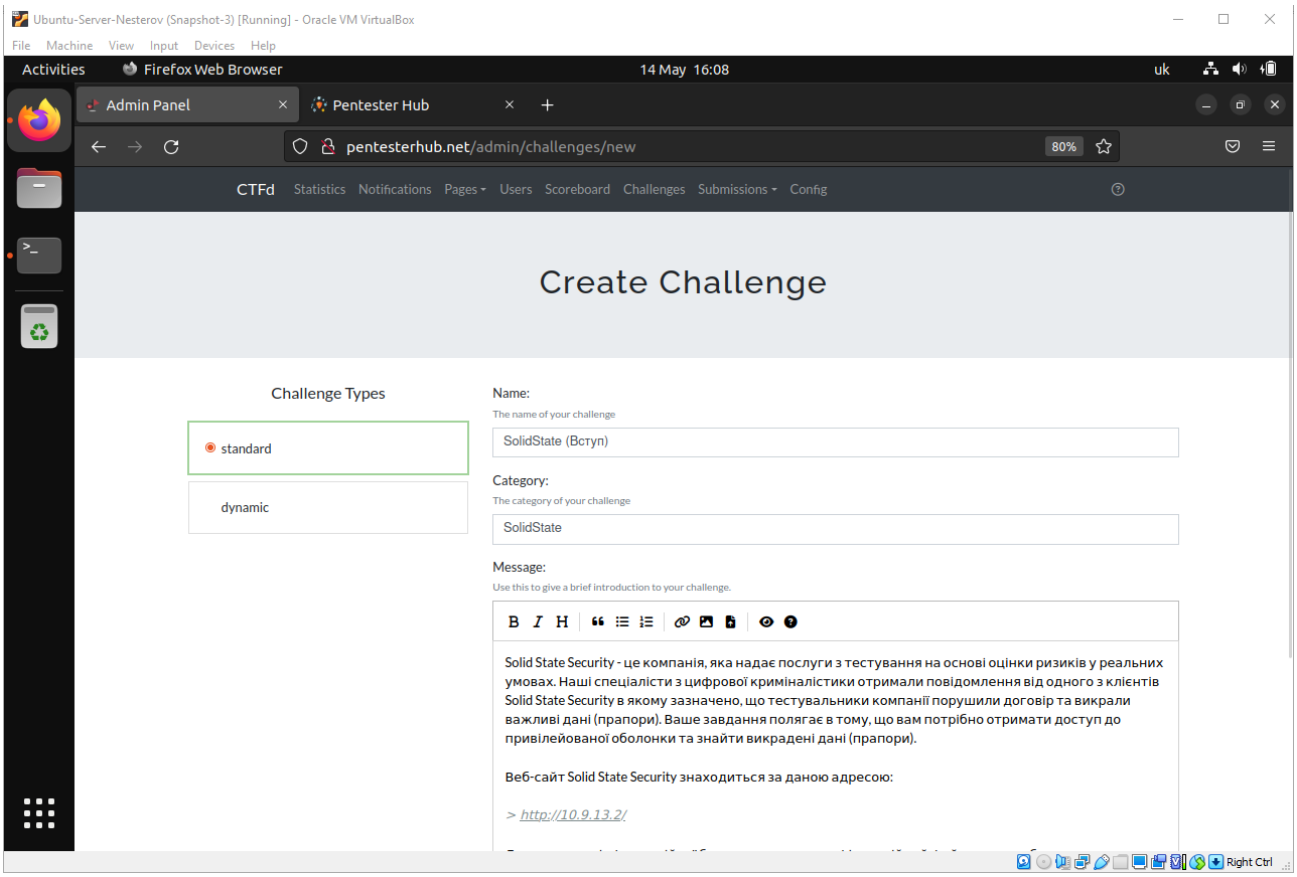


Рисунок 3.4.1 – Створення першого завдання в серії SolidState (Вступ)

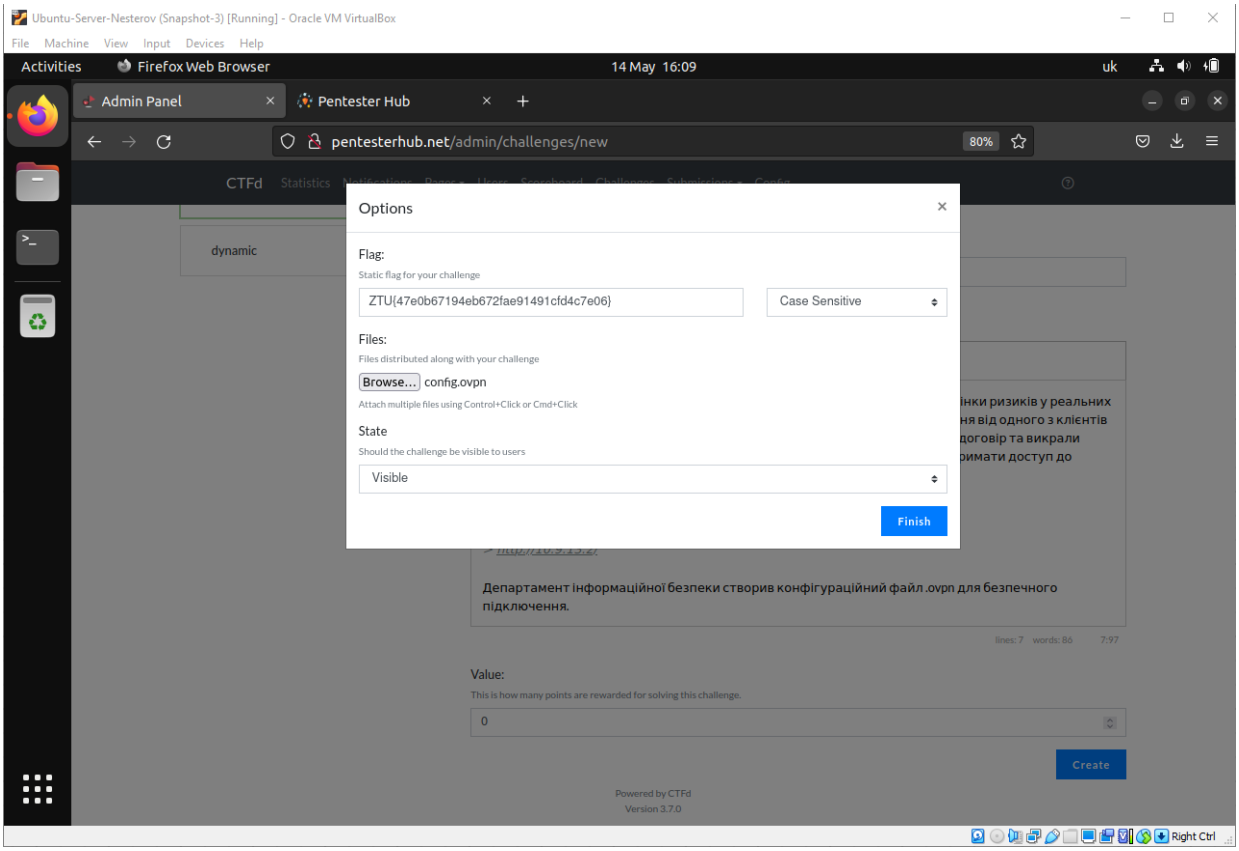


Рисунок 3.4.2 – Задання прапора для виконання завдання та завантаження конфігураційного файлу .ovpn

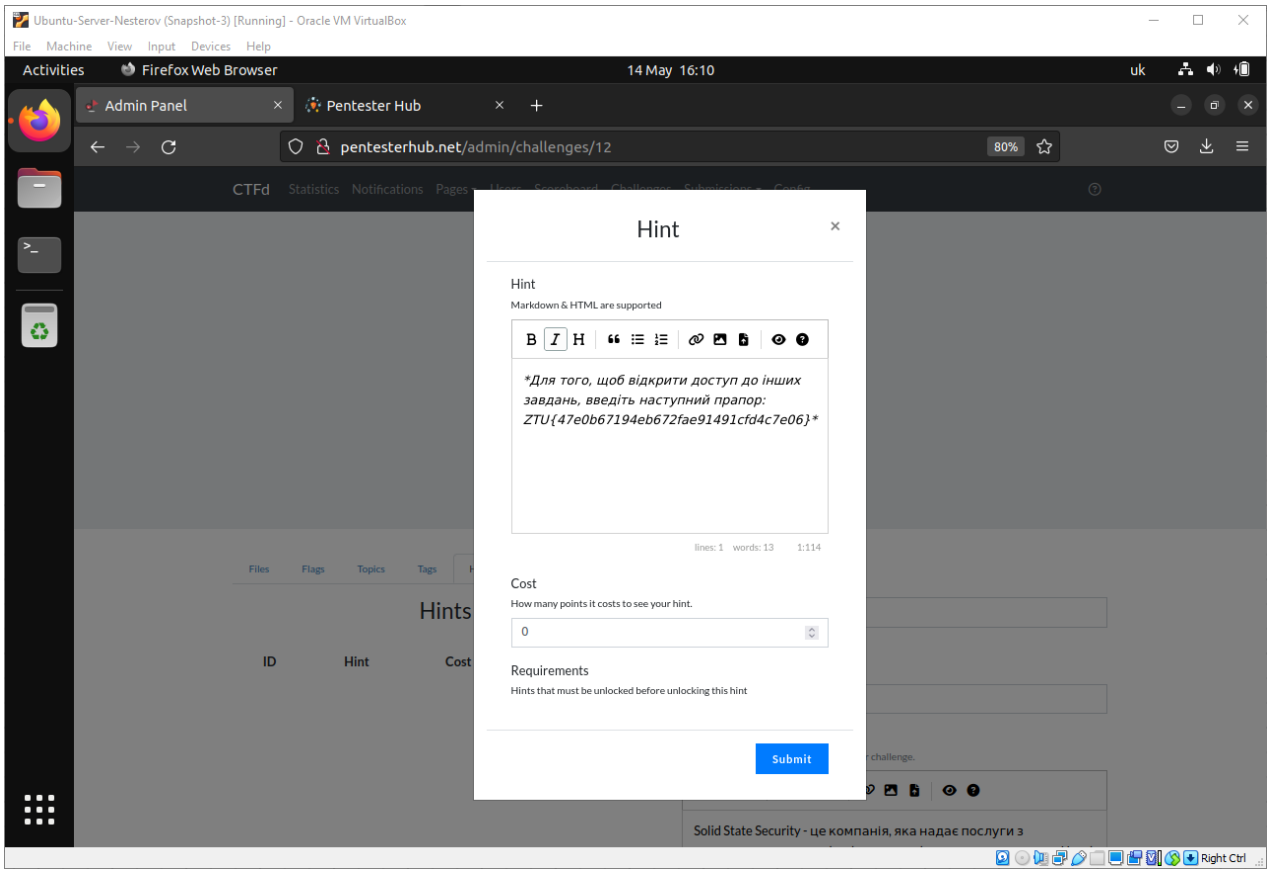


Рисунок 3.4.3 – Створення безкоштовної підказки для завдання

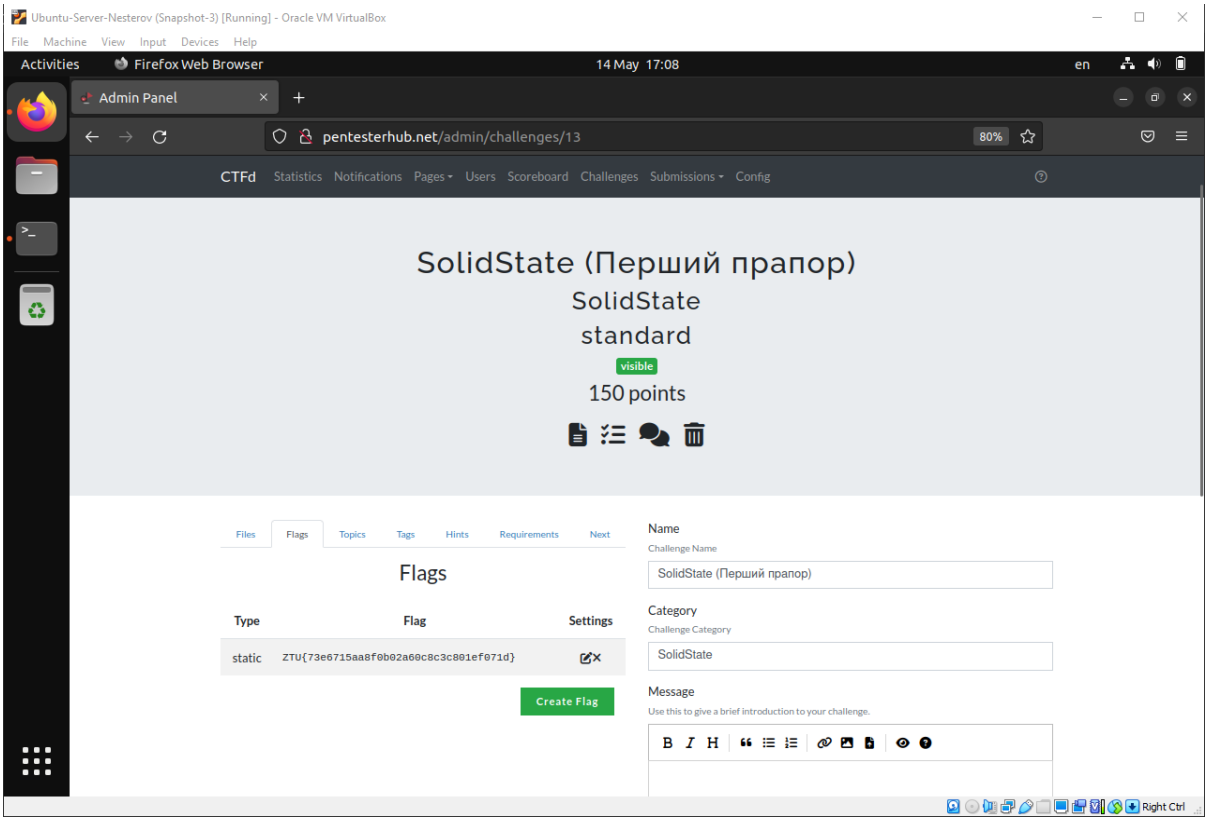


Рисунок 3.4.4 – Створення другого завдання в серії SolidState (Перший прапор), задання прапора для виконання завдання

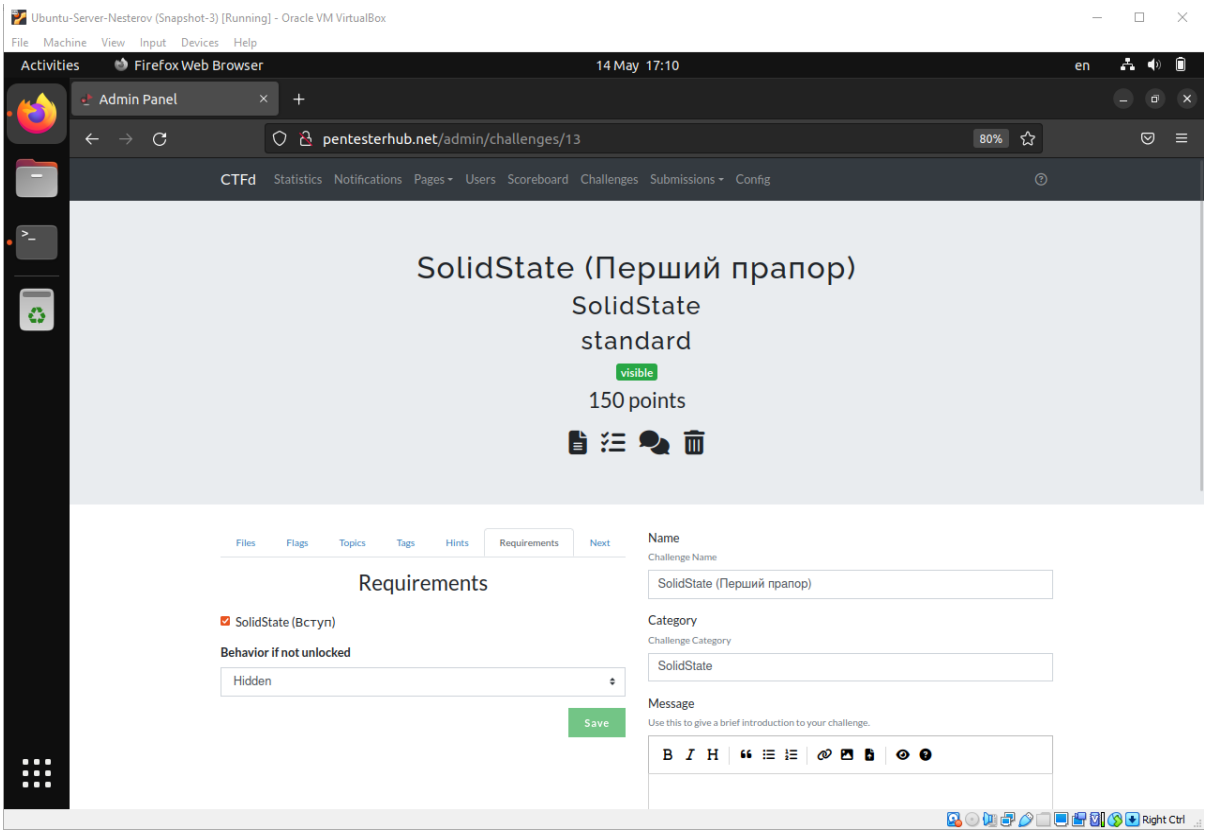


Рисунок 3.4.5 – Створення вимоги, що дане завдання стане доступним лише після виконання SolidState (Вступ)

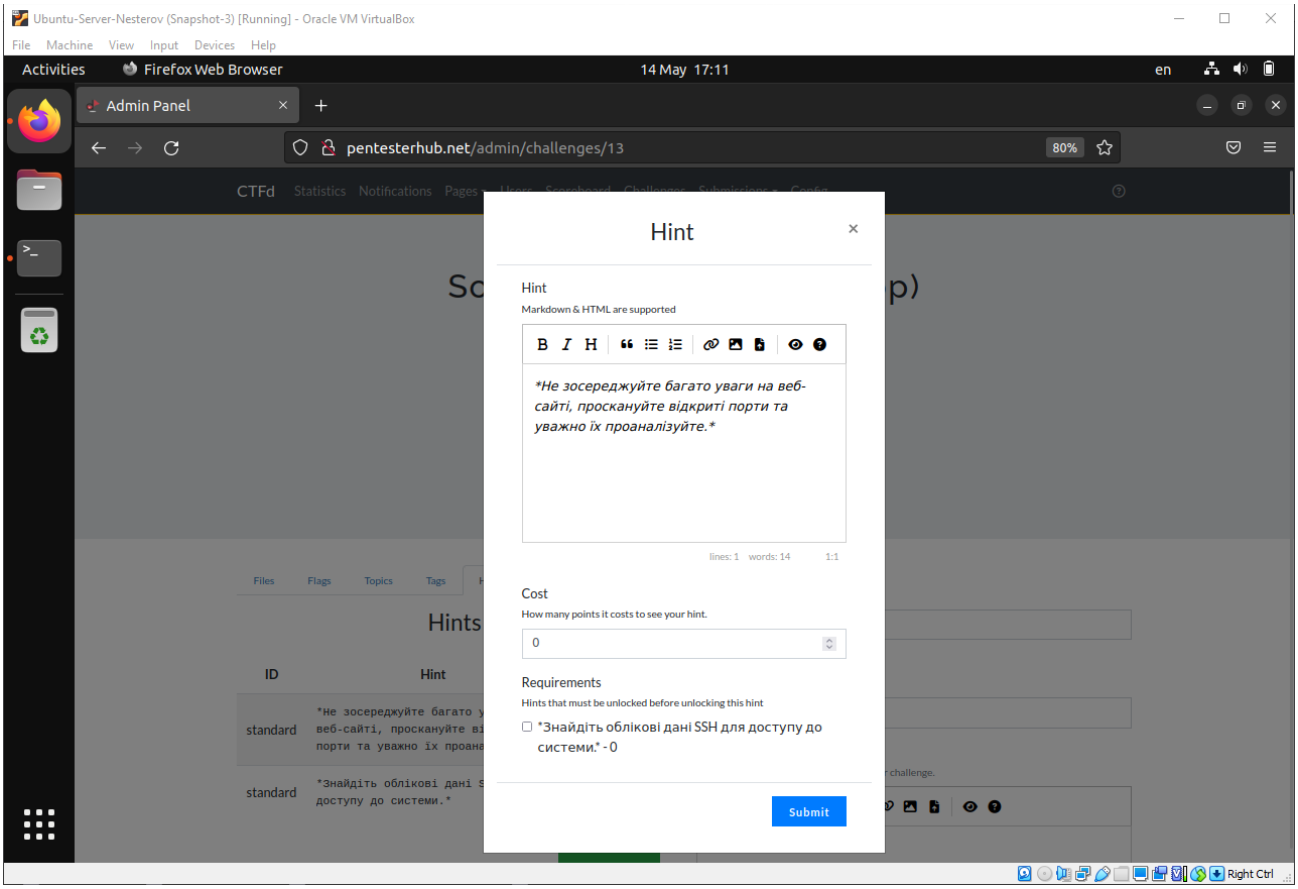


Рисунок 3.4.6 – Створення першої безкоштовної підказки для завдання

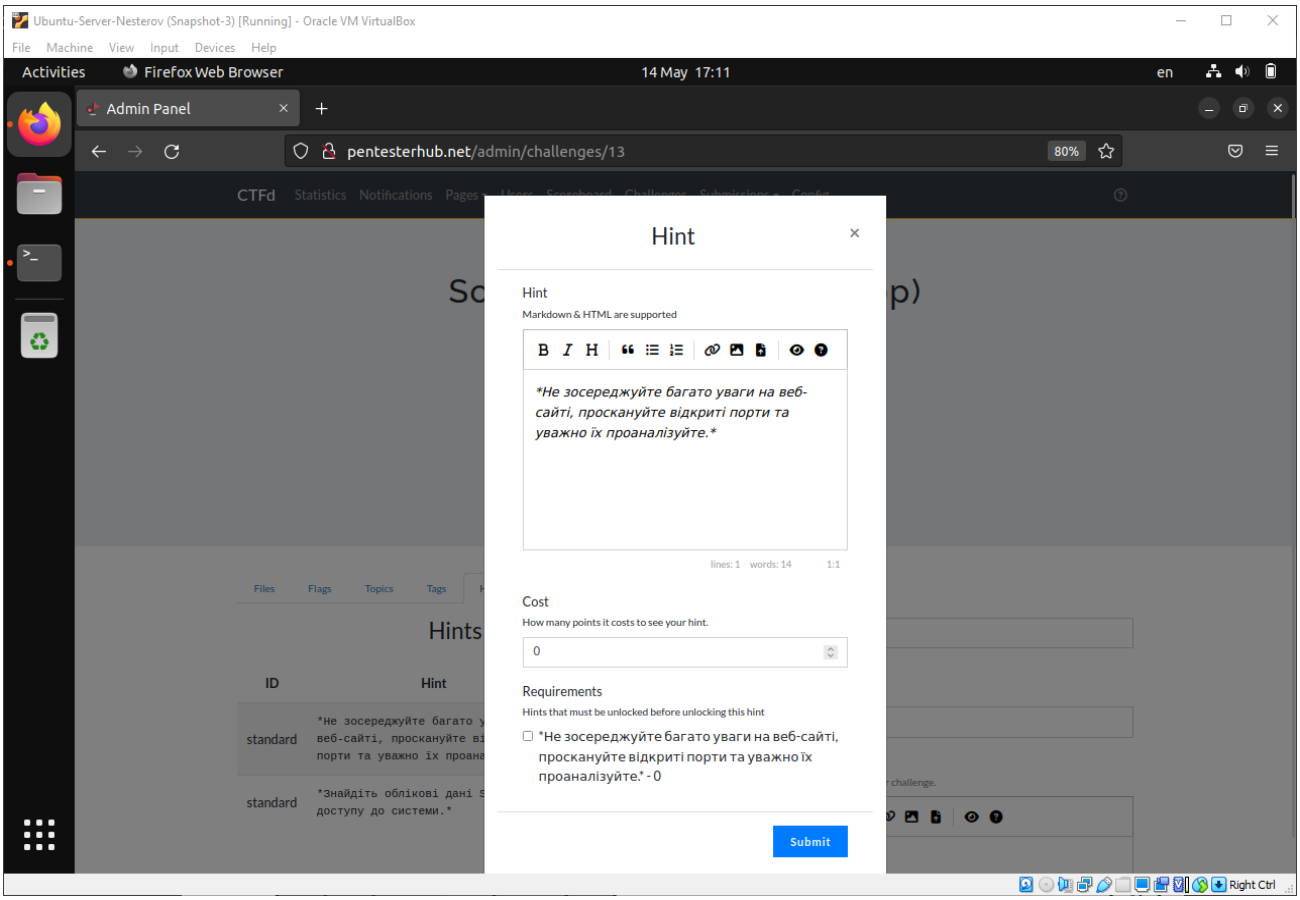


Рисунок 3.4.7 – Створення другої безкоштовної підказки для завдання

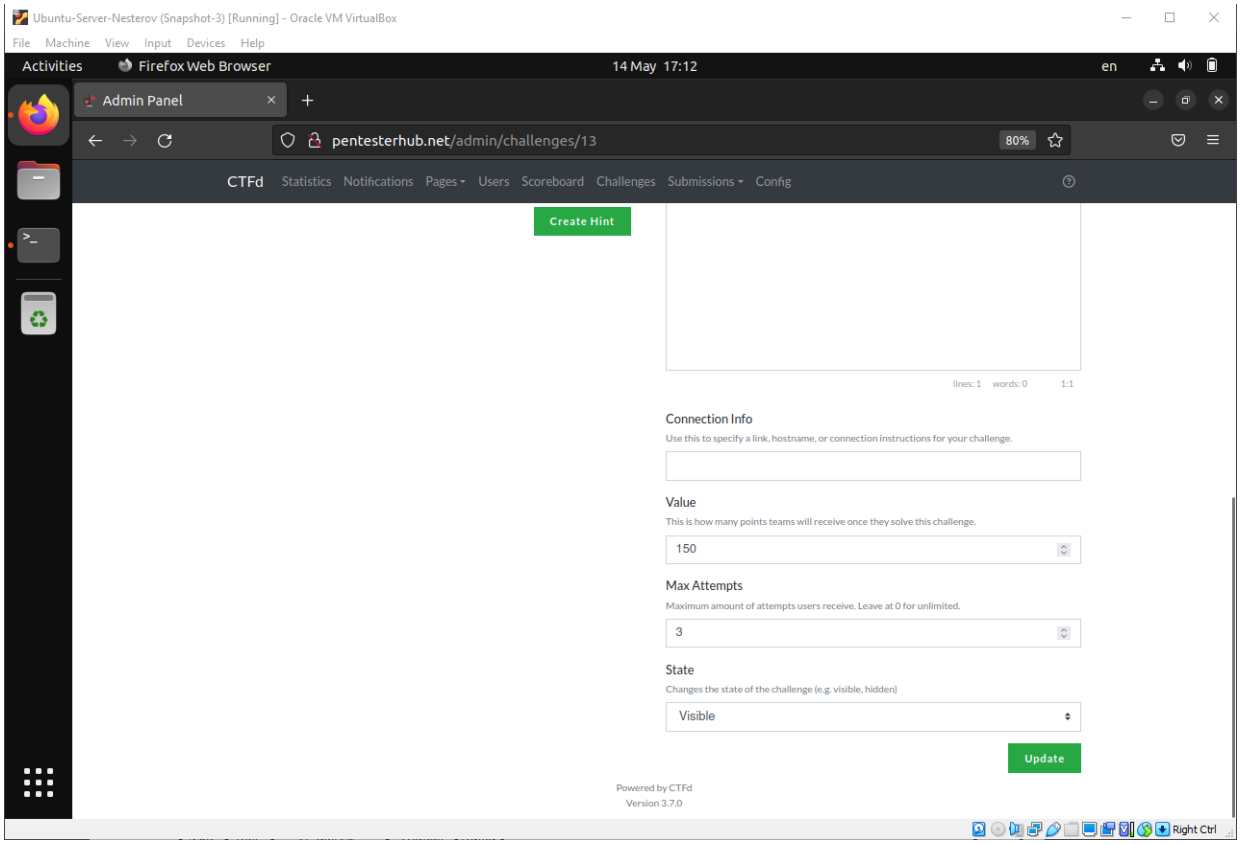


Рисунок 3.4.8 – Встановлення обмеження в максимальну кількість спроб

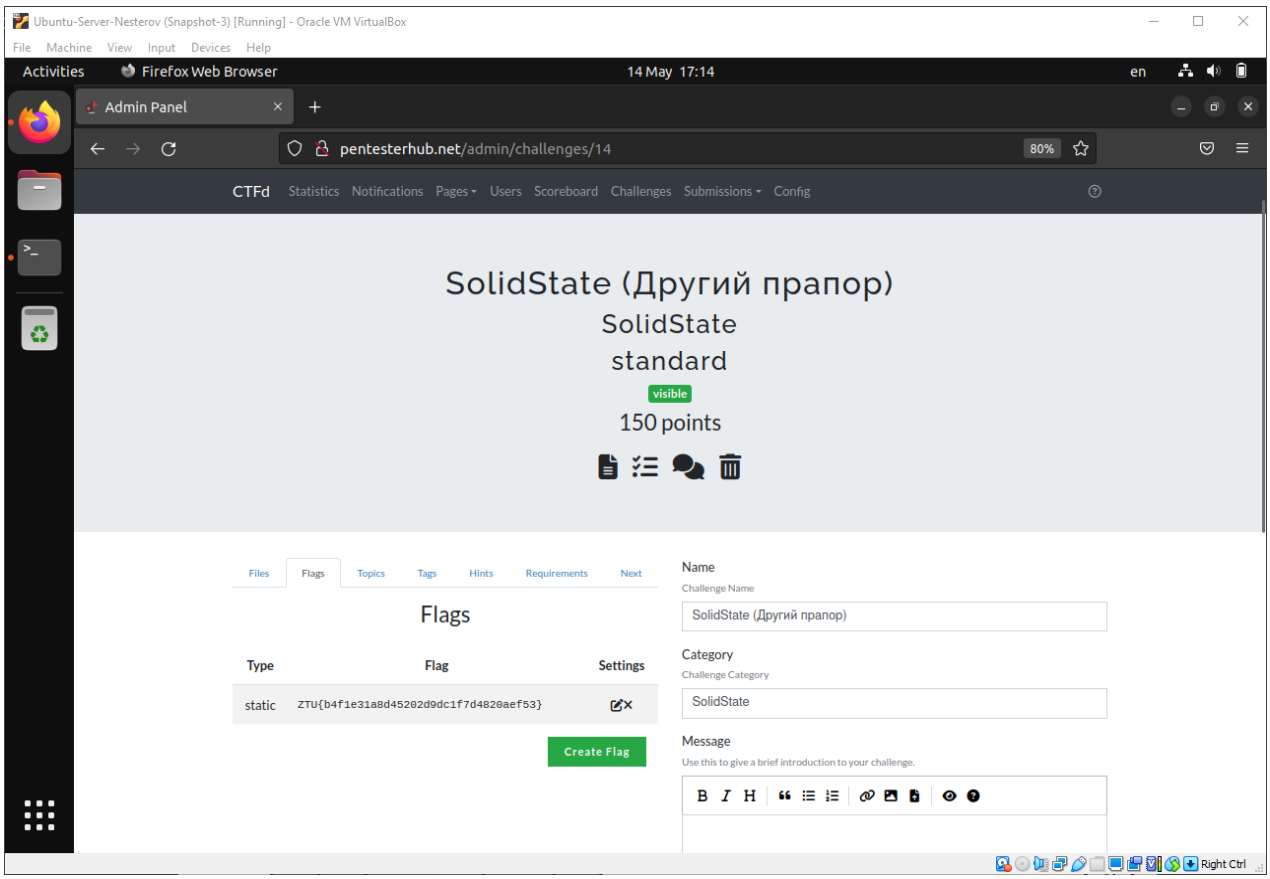


Рисунок 3.4.9 – Створення третього/фінального завдання в серії SolidState (Другий прапор), задання прапора для виконання завдання

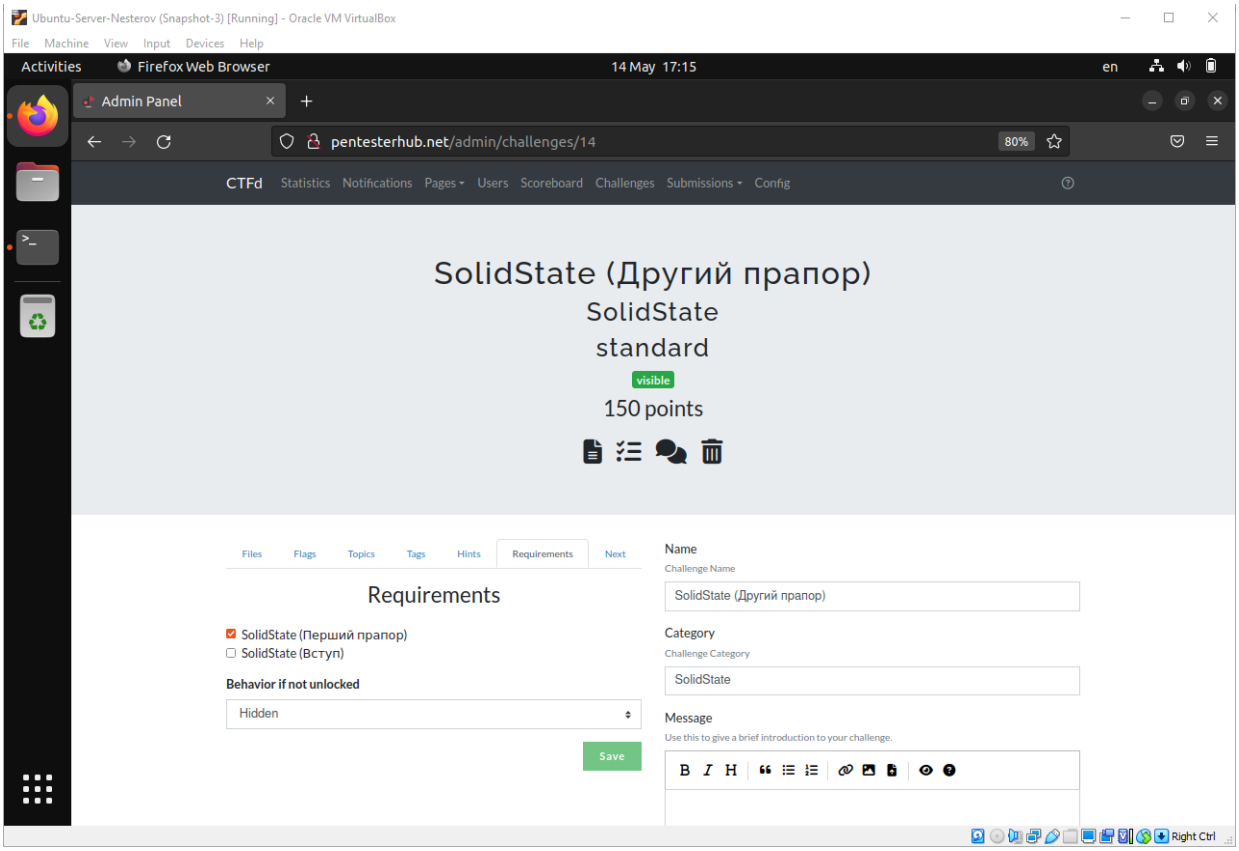


Рисунок 3.4.10 – Створення вимоги, що дане завдання стане доступним після виконання SolidState (Перший прапор)

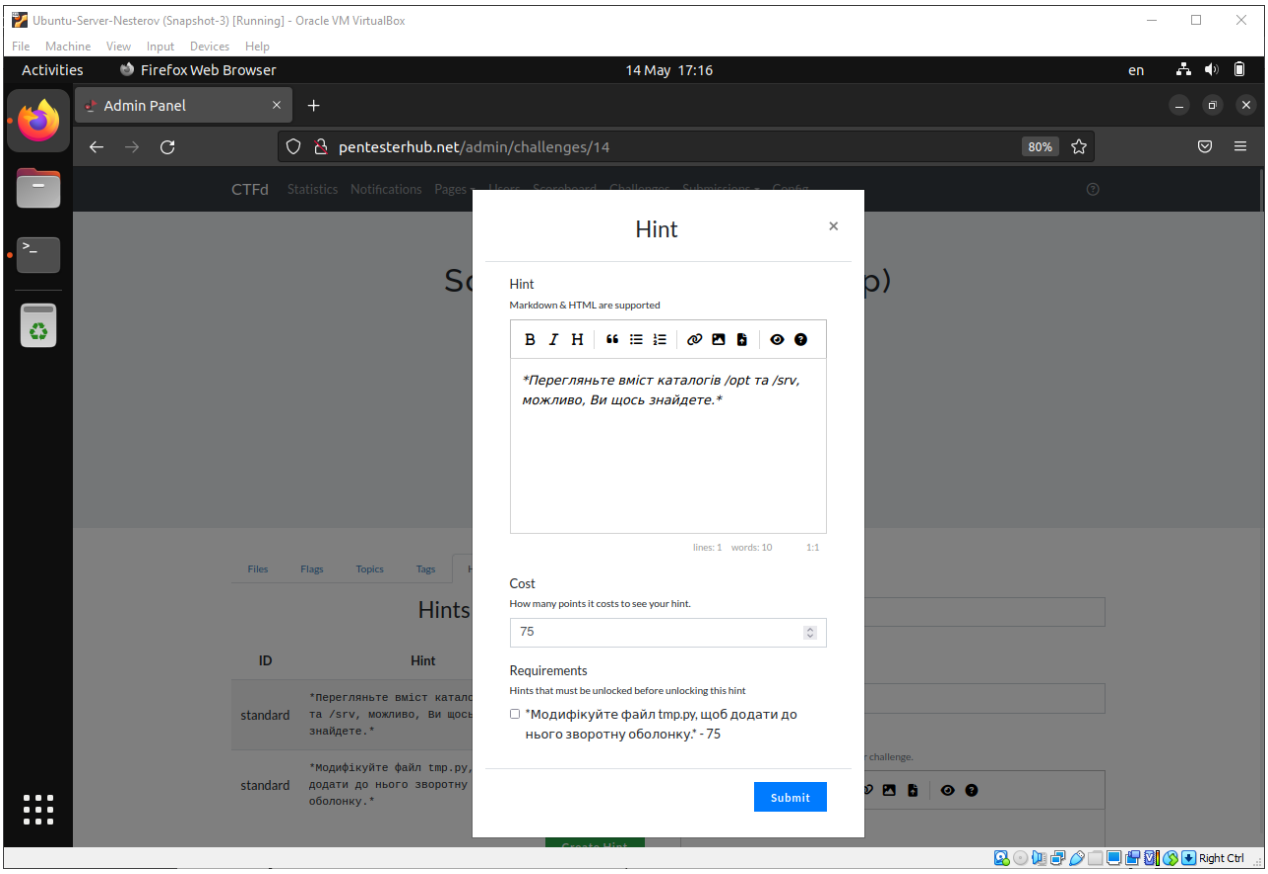


Рисунок 3.4.11 – Створення першої підказки для завдання, яка коштує 75 балів

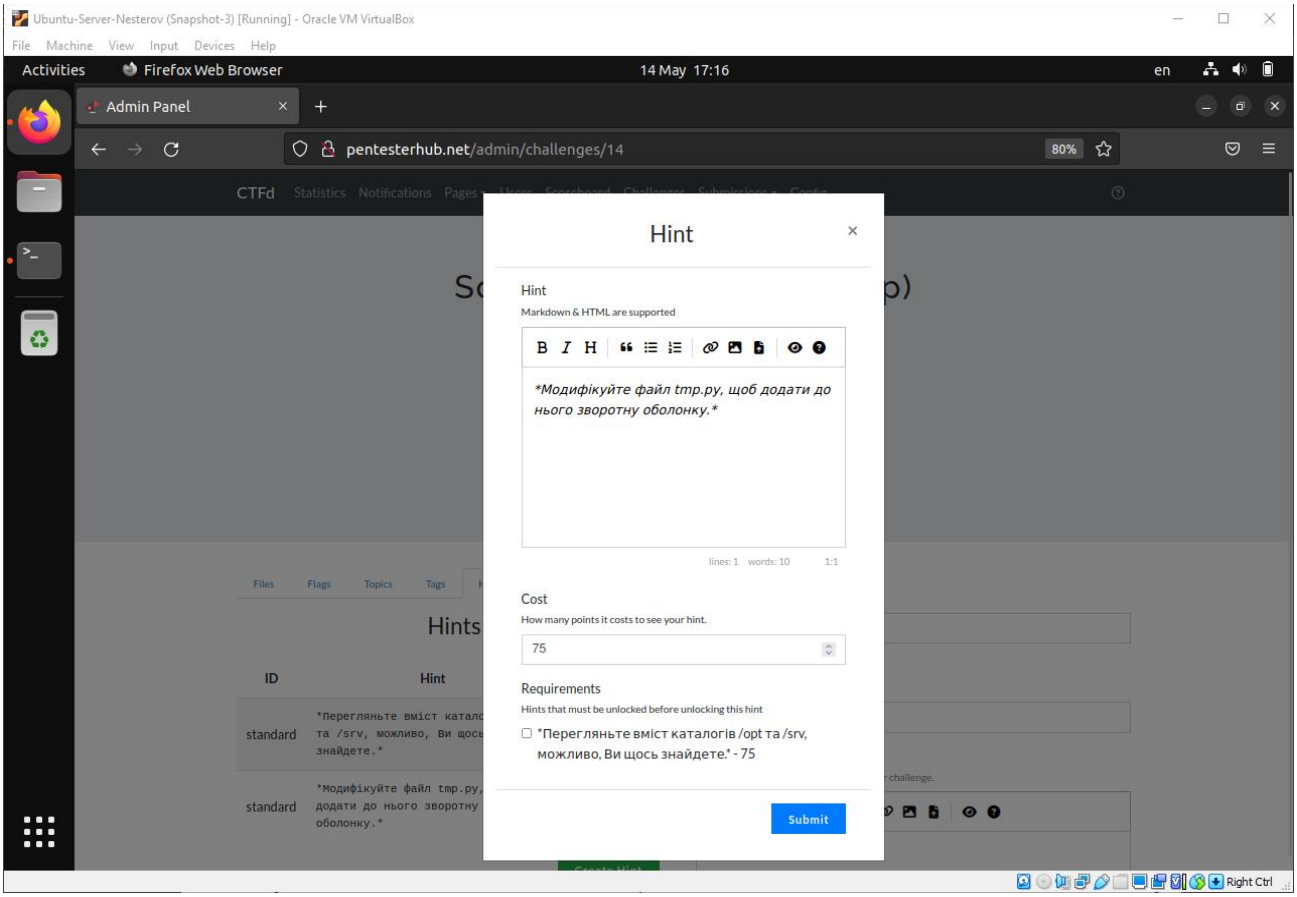


Рисунок 3.4.12 – Створення другої підказки для завдання, яка коштує 75 балів

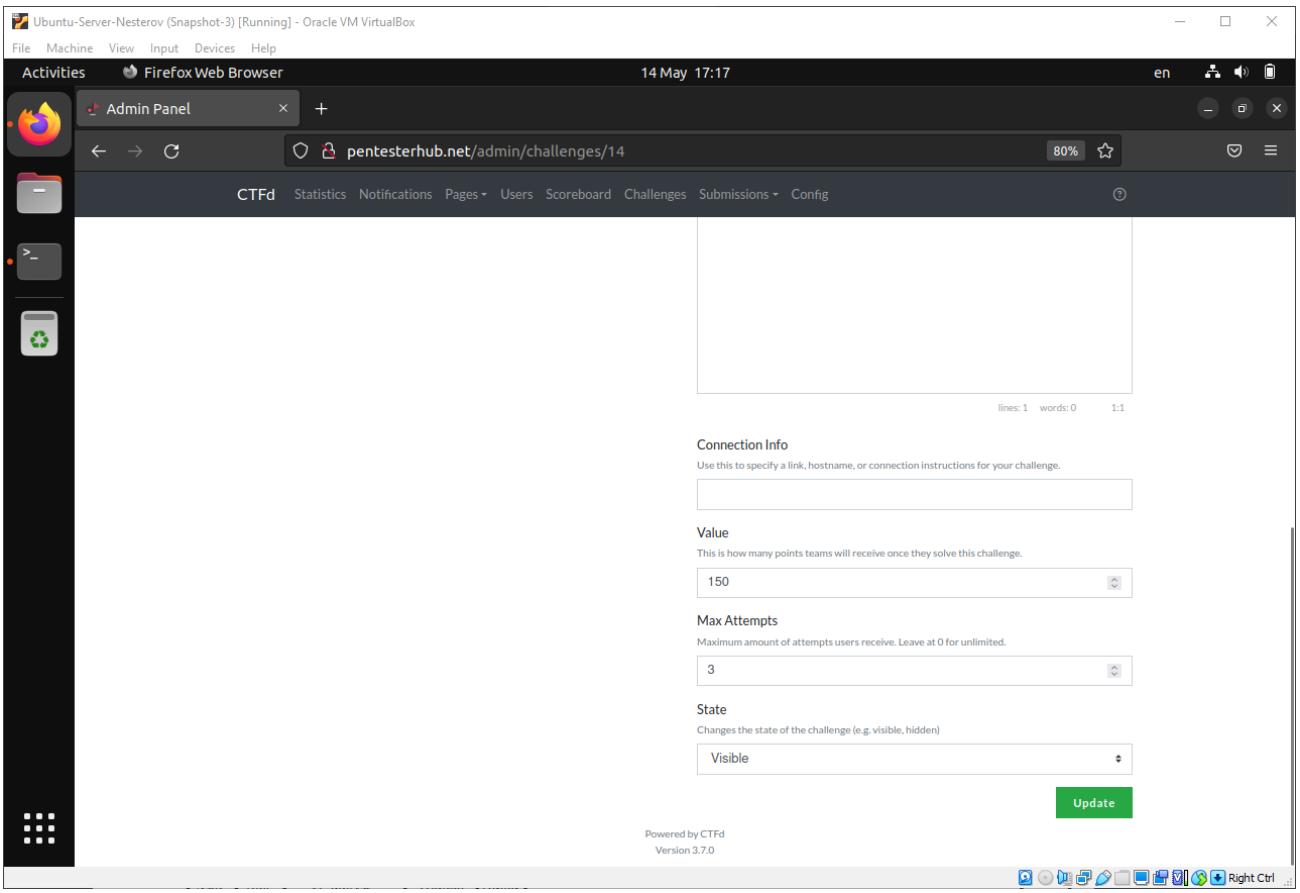


Рисунок 3.4.13 – Встановлення обмеження в максимальну кількість спроб

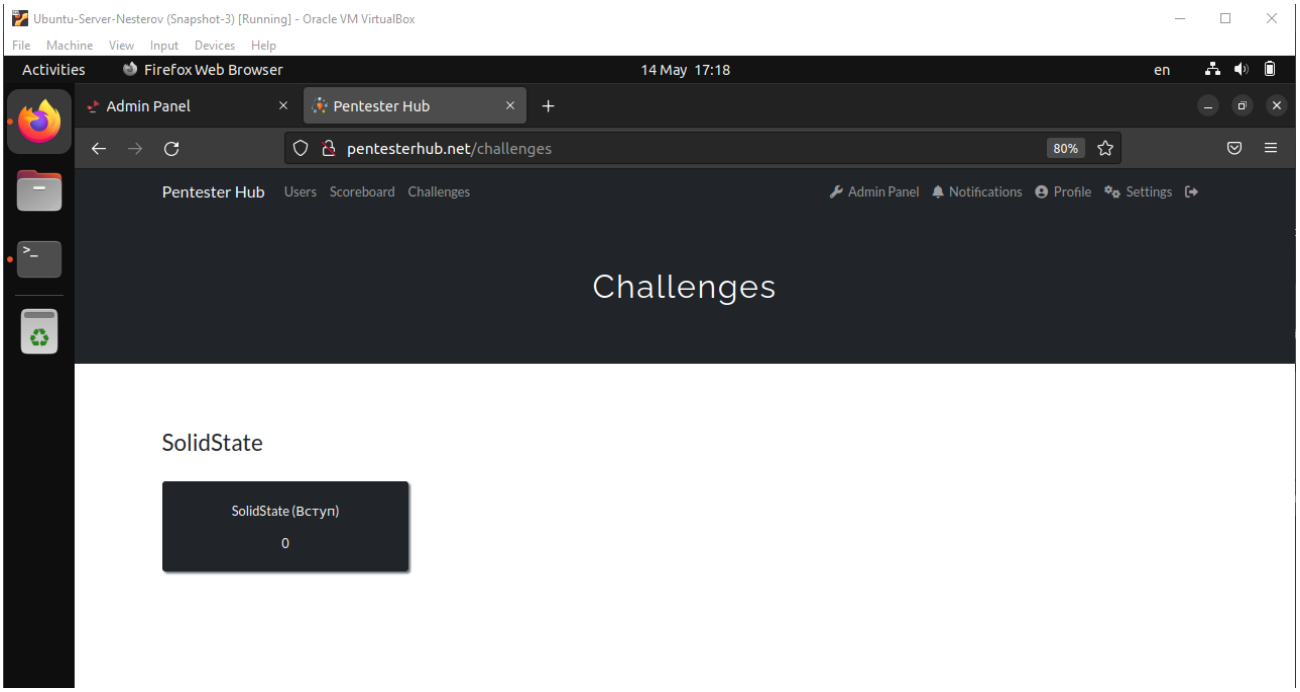


Рисунок 3.4.14 – Серія завдань для машини SolidState успішно створена

Створення серій завдань для машин WebDeveloper, Straylight, zico2 та DC-1 проводяться аналогічно.

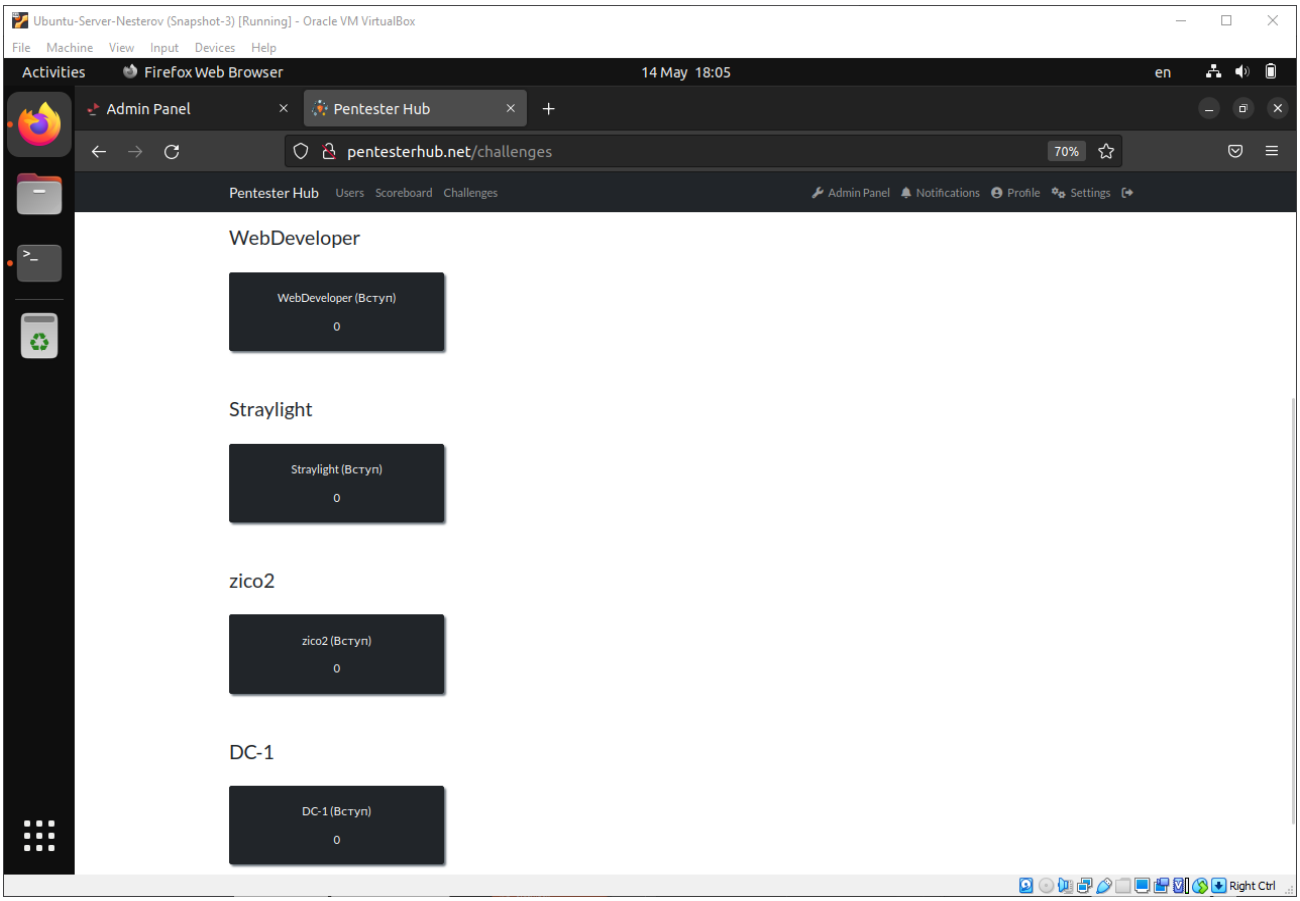


Рисунок 3.4.15 – Серії завдань для машин WebDeveloper, Straylight, zico2 та DC-1 успішно створені

3.5 Тестування роботи мережі та функціоналу платформи

Розпочнемо з тестування встановлених налаштувань в мережі.

```
R-Clients-9-13-1#ping cisco.com

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 72.163.4.185, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 192/192/196 ms
R-Clients-9-13-1#ping 200.9.13.1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 200.9.13.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 40/46/64 ms
```

Рисунок 3.5.1 – Перевірка доступу до Інтернету та наявності зв'язку між підмережами LAN-Clients та LAN-Platform на R-Clients-9-13-1

```
R-Platform-9-13-1#ping 10.9.13.2

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.9.13.2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 8/23/64 ms
R-Platform-9-13-1#ping 10.9.13.3

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.9.13.3, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 8/15/32 ms
R-Platform-9-13-1#ping 10.9.13.4

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.9.13.4, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 4/13/28 ms
R-Platform-9-13-1#ping 10.9.13.5

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.9.13.5, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 12/16/28 ms
R-Platform-9-13-1#ping 10.9.13.6

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.9.13.6, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 4/15/28 ms
```

Рисунок 3.5.2 – Перевірка наявності зв'язку всередині LAN VMs

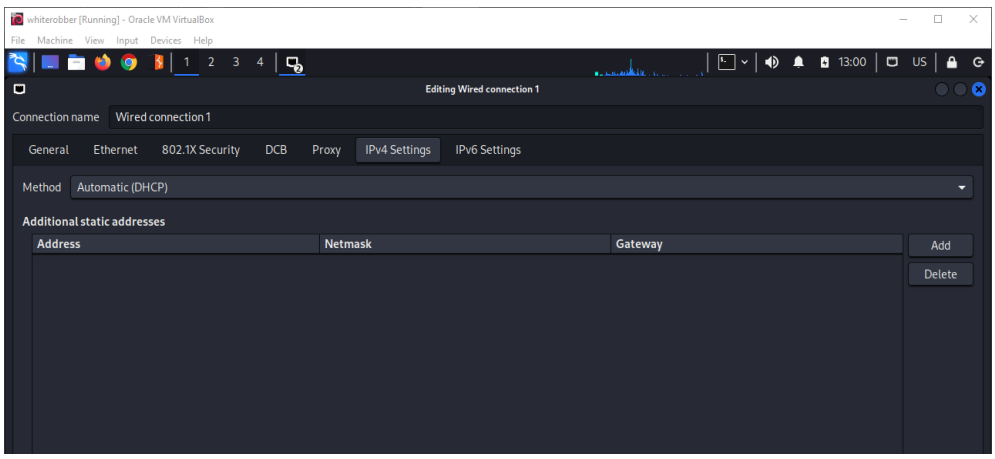


Рисунок 3.5.3 – Отримання DHCP на WS-9-13-1

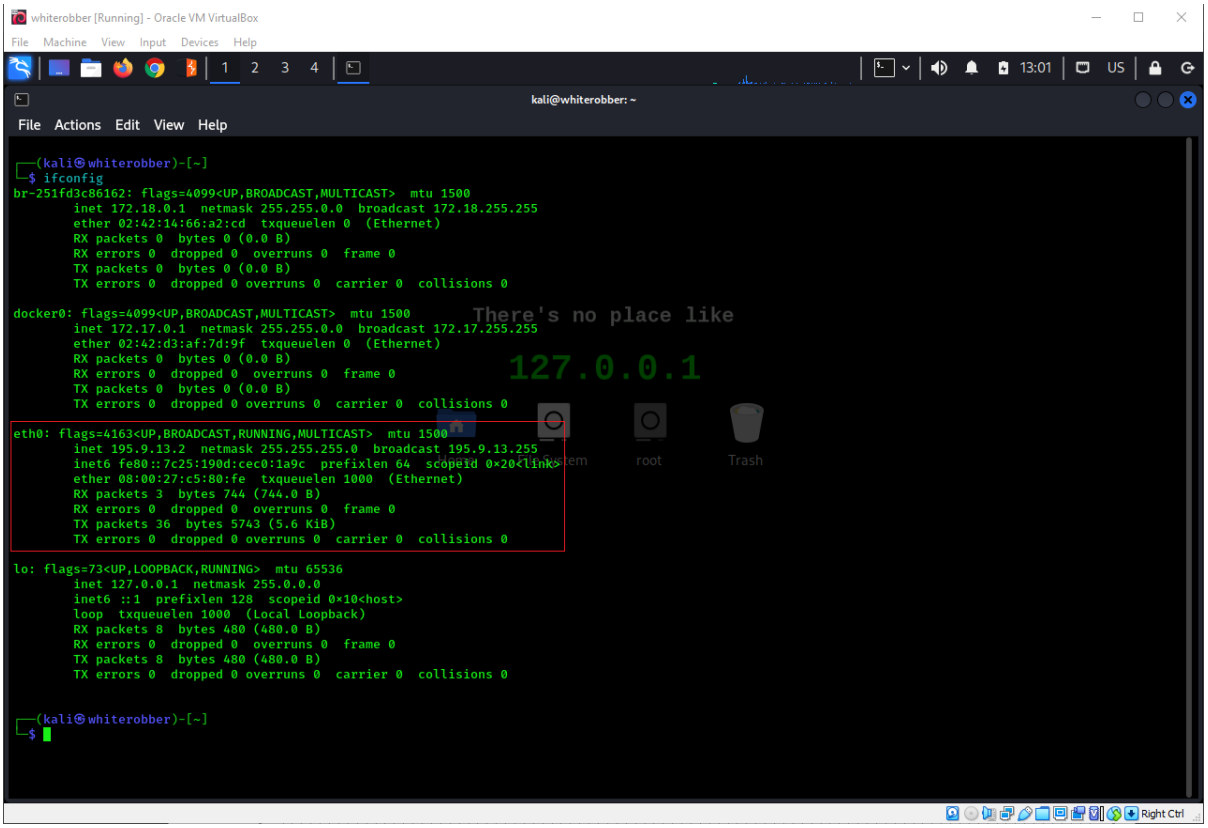


Рисунок 3.5.4 – Перевіряємо IP-адресу на WS-9-13-1

Встановлені налаштування в мережі успішно функціонують, перейдемо до тестування платформи.

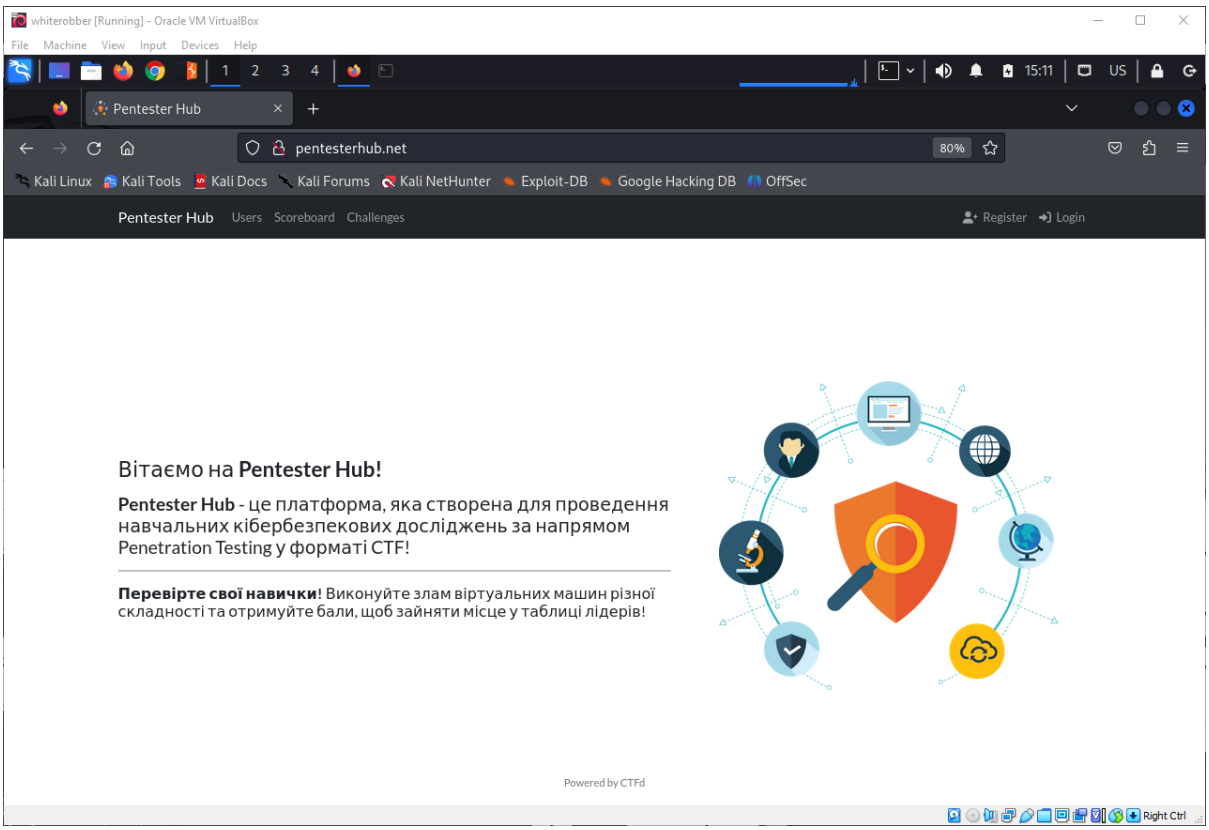


Рисунок 3.5.5 – Успішне відкриття платформи

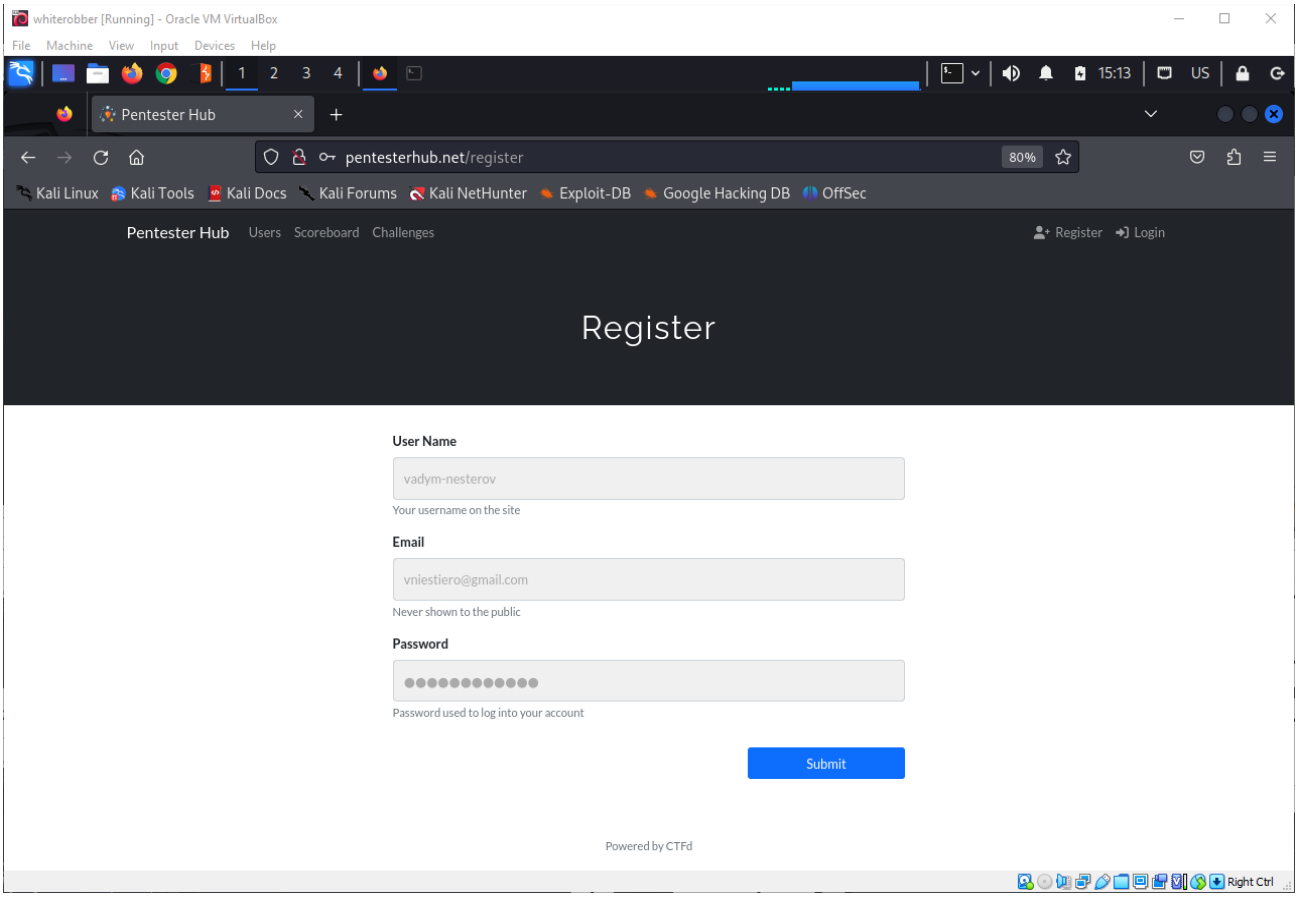


Рисунок 3.5.6 – Створення облікового запису на платформі

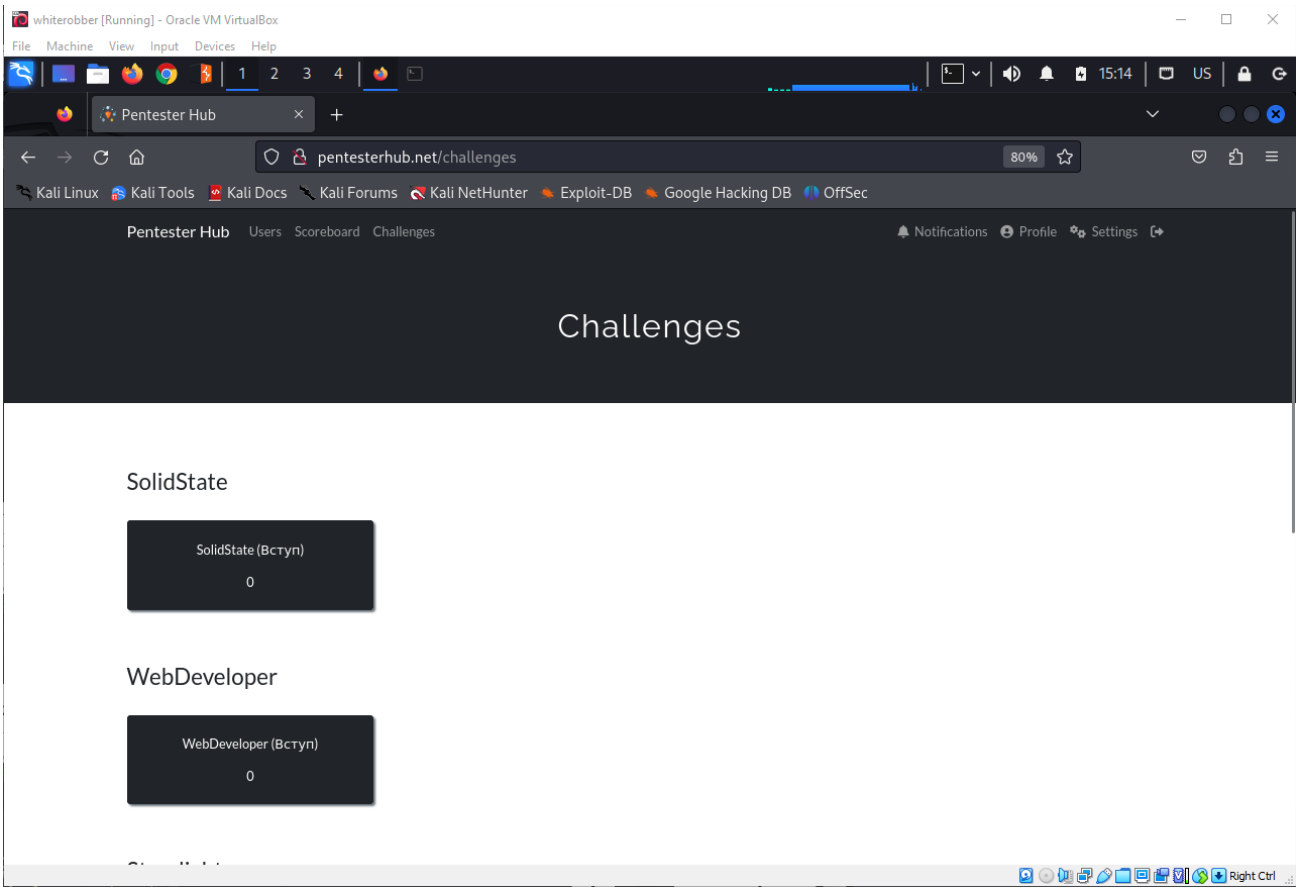


Рисунок 3.5.7 – Обліковий запис успішно створено, відкрито вкладку Challenges

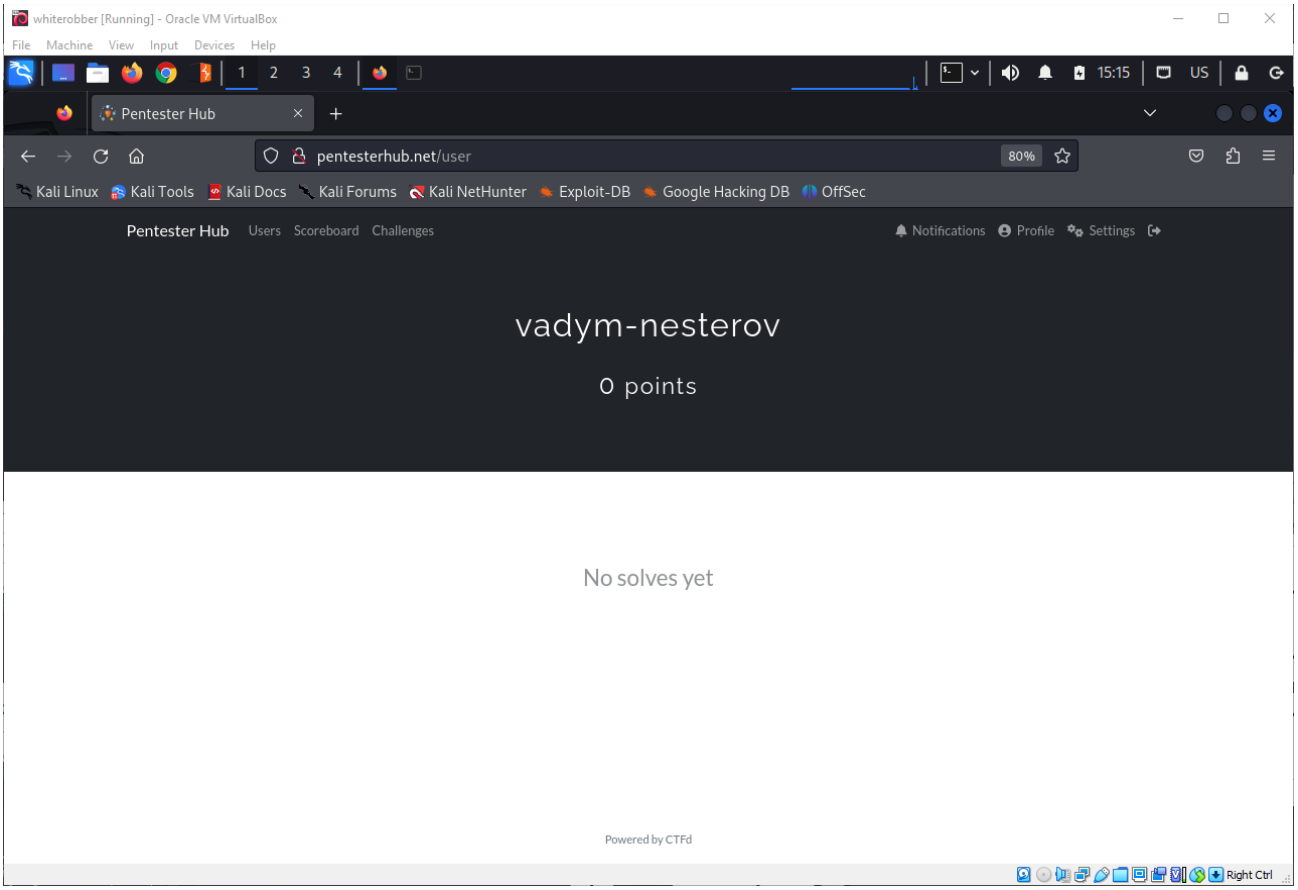


Рисунок 3.5.8 – Профіль користувача

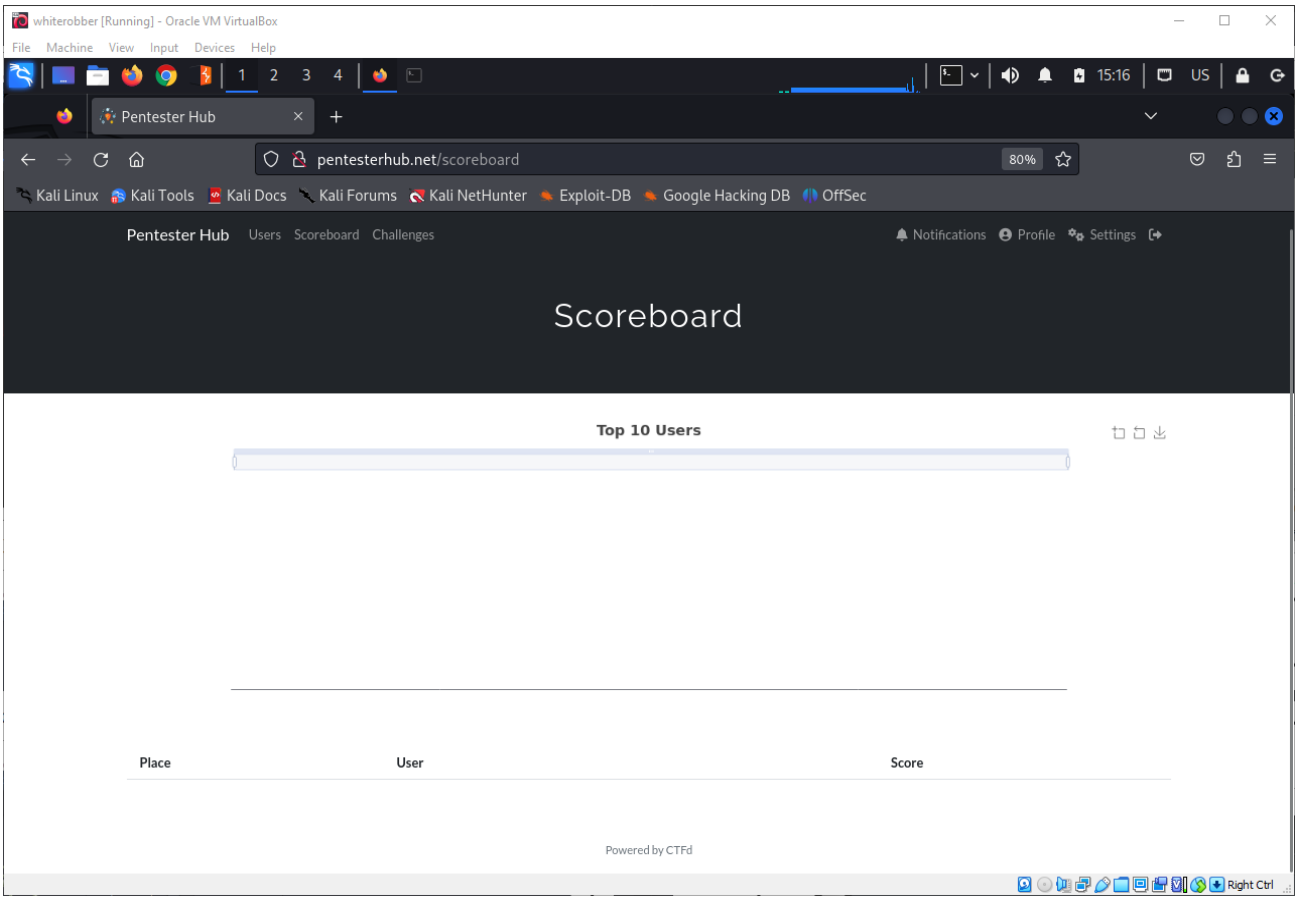


Рисунок 3.5.9 – Таблиця розподілення балів

Розпочнемо вирішення серії завдань для машини SolidState.

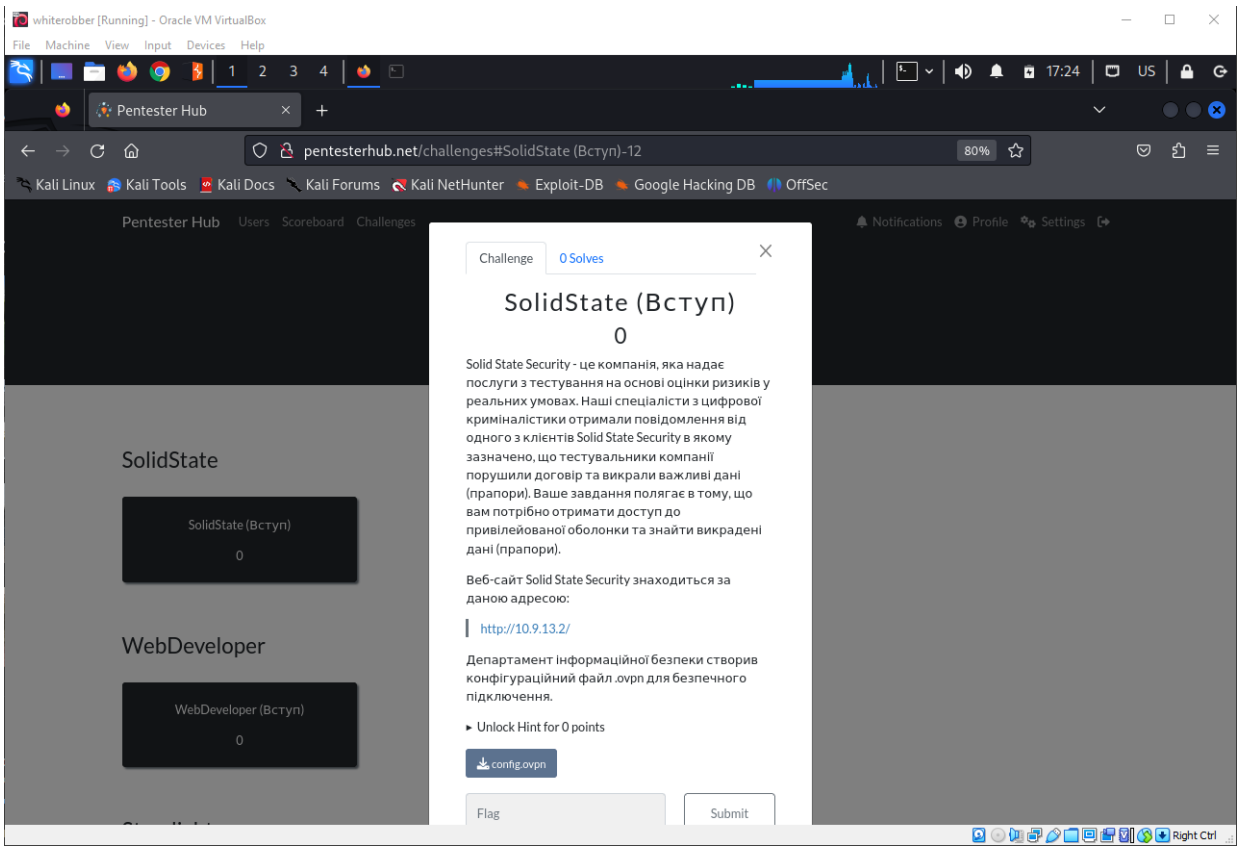


Рисунок 3.5.10 – Перше завдання SolidState (Вступ)

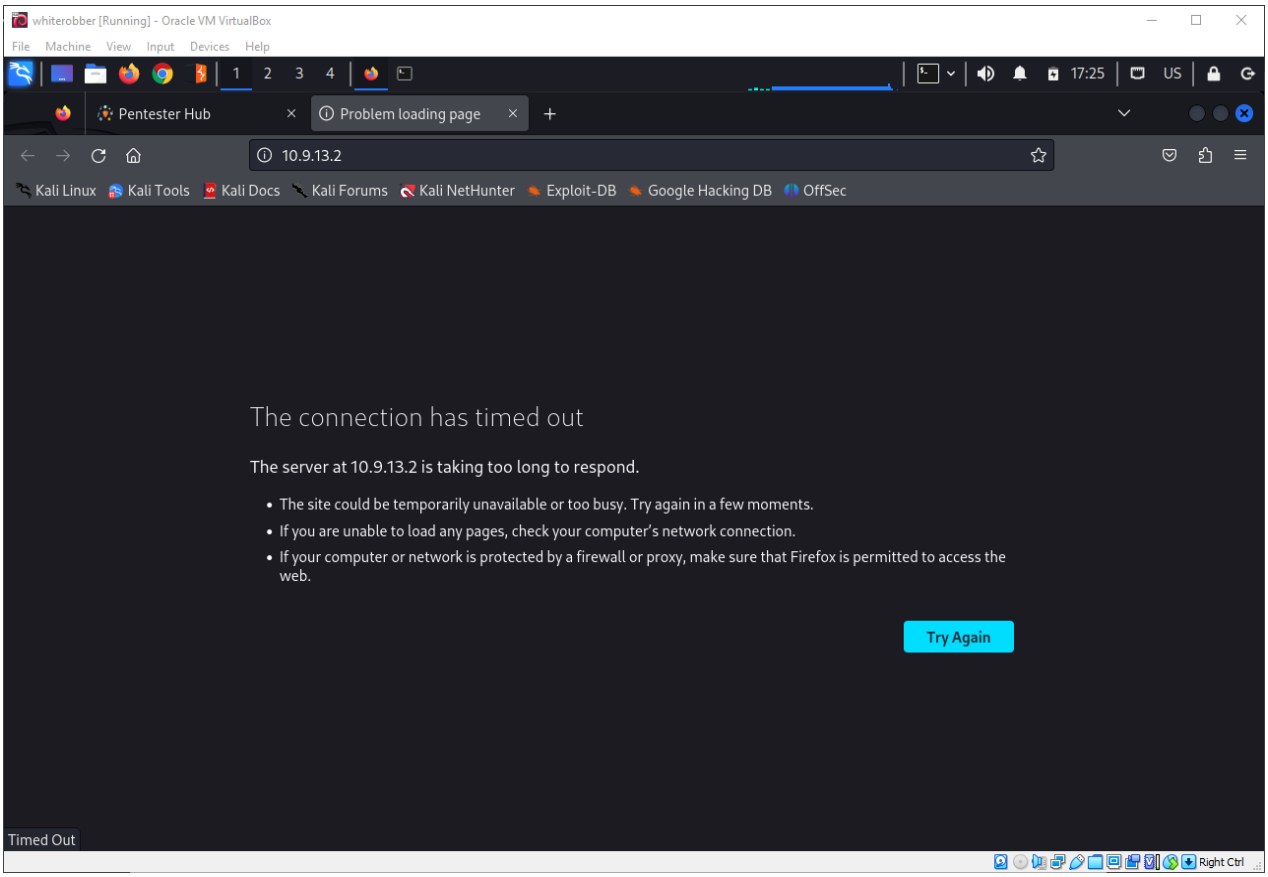


Рисунок 3.5.11 – Відкриття веб-сайту машини SolidState, зв'язок відсутній

Через те, що VM-жертва SolidState знаходиться в підмережі LAN VMs, яка немає доступу в Інтернет, єдиний спосіб отримати доступ до SolidState це виконати підключення до OpenVPN сервера Server-9-13-1 використовуючи конфігураційний файл .ovpn, який наведено в завдані SolidState (Вступ).

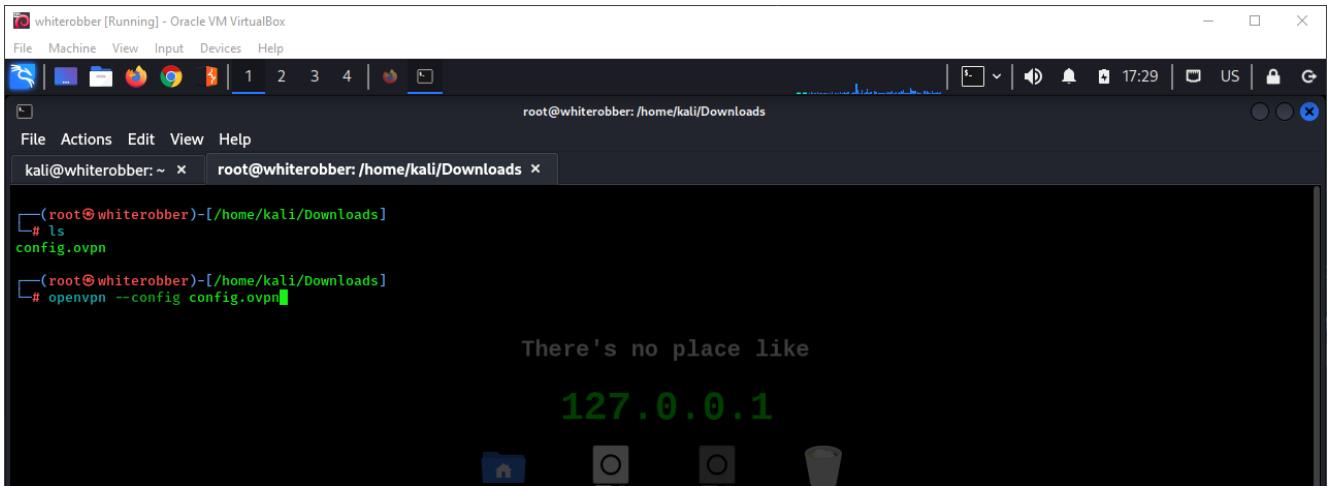


Рисунок 3.5.12 – Підключення до OpenVPN сервера Server-9-13-1 використовуючи конфігураційний файл .ovpn

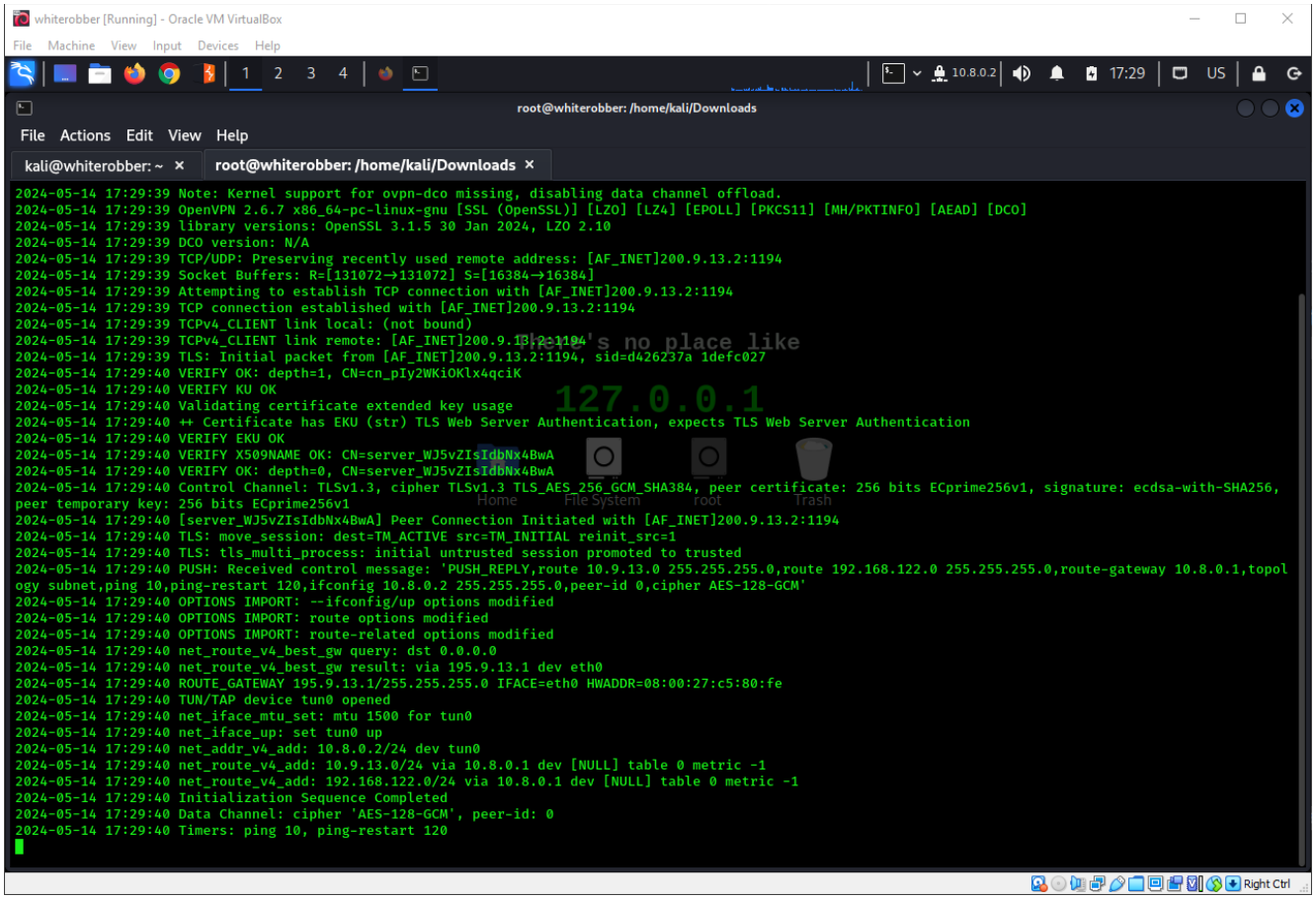


Рисунок 3.5.13 – Підключення до OpenVPN сервера Server-9-13-1 використовуючи конфігураційний файл .ovpn успішно виконано

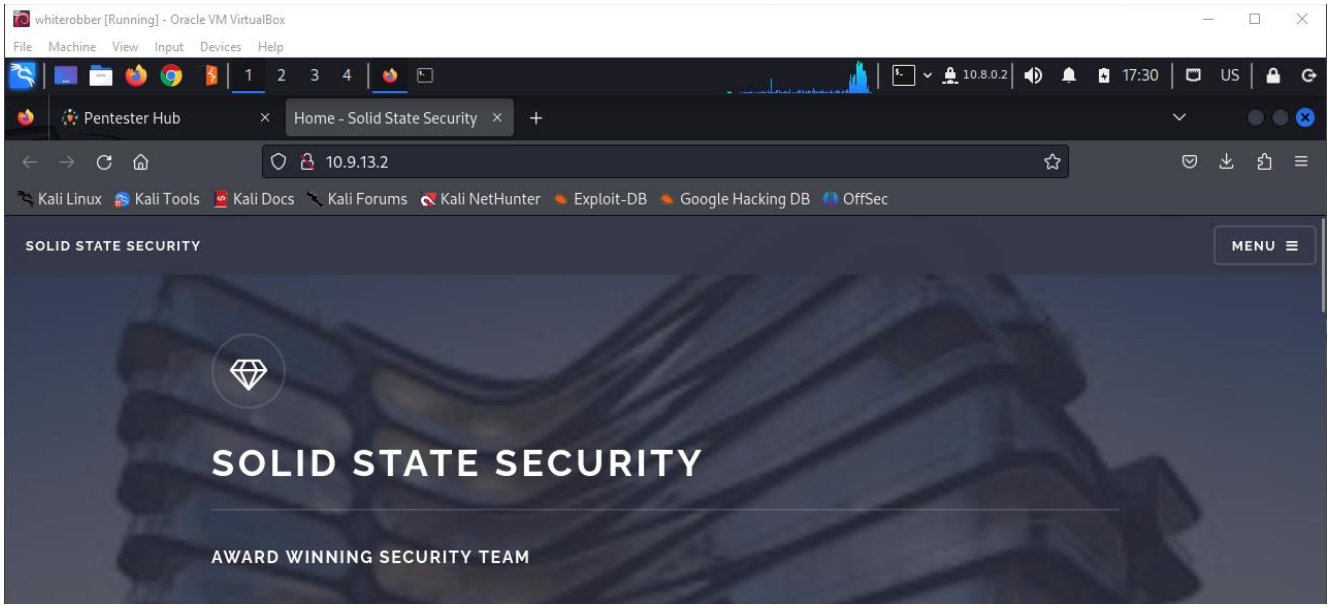


Рисунок 3.5.14 – Успішне відкриття веб-сайту машини SolidState

Було проведено ознайомлення з описом завдання, виконано підключення до OpenVPN сервера та успішно відкрито веб-сайту машини SolidState, наступним кроком є перегляд безкоштовної підказки для того, щоб надіслати прапор для виконання завдання та відкрити наступне завдання в серії.

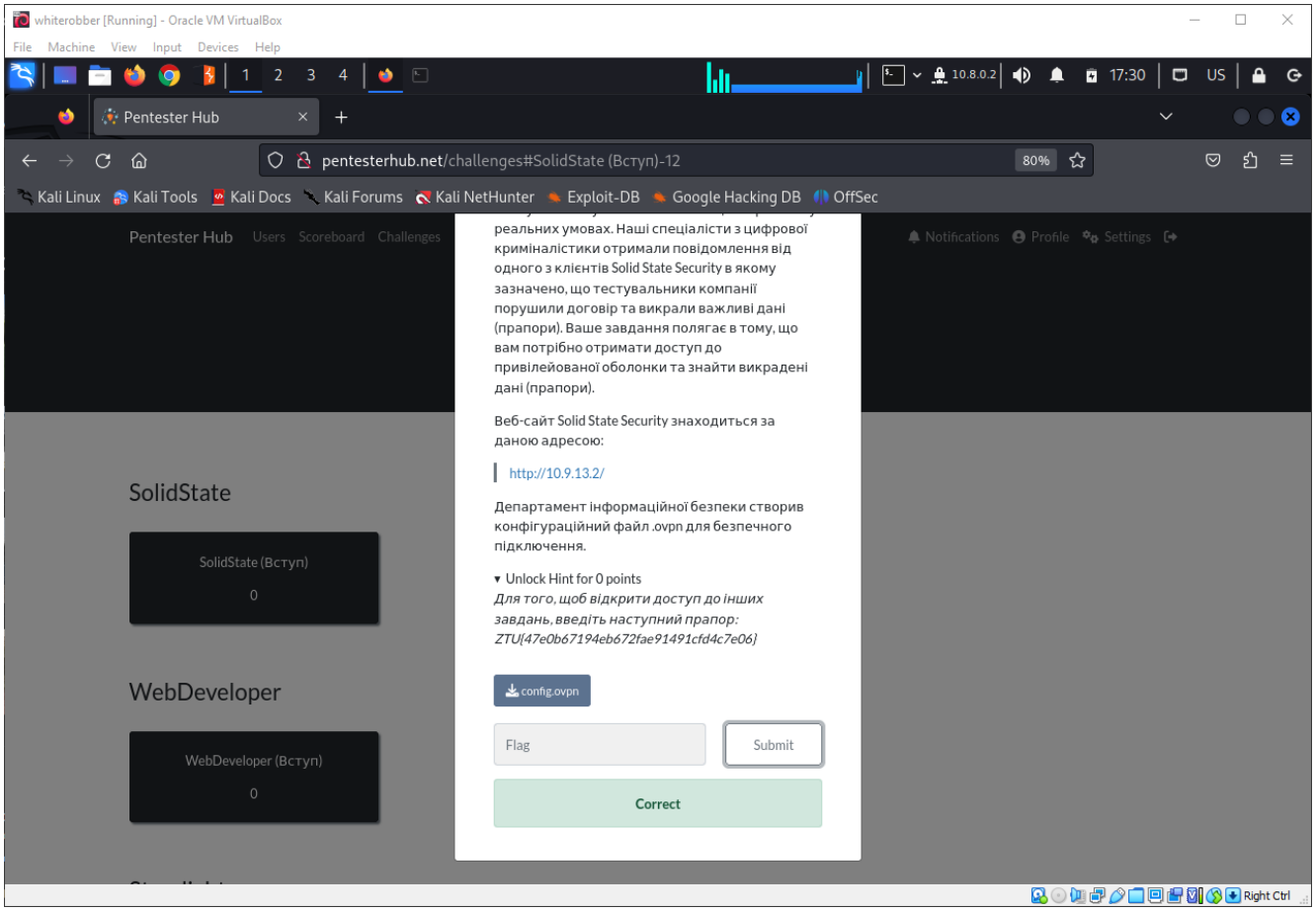


Рисунок 3.5.15 – Прапор успішно надіслано, завдання зараховано

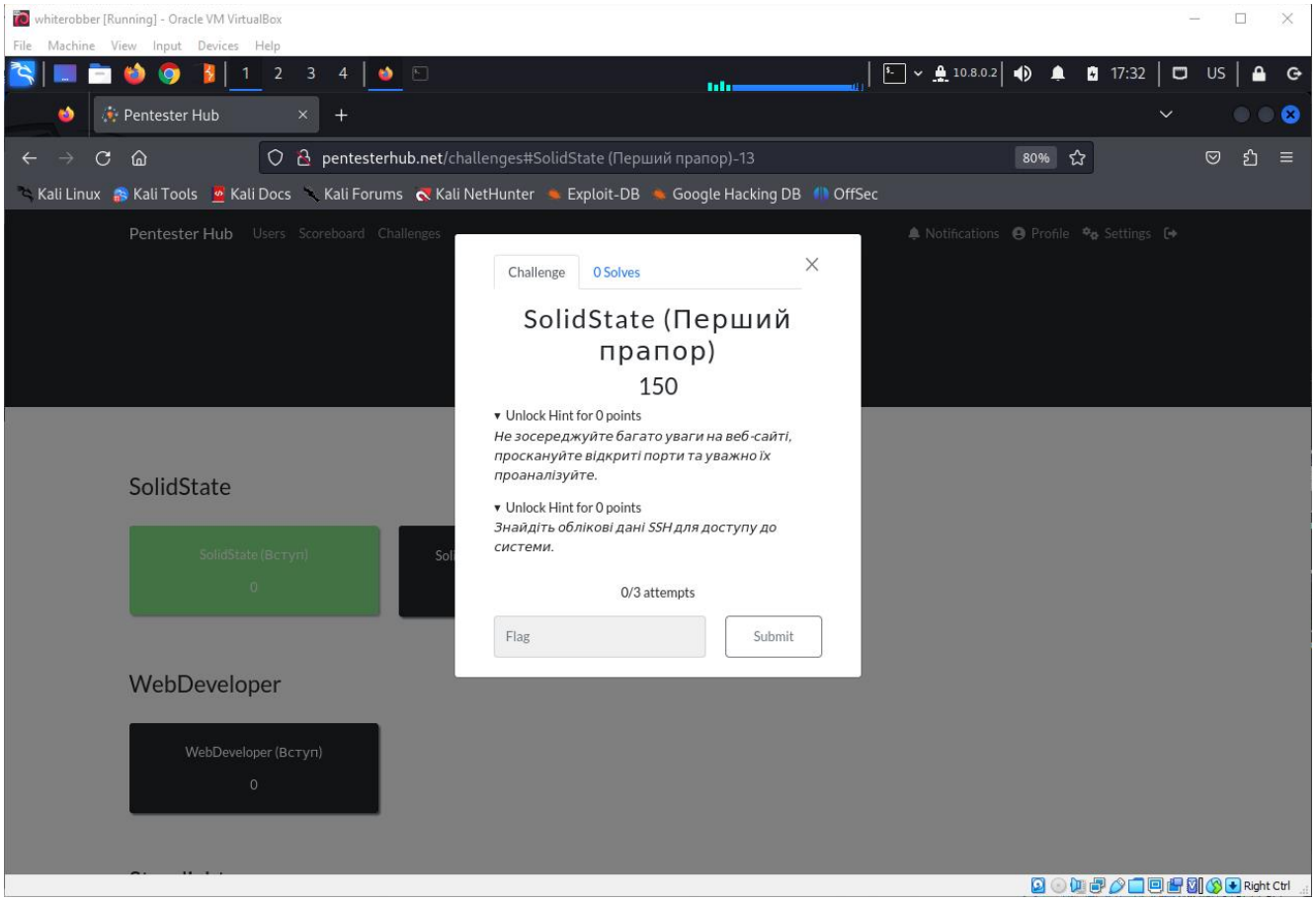


Рисунок 3.5.16 – Друге завдання SolidState (Перший прапор)

Звертаючи увагу на першу безкоштовну підказку можна зробити висновок, що потрібно виконати сканування відкритих портів та спиратися на інформацію, яка буде отримана після сканування.

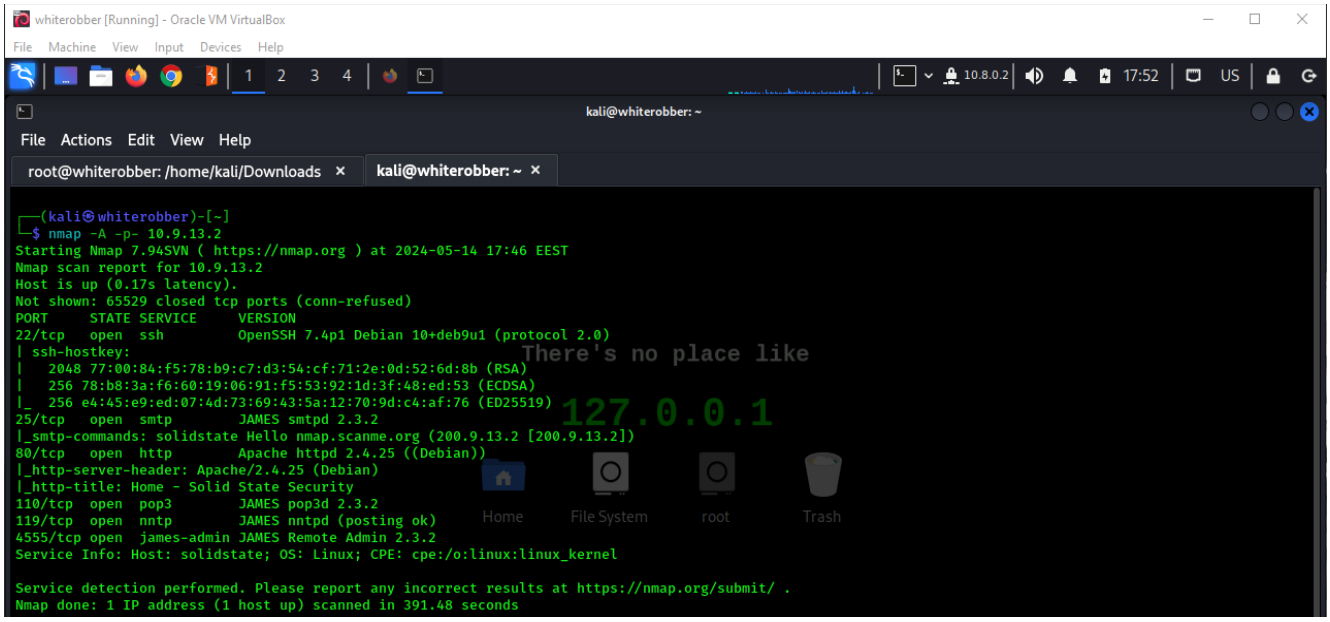


Рисунок 3.5.17 – Результат сканування відкритих портів за допомогою інструмента nmap

Звертаючи увагу на результат сканування, помічено цікавий порт 4555/tcp (james-admin), спробуємо виконати підключення до даного порту використовуючи утиліту netcat.

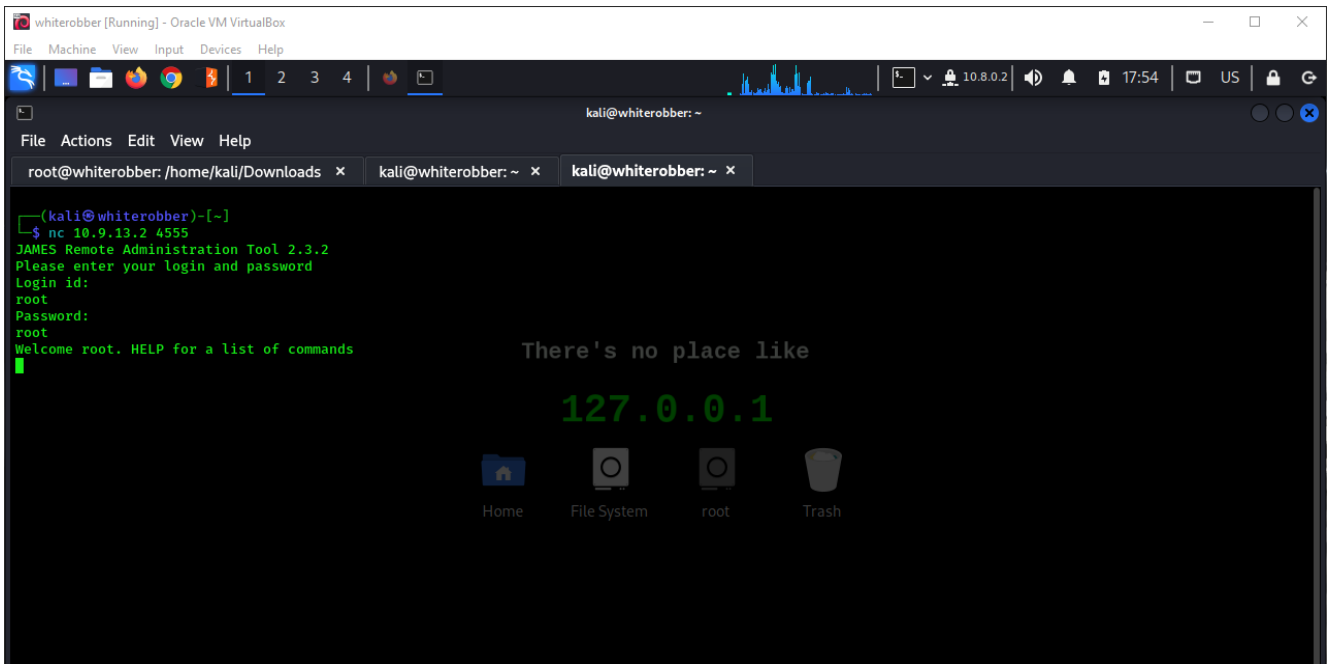


Рисунок 3.5.18 – Результат підключення до порту 4555 використовуючи утиліту netcat, для виконання входу були використані облікові дані за замовчуванням root:root

Наступним кроком є використання команди help для того, щоб отримати список доступних команд.

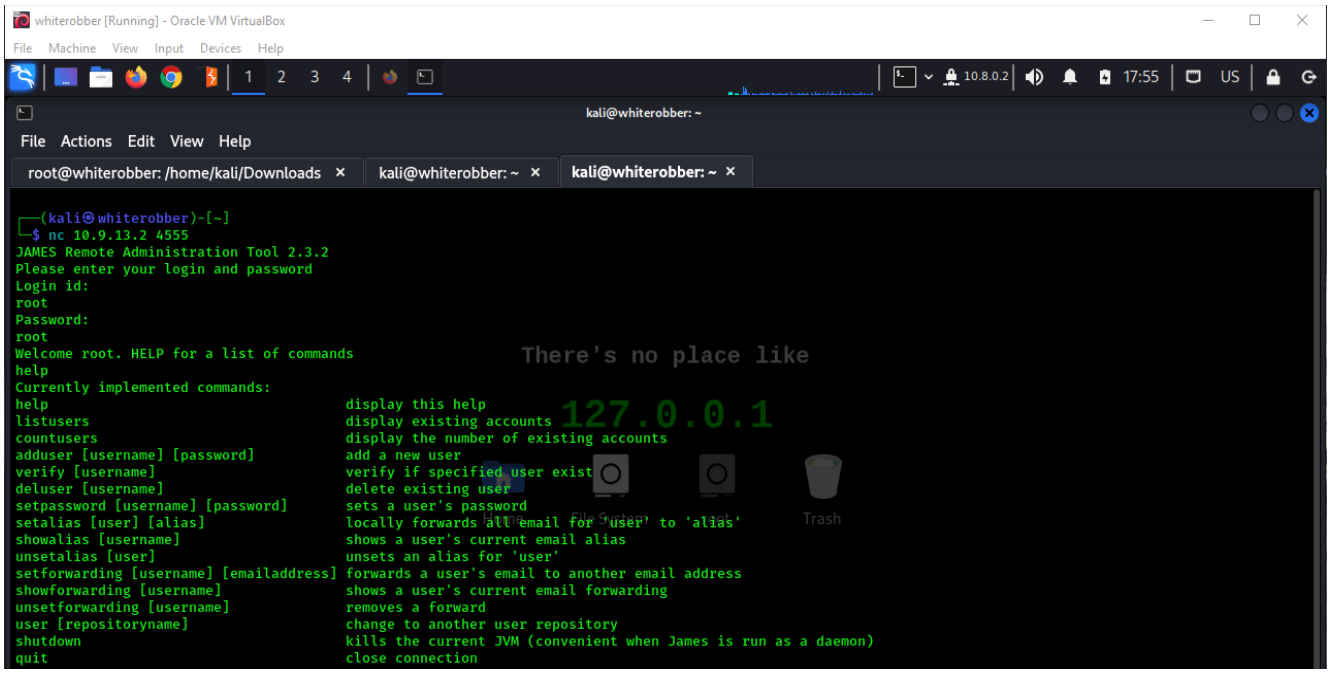


Рисунок 3.5.19 – Результат використання команди help

Уважно переглянувши всі доступні команди, спробуємо виконати команду listusers та вивести список наявних користувачів.

```
listusers
Existing accounts 5
user: james
user: thomas
user: john
user: mindy
user: mailadmin
```

Рисунок 3.5.20 – Результат використання команди listusers

Команда listusers показала п'ять наявних користувачів, оскільки ми маємо права root спробуємо змінити пароль для кожного з користувачів використовуючи команду setpassword.

```
setpassword james james
Password for james reset
setpassword thomas thomas
Password for thomas reset
setpassword john john
Password for john reset
setpassword mindy mindy
Password for mindy reset
setpassword mailadmin mailadmin
Password for mailadmin reset
```

Рисунок 3.5.21 – Результат використання команди setpassword

Повертаючись до результату сканування відкритих портів звертаємо увагу, на порт 110/tcp (pop3), спробуємо виконати підключення до даного порту, для перевірки пошти кожного з наявних користувачів використовуючи telnet.

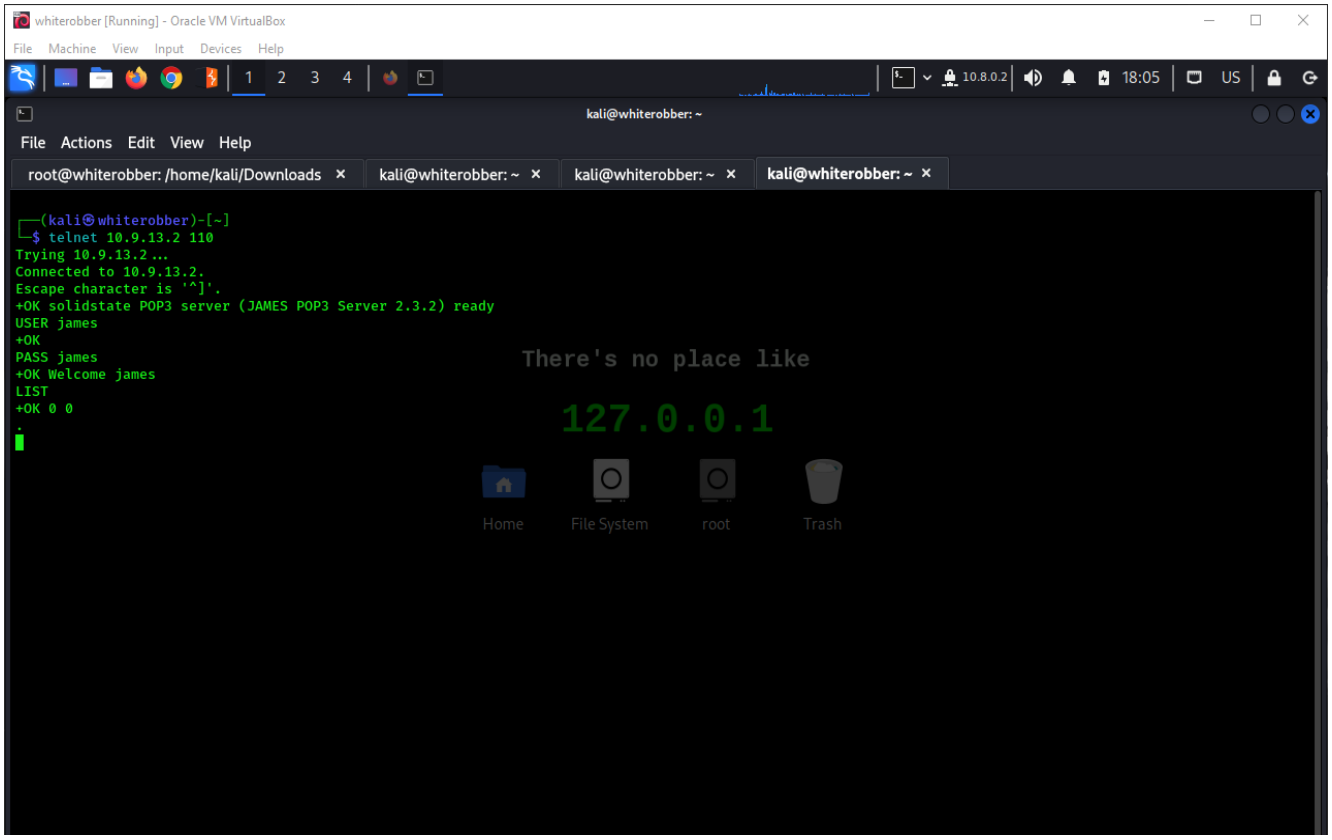


Рисунок 3.5.22 – У користувача james відсутні повідомлення

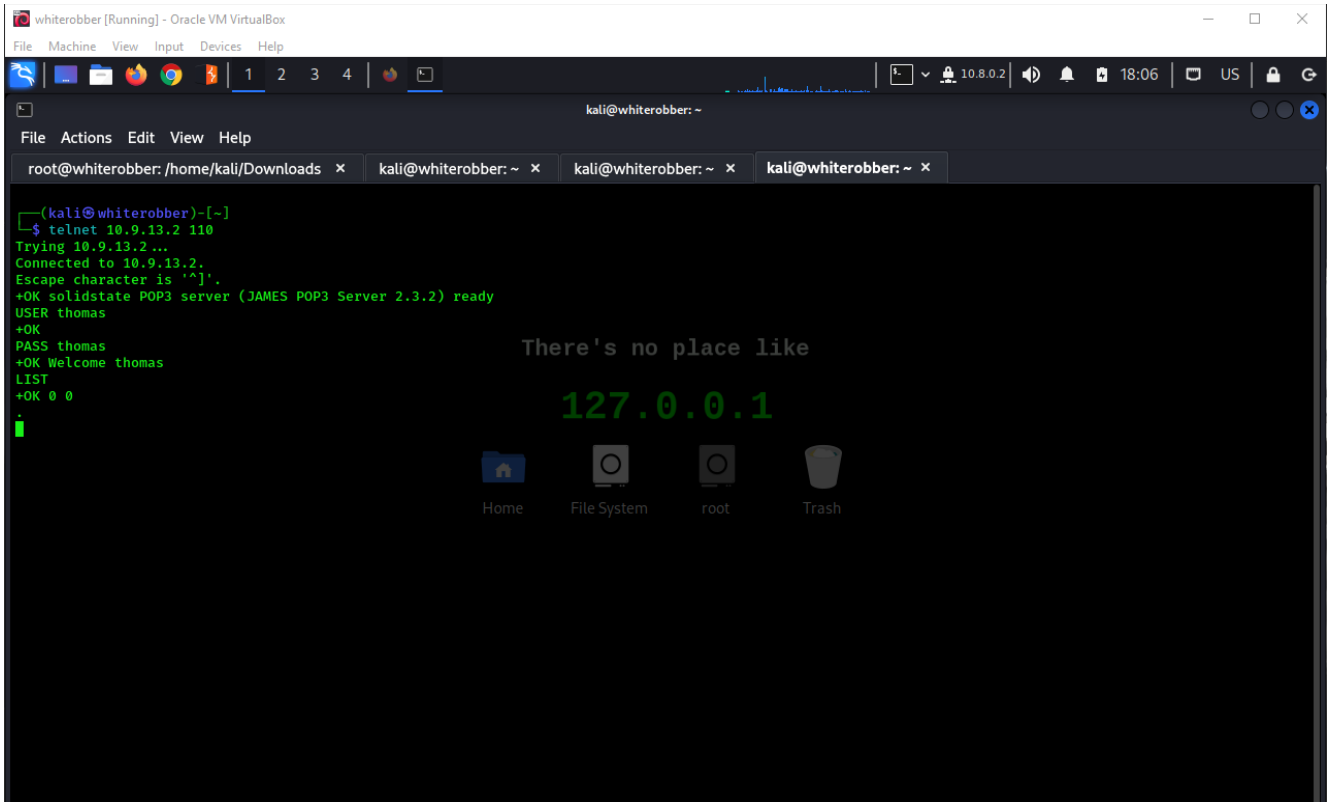


Рисунок 3.5.23 – У користувача thomas відсутні повідомлення

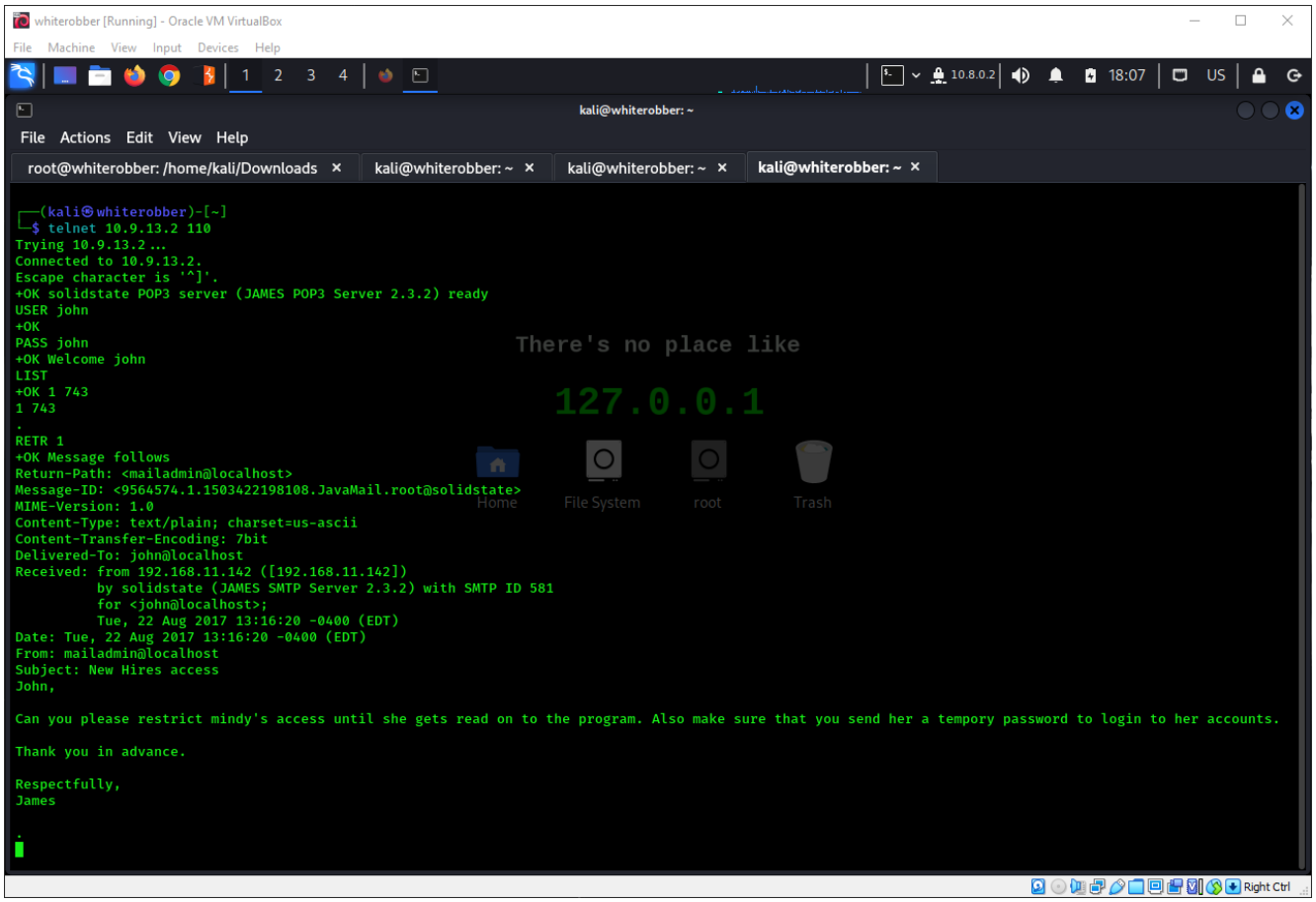


Рисунок 3.5.24 – У користувача john помічено одне повідомлення, перегляд повідомлення

Переглянувши повідомлення на пошті john, можна зробити висновок, що на пошті користувача mindy може бути щось цікаве.

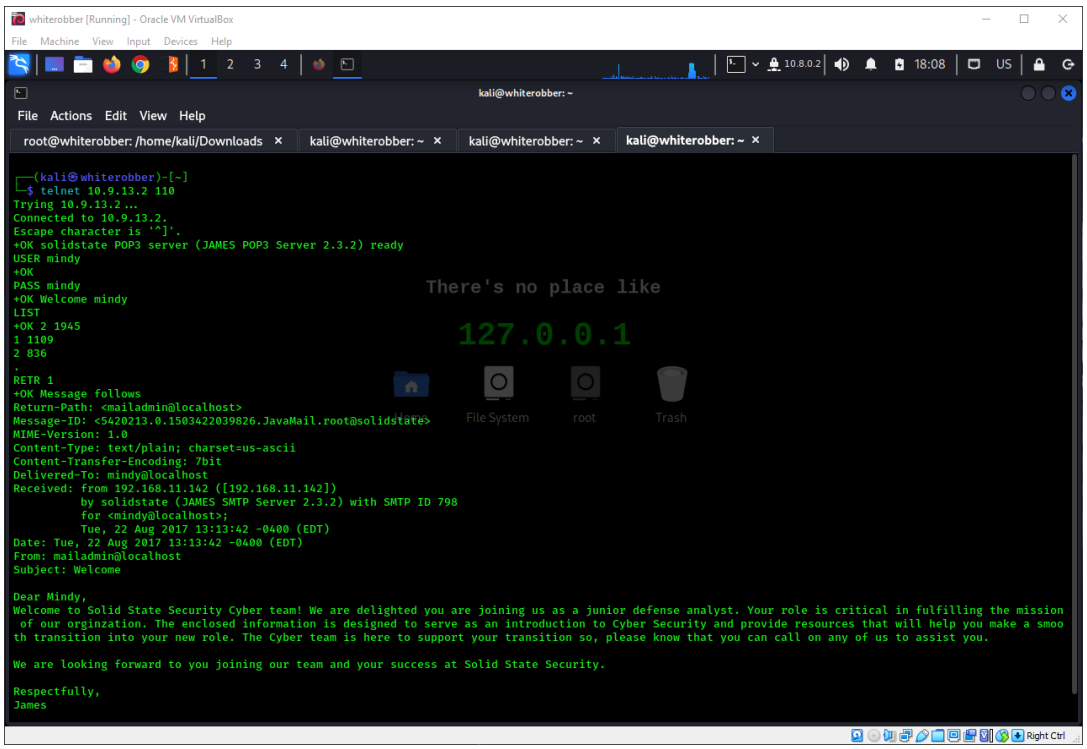


Рисунок 3.5.25 – У користувача mindy помічено два повідомлення, перегляд першого повідомлення

Переглянувши перше повідомлення на пошті mindy стало зрозуміло, що вона новий співробітник в Solid State Security Cyber, переглянемо друге повідомлення.

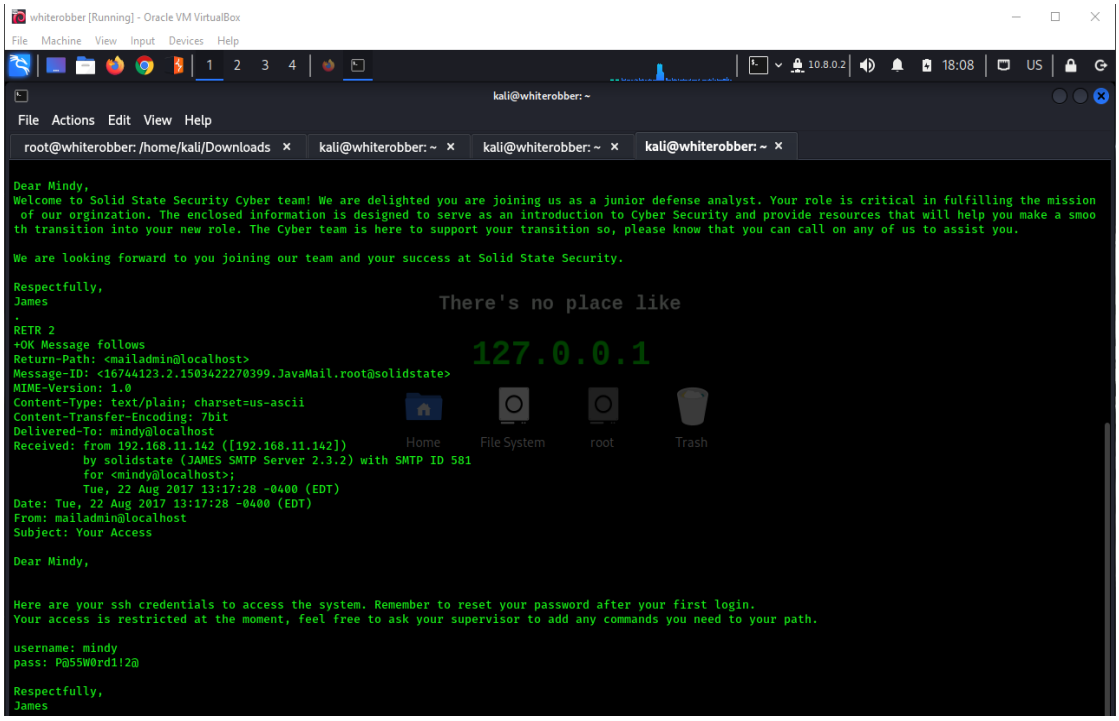


Рисунок 3.5.26 – Перегляд другого повідомлення

Переглянувши друге повідомлення на пошті mindy було помічено облікові дані SSH. В другій безкоштовній підказці також було вказано, що потрібно знайти облікові дані SSH.

Користувач mailadmin нажаль не має електронної пошти.

Маючи облікові дані SSH mindy виконаємо підключення та спробуємо знайти прапор для вирішення завдання.

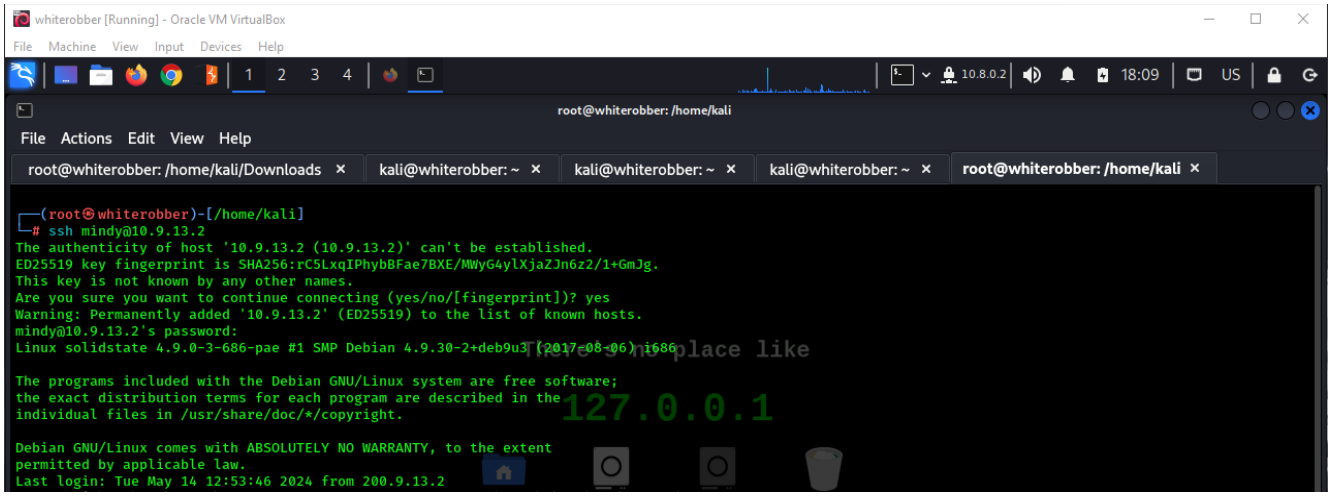


Рисунок 3.5.27 – Виконуємо підключення по SSH

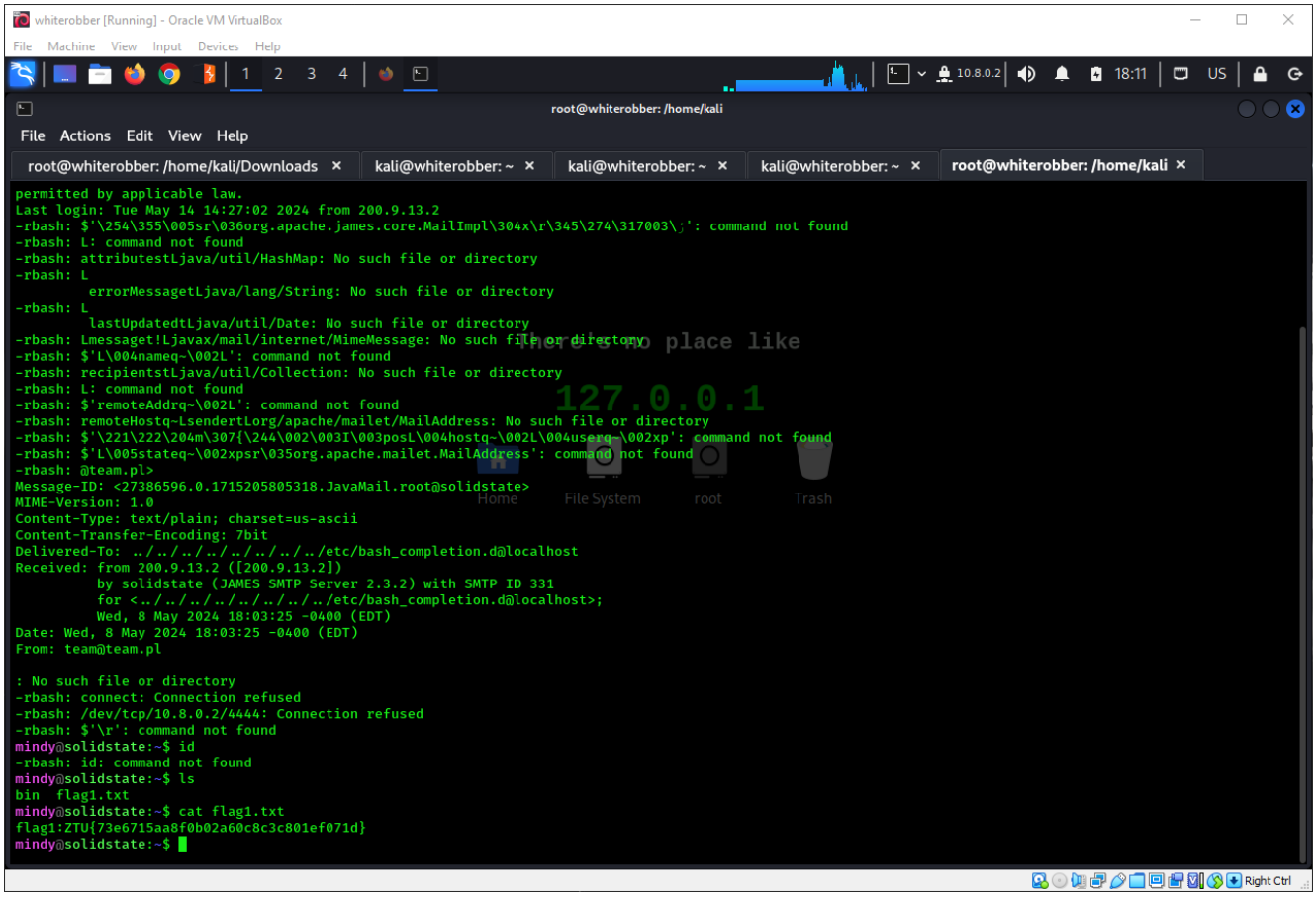


Рисунок 3.5.28 – Підключення успішне, перший прапор успішно знайдено

Для перевірки функціоналу спробуємо надіслати некоректний прапор.

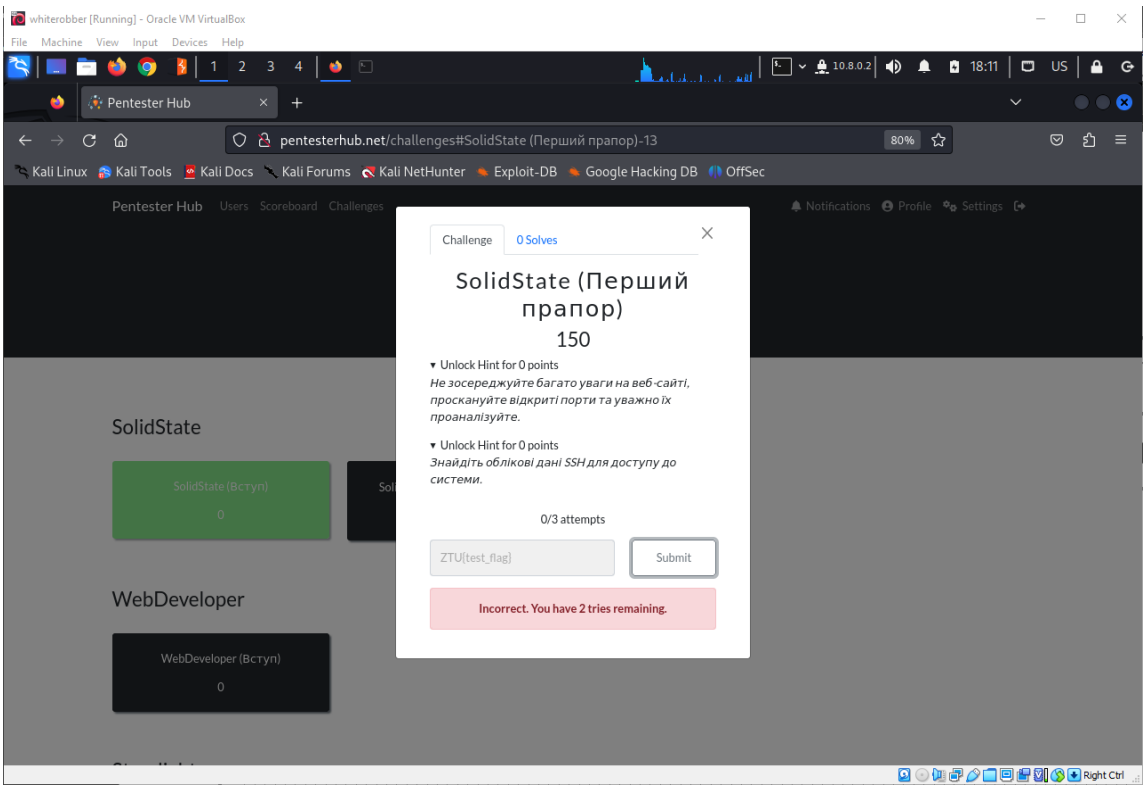


Рисунок 3.5.29 – Перевірка працює, виявлено некоректний прапор, кількість спроб дорівнює двом

Тепер надішлемо знайдений прапор.

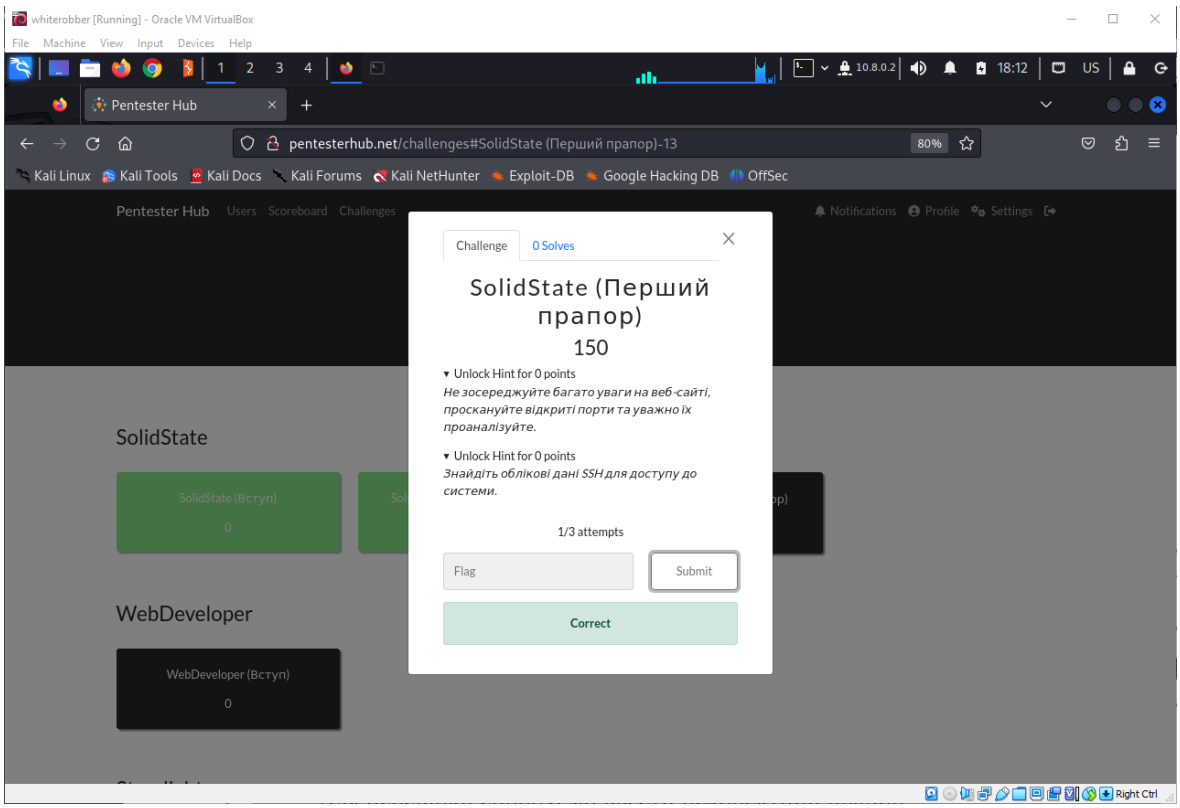


Рисунок 3.5.30 – Прапор успішно надіслано, завдання зараховано

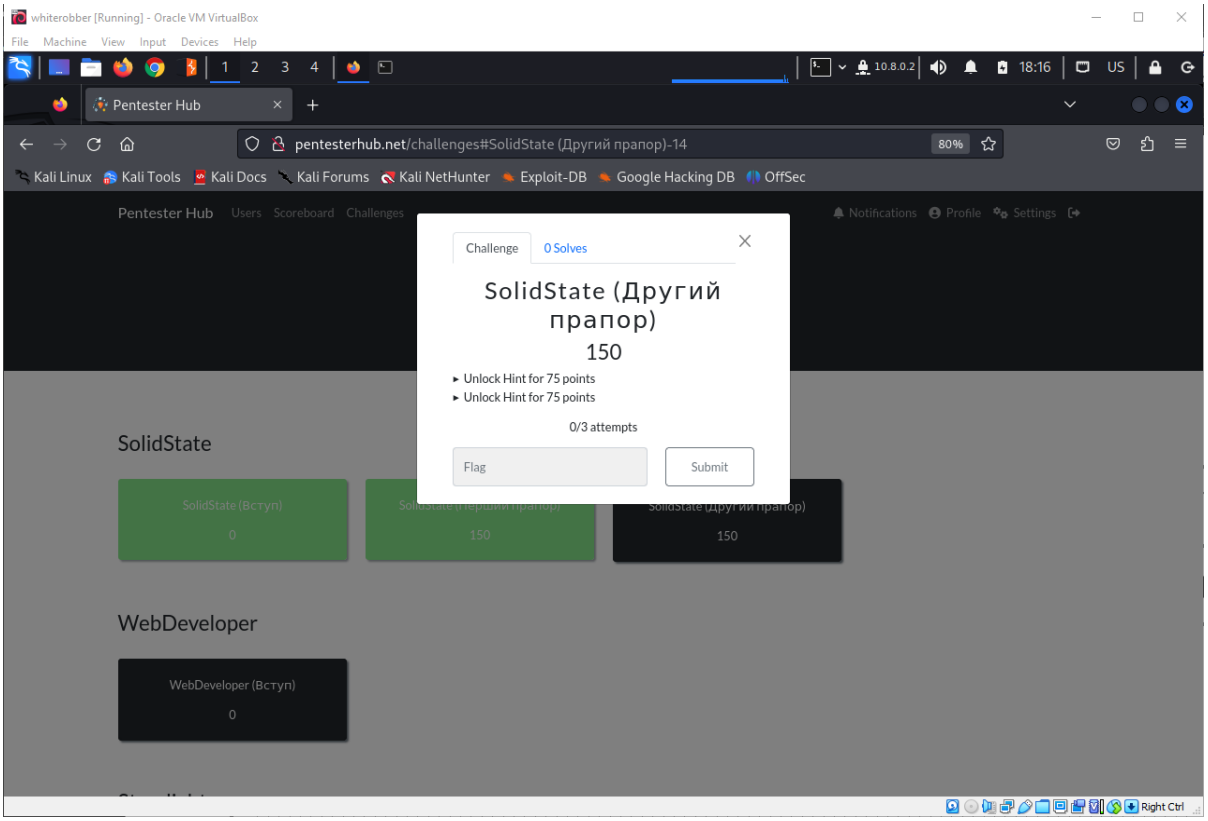


Рисунок 3.5.31 – Третє/фінальне завдання SolidState (Другий прапор)

В завдані зазначено дві заблоковані підказки, для перевірки функціоналу розблокуємо кожну з них.

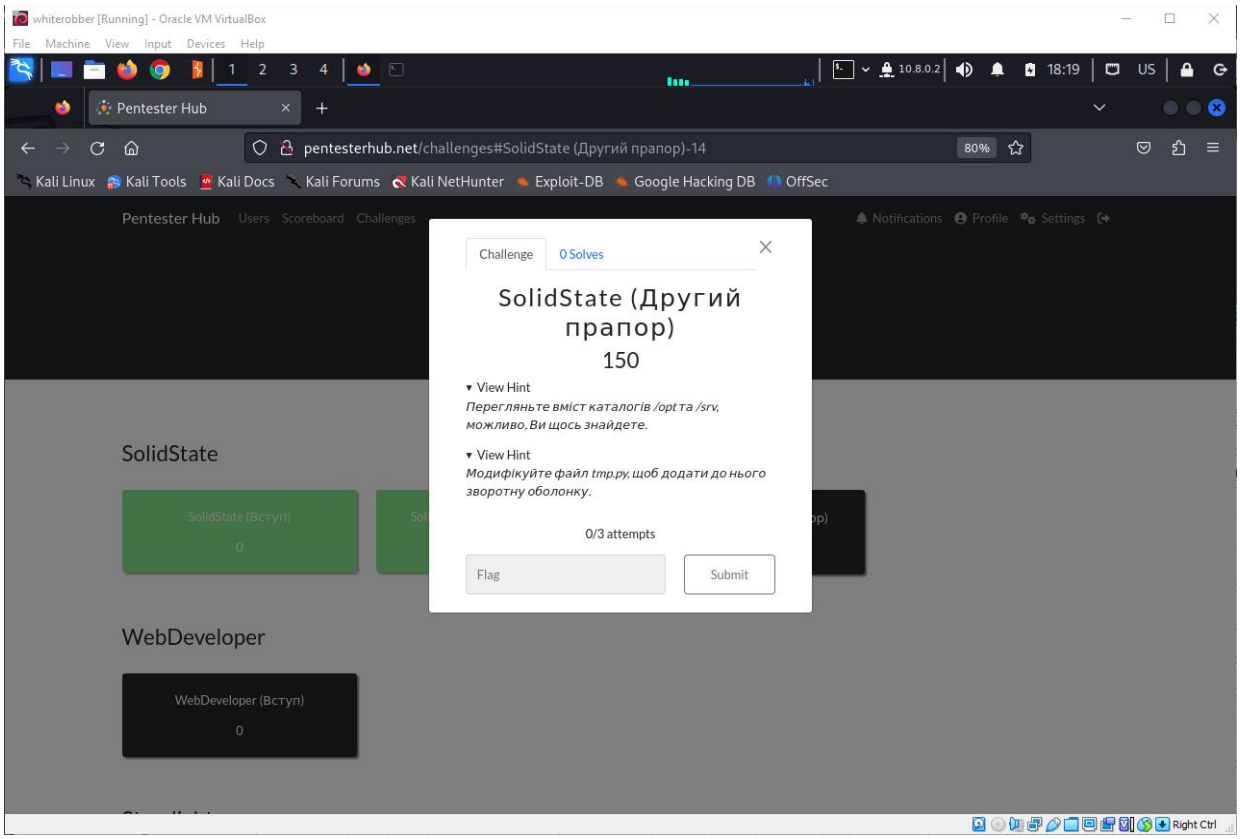


Рисунок 3.5.32 – Розблоковано дві підказки

В попередньому завдані виконавши підключення по SSH та комаду id було отримано повідомлення «-rbash: id: command not found» це означає, що оболонка обмежена.

Один із способів виконати підключення по SSH використовуючи оболонку bash це створити її за допомогою «python -c "import pty;pty.spawn('/bin/bash')"».

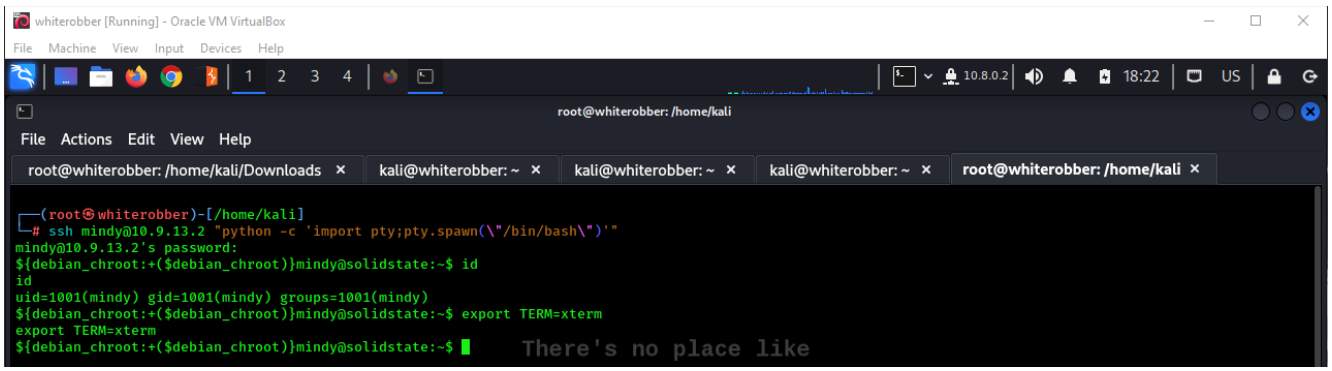


Рисунок 3.5.33 – Виконуємо підключення по SSH,
оболонку bash успішно створено

Після отримання оболонки bash, згідно вказаній інформації в першій розблокованій підказці, переглянемо вміст каталогів opt та srv.

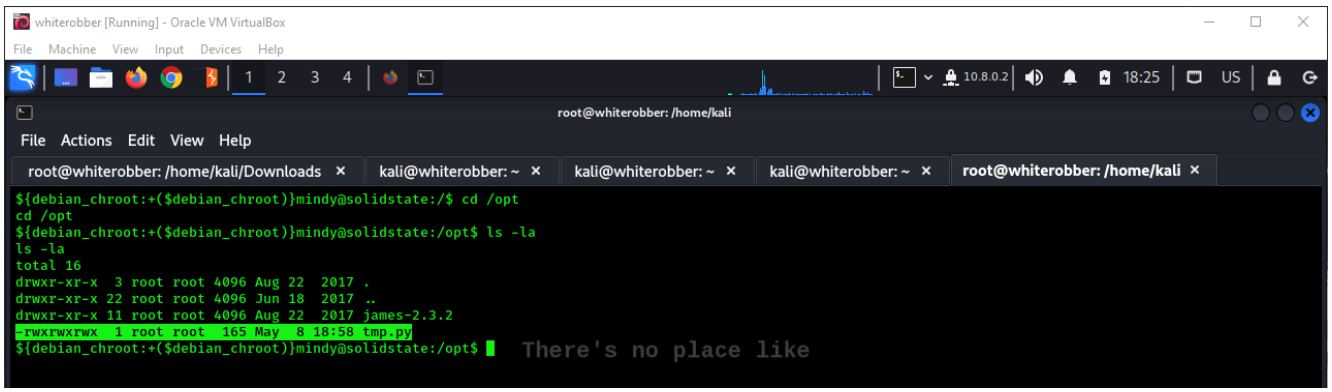


Рисунок 3.5.34 – Переглядаємо вміст каталогу opt, знаходимо цікавий файл tmp.py, яким володіє користувач root та який можна модифікувати

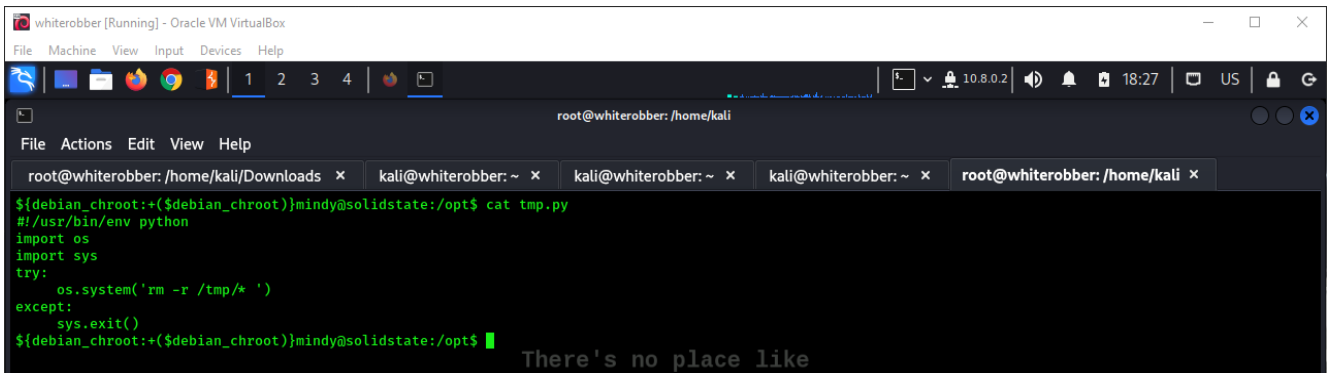


Рисунок 3.5.35 – Переглядаємо вміст файлу tmp.py

Згідно вказаній інформації в другій розблокованій підказці, модифікуємо файл tmp.py та спробуємо додати в нього зворотну оболонку.

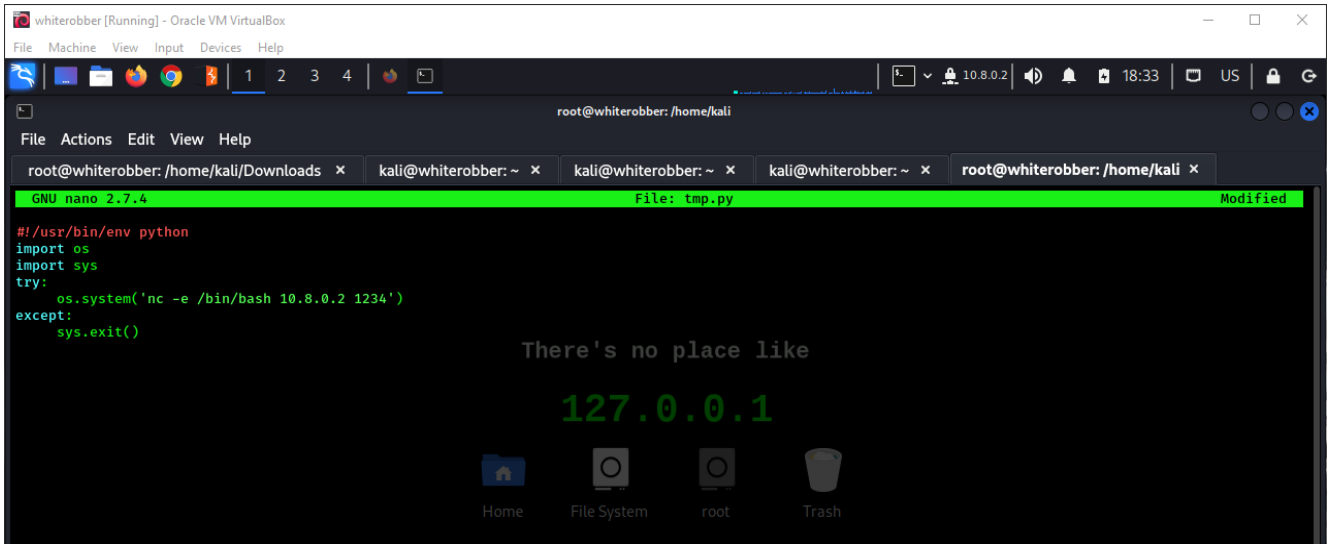


Рисунок 3.5.36 – Редагування файлу tmp.py, додання зворотної оболонки

Зворотну оболонку успішно додано, створимо нове вікно терміналу та почнемо прослуховувати встановлений порт 1234.

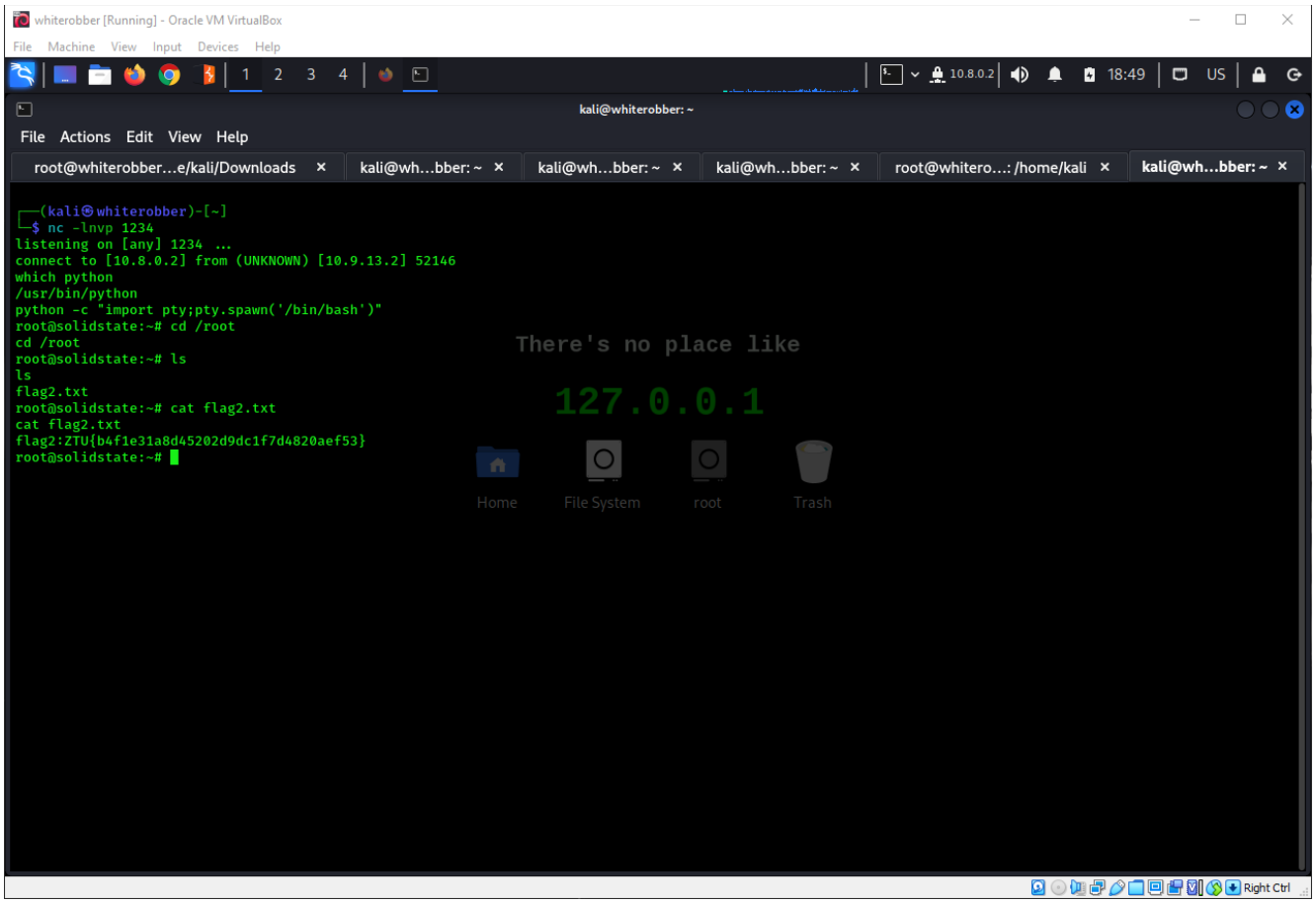


Рисунок 3.5.37 – Зворотна оболонка спрацювала, підключення успішне, другий прапор успішно знайдено

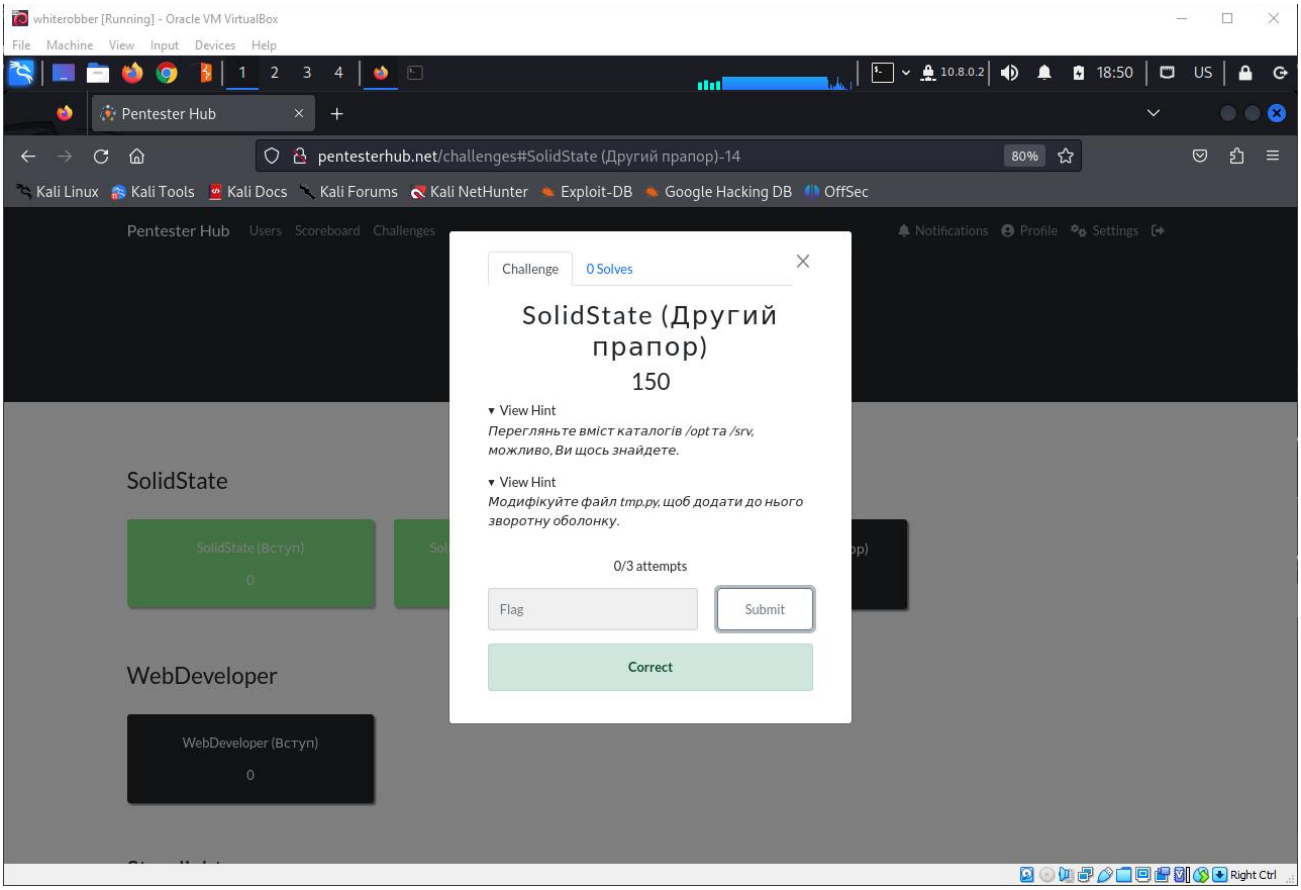


Рисунок 3.5.38 – Прапор успішно надіслано, завдання зараховано

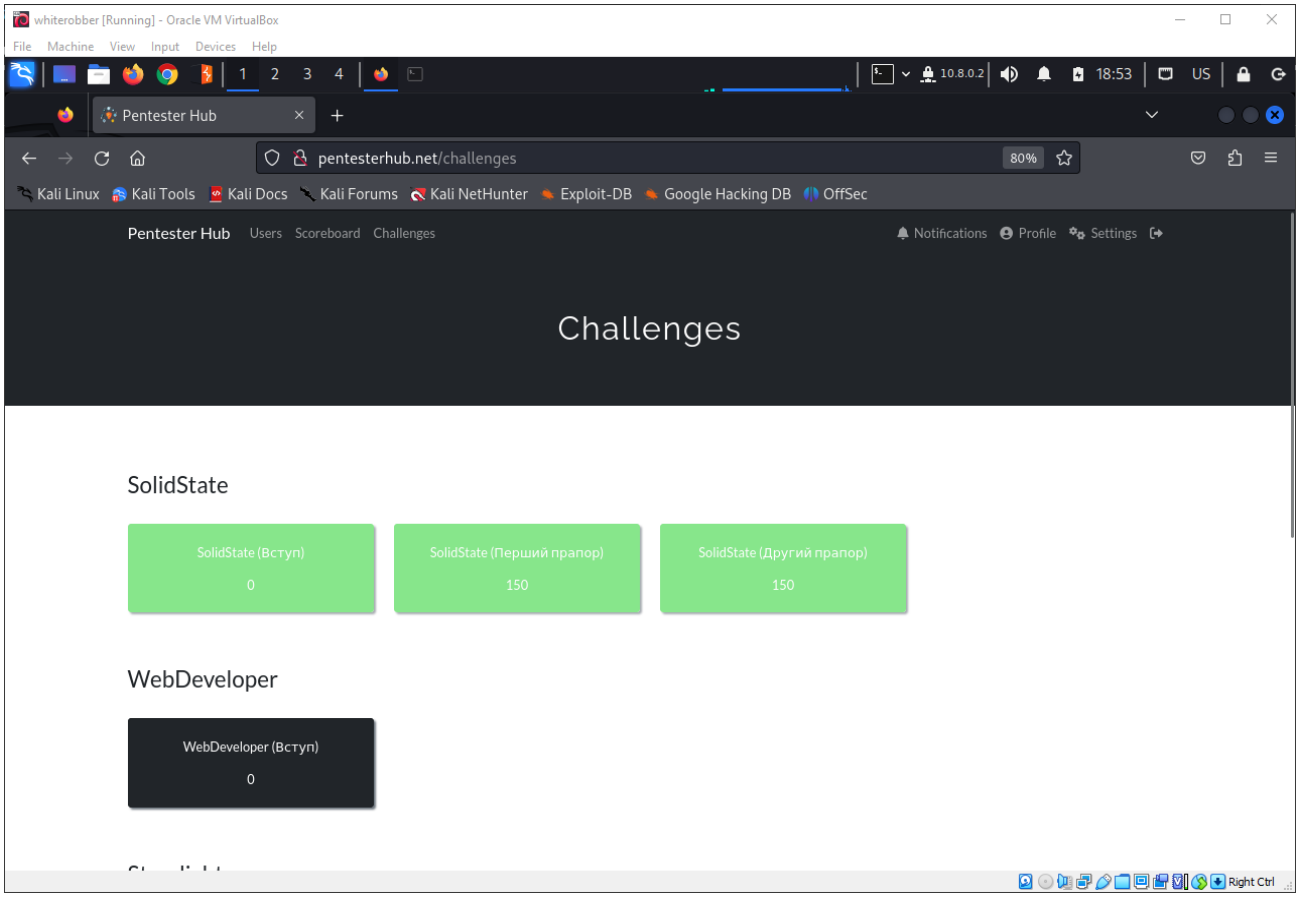


Рисунок 3.5.39 – Серія завдань SolidState успішно протестована (вирішена)

Розпочнемо вирішення серії завдань для машини WebDeveloper.

Зазначимо, процес ознайомлення з описом завдання, виконання підключення до OpenVPN сервера, успішне відкриття веб-сайту машини WebDeveloper та перегляд безкоштовної підказки для того, щоб надіслати прапор для виконання завдання та відкрити наступне завдання в серії аналогічний до SolidState.

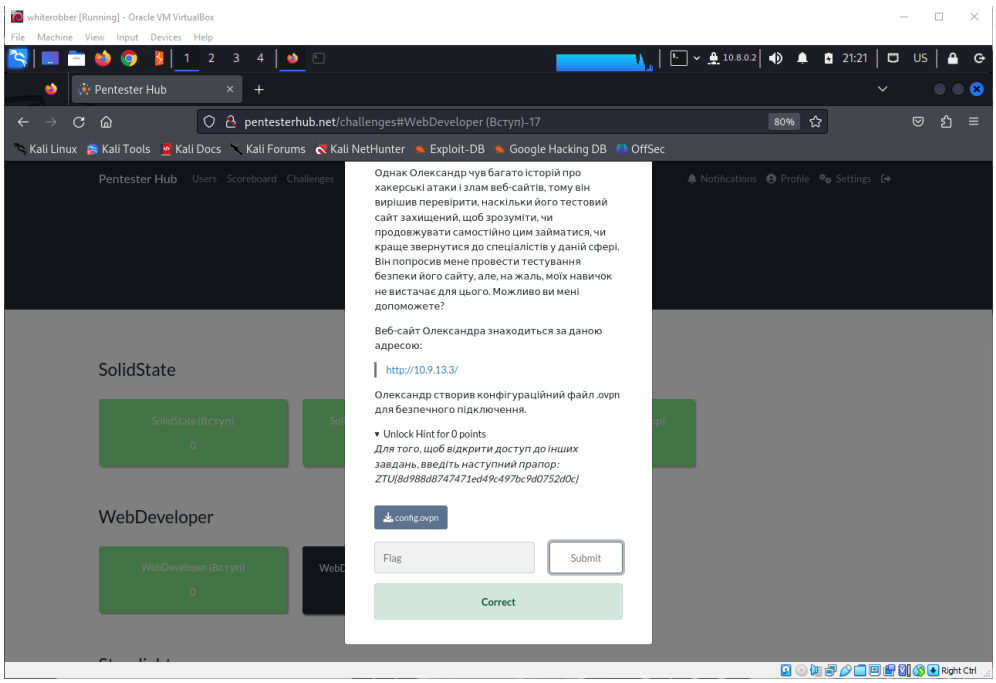


Рисунок 3.5.40 – Перше завдання WebDeveloper (Вступ), прапор успішно надіслано, завдання зараховано

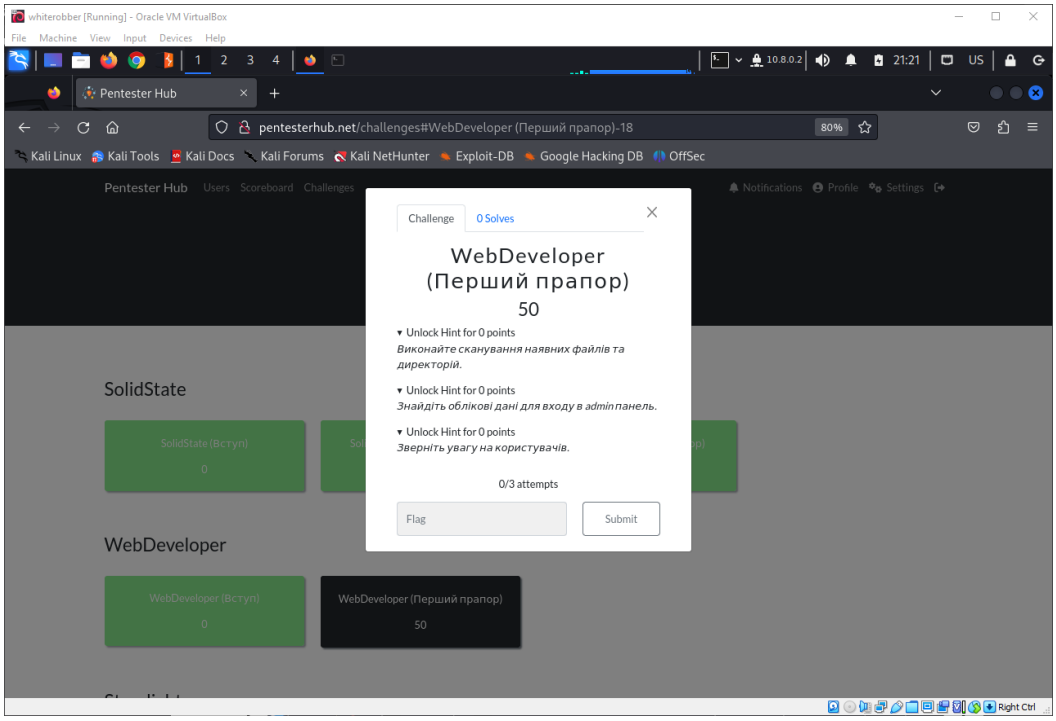


Рисунок 3.5.41 – Друге завдання WebDeveloper (Перший прапор)

Звертаючи увагу на першу безкоштовну підказку можна зробити висновок, що потрібно виконати сканування сканування наявних файлів та директорій.

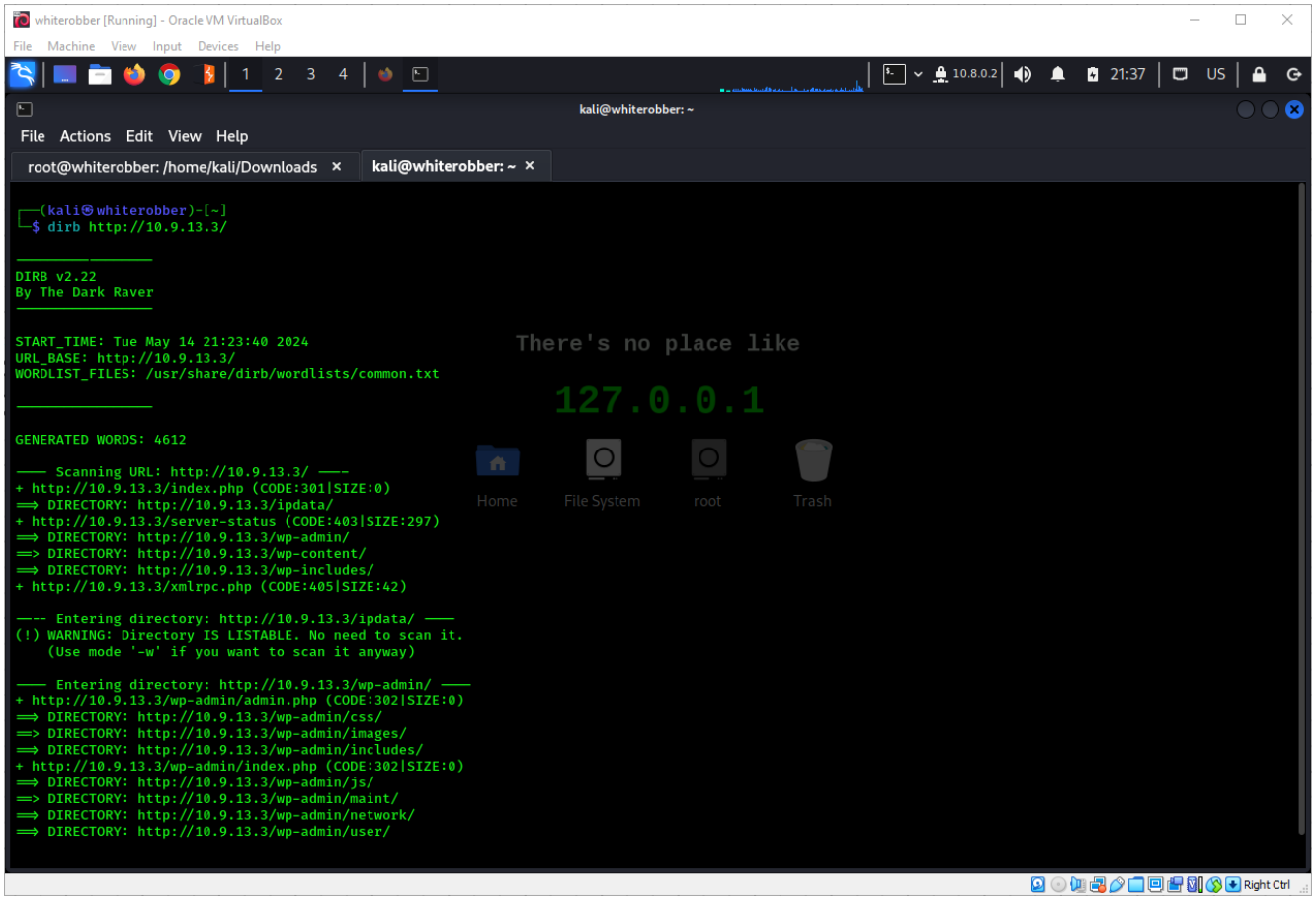


Рисунок 3.5.42 – Результат сканування наявних файлів та директорій за допомогою інструмента dirb

Звертаючи увагу на результат сканування, можна зробити висновок, що сайт використовує CMS WordPress, але найбільший інтерес викликає директорія ipdata, спробуємо її переглянути.

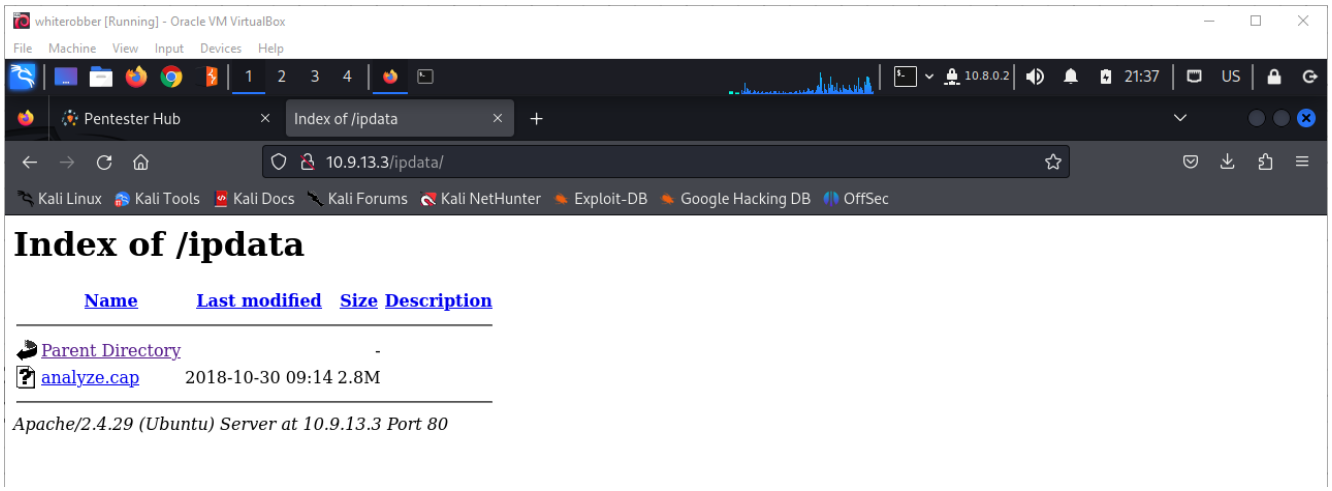


Рисунок 3.5.43 – Перегляд директорії ipdata

Всередині директорії ipdata знаходиться файл analyze.cap, завантажимо даний файл та спробуємо проаналізувати.

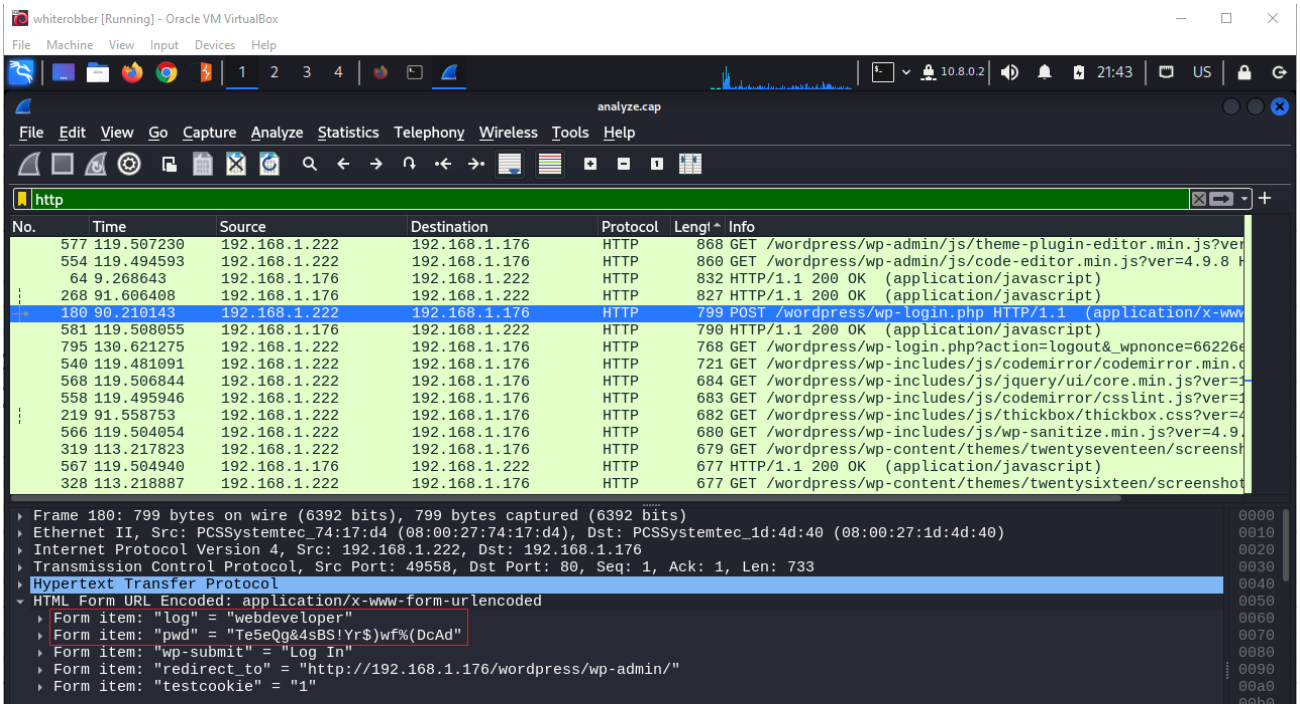


Рисунок 3.5.44 – Файл analyze.cap успішно проаналізовано за допомогою утиліти Wireshark

Всередині файлу analyze.cap були знайдені облікові дані адміністратора WordPress, згідно вказаній інформації в другій безкоштовній підказці спробуємо виконати вхід в admin панель.

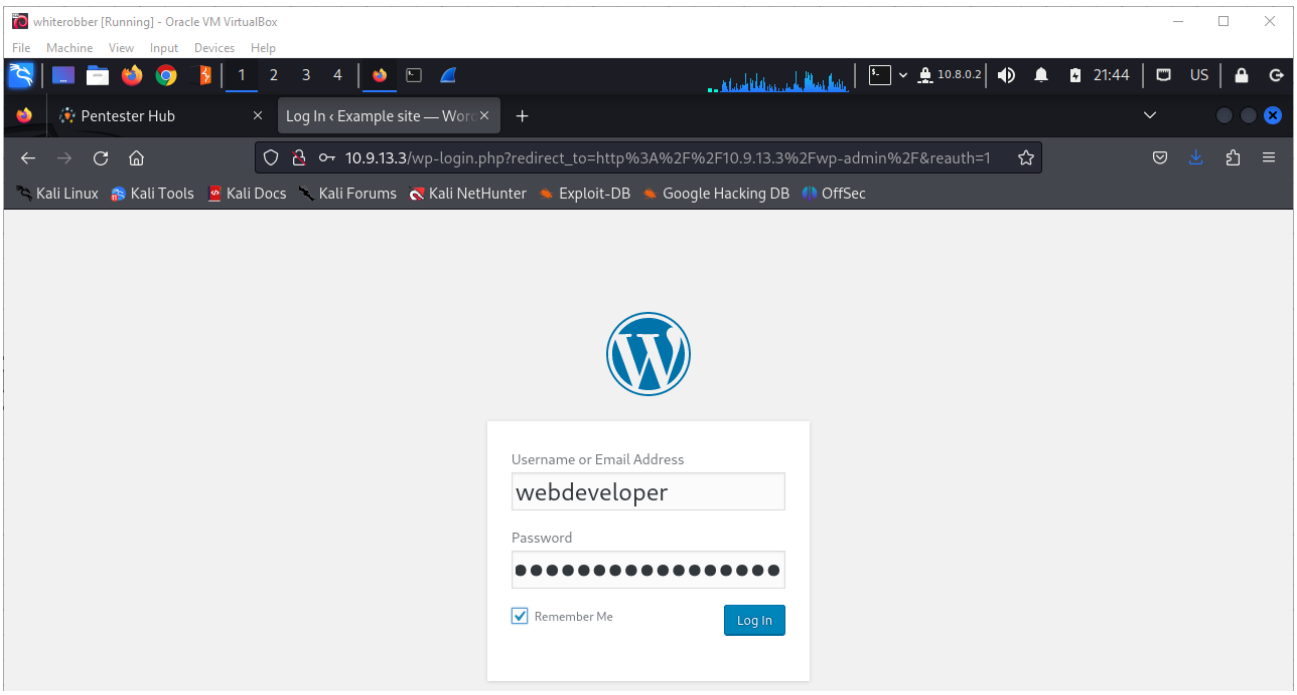


Рисунок 3.5.45 – Виконання входу в admin панель

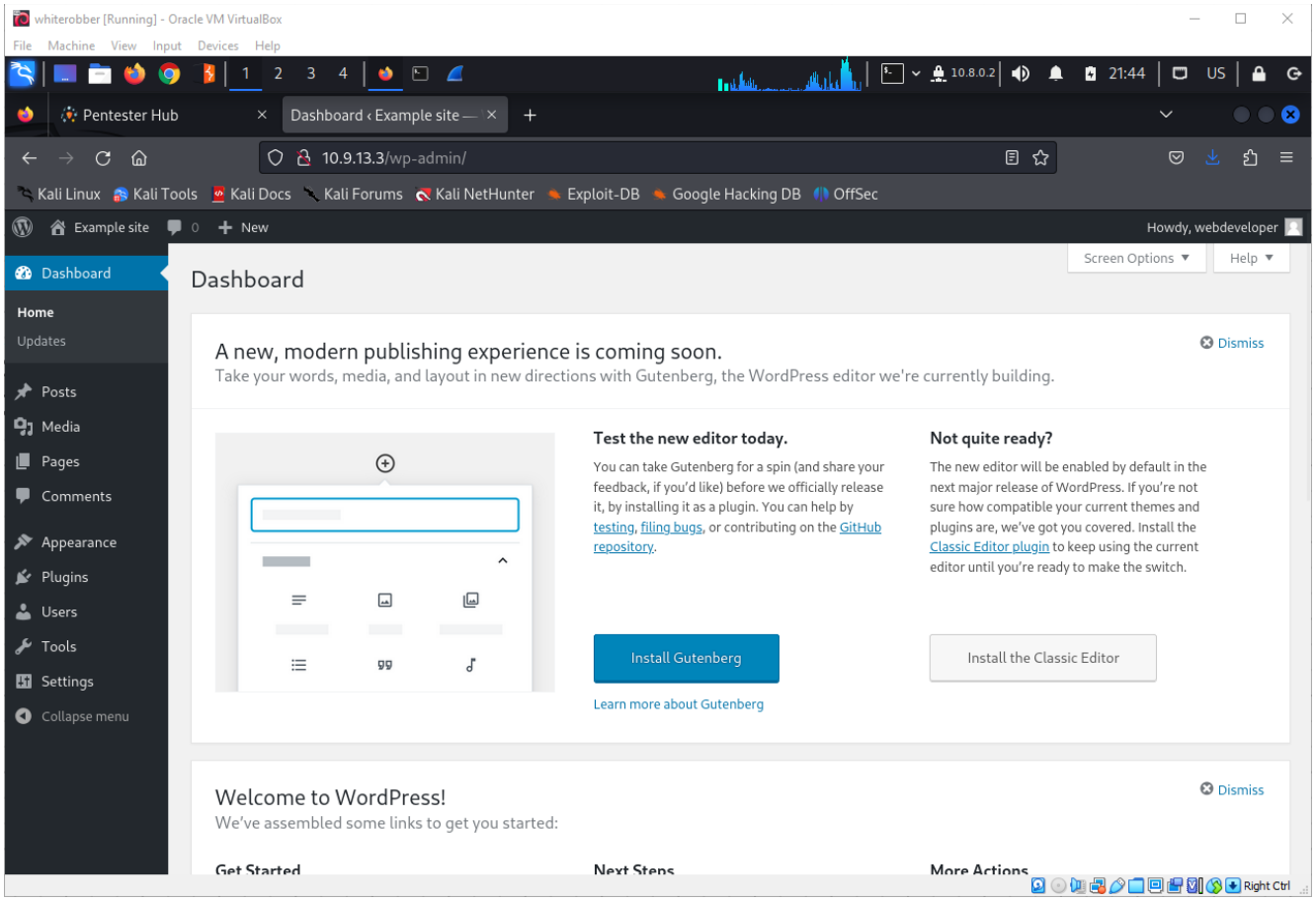


Рисунок 3.5.46 – Вхід успішно виконано, отримано доступ до admin панелі

Звертаючи увагу на третю безкоштовну підказку можна зробити висновок, що потрібно переглянути вкладку Users.

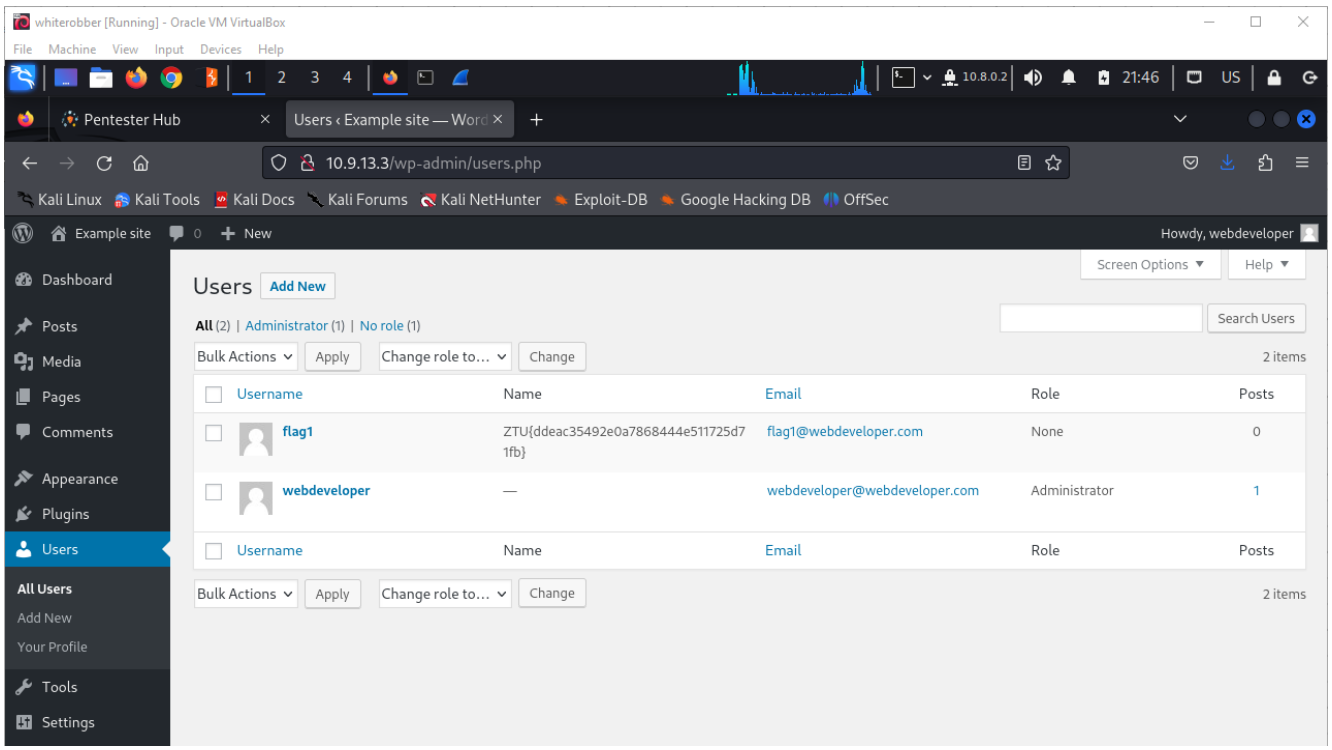


Рисунок 3.5.47 – Вкладка Users, перший прапор успішно знайдено

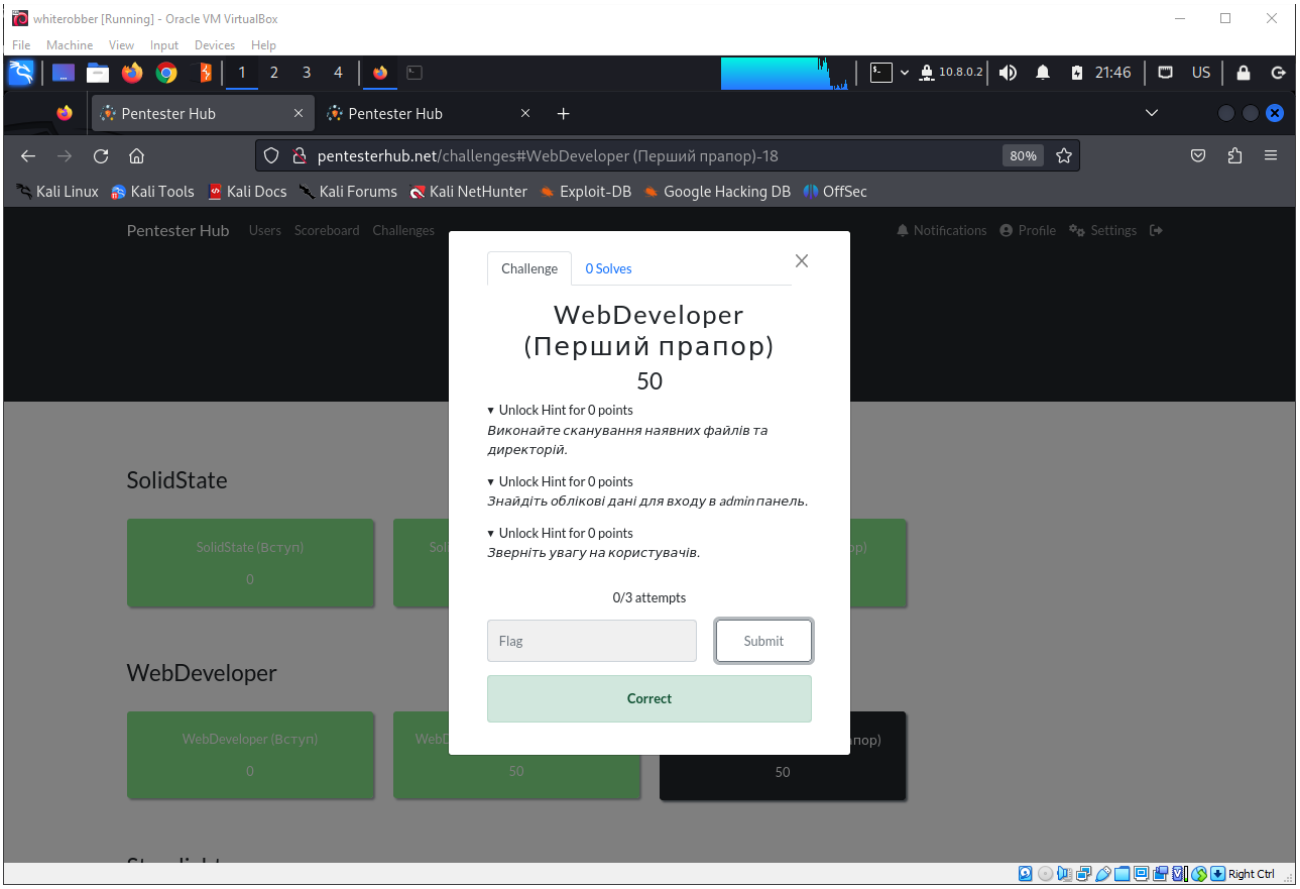


Рисунок 3.5.48 – Прапор успішно надіслано, завдання зараховано

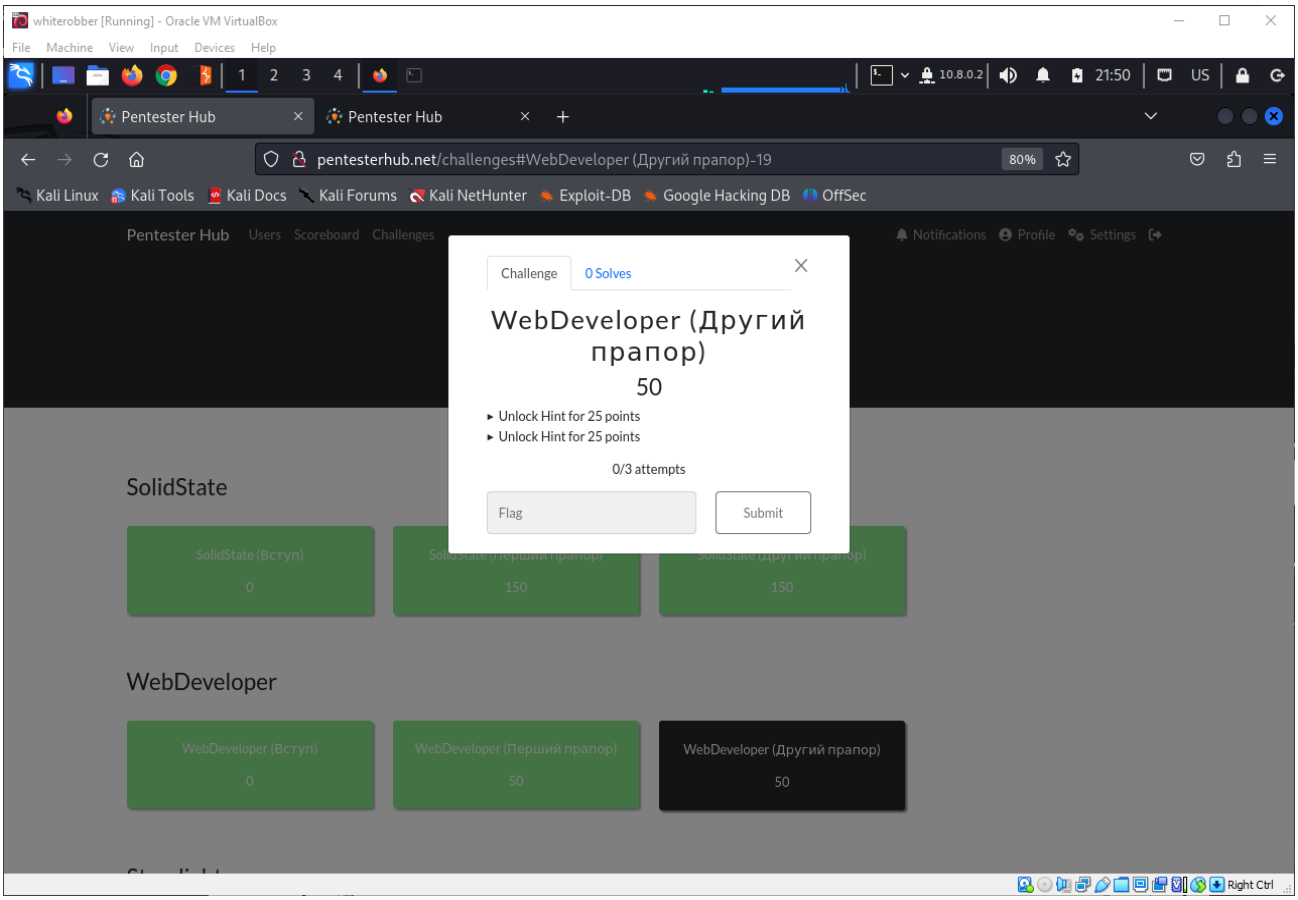


Рисунок 3.5.49 – Третє завдання WebDeveloper (Другий прапор)

Оскільки ми отримали доступ до admin панелі, перейдемо до Edit Themes та виберемо тему Twenty Fifteen, знайдемо серед файлів даної теми файл 404.php та спробуємо додати в нього зворотну оболонку.

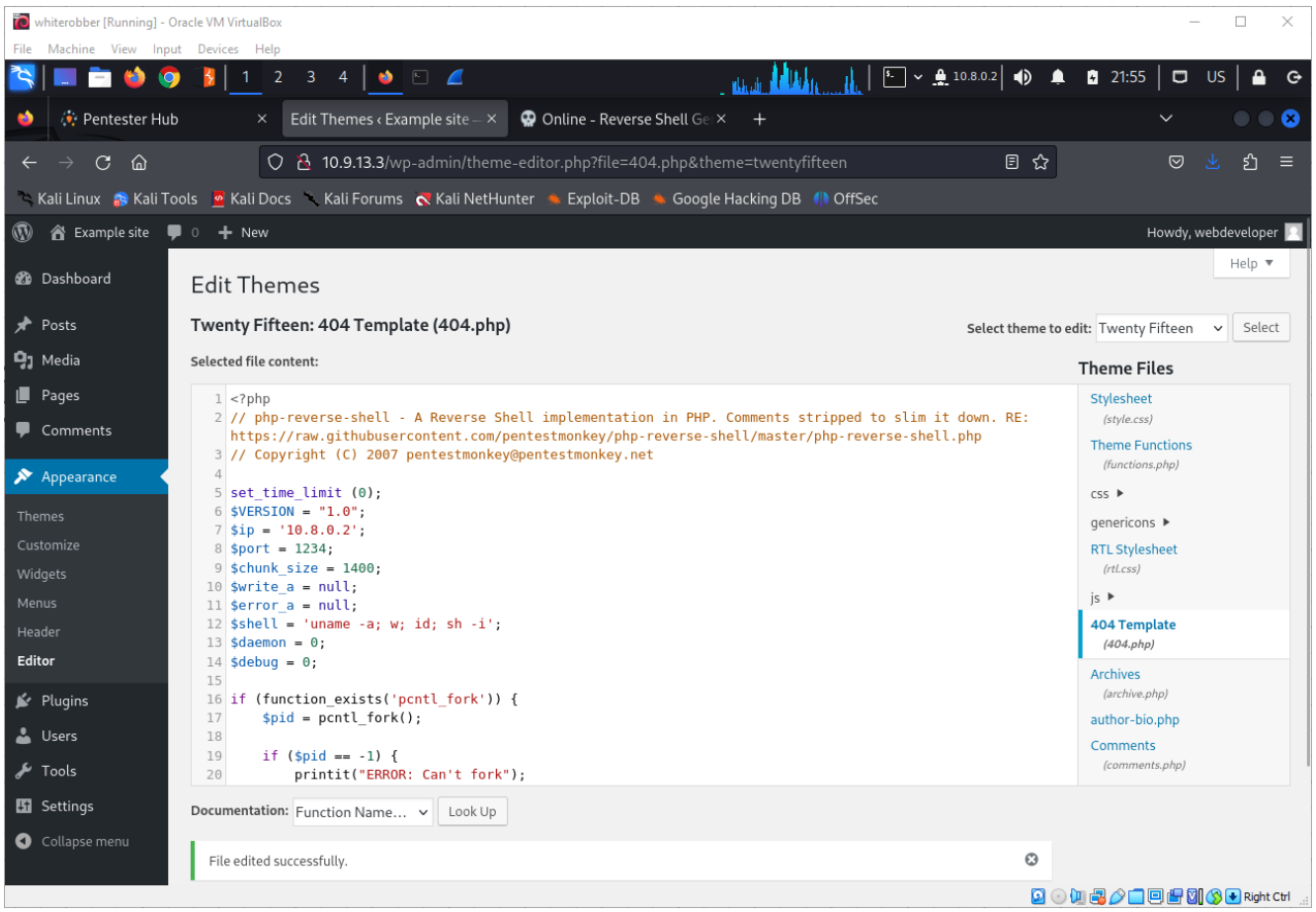


Рисунок 3.5.50 – Редагування файлу 404.php, додання зворотної оболонки

Зворотну оболонку успішно додано, створимо нове вікно терміналу та почнемо прослуховувати встановлений порт 1234, для активації зворотної оболонки перейдемо за наступним шляхом у пошуковому рядку браузера:

– 10.9.13.3/wp-content/themes/twentyfifteen/404.php;

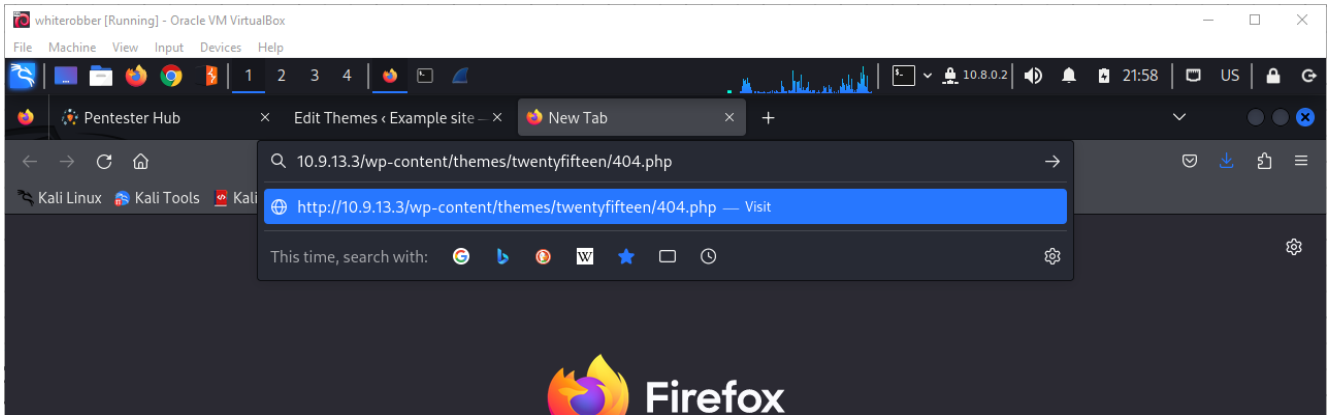


Рисунок 3.5.51 – Активація зворотної оболонки

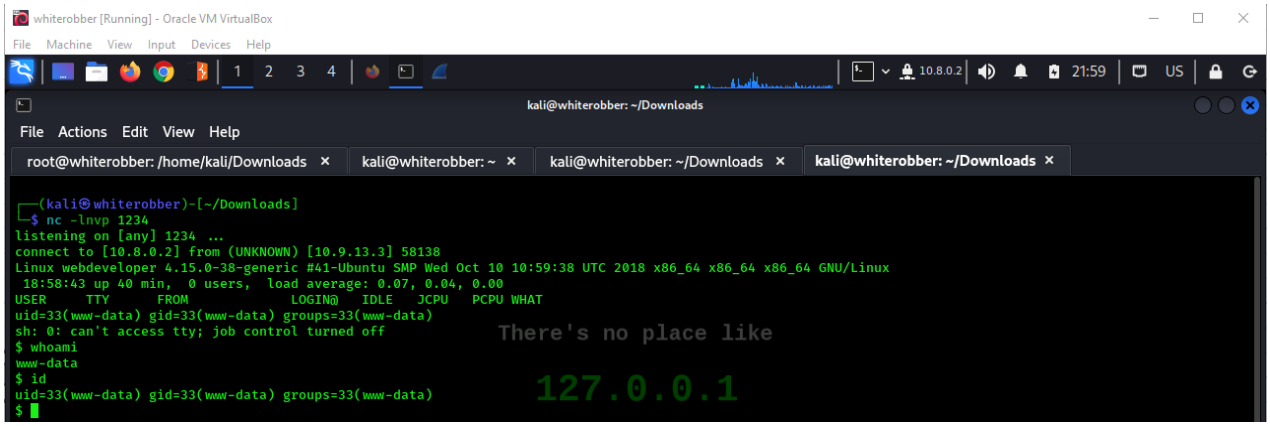


Рисунок 3.5.52 – Зворотна оболонка спрацювала, підключення успішне

Оскільки ми є користувачем www-data та сайт використовує CMS WordPress, переглянемо вміст файлу wp-config.php та спробуємо знайти облікові дані.

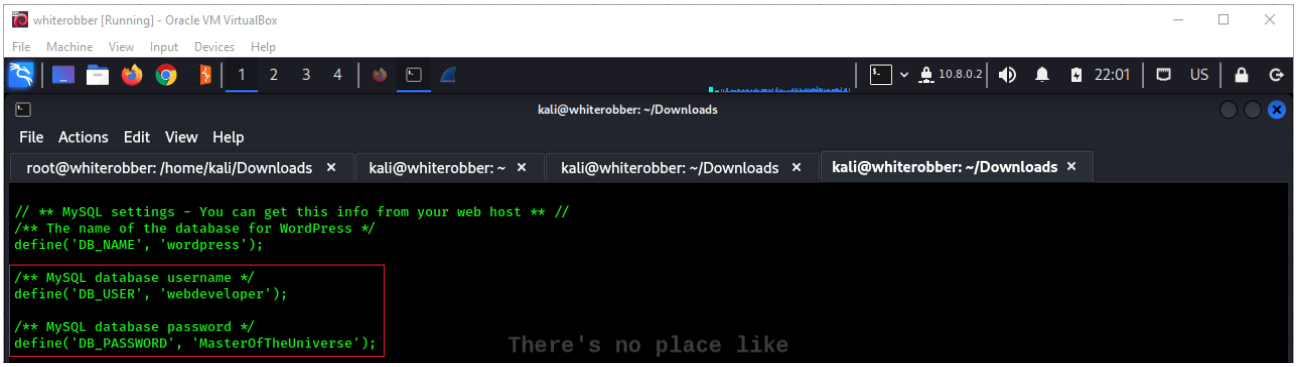


Рисунок 3.5.53 – Знайдено облікові дані MySQL користувача webdeveloper

Незважаючи на те, що були знайдені облікові дані MySQL, спробуємо виконати підключення по SSH використовуючи ці облікові дані.

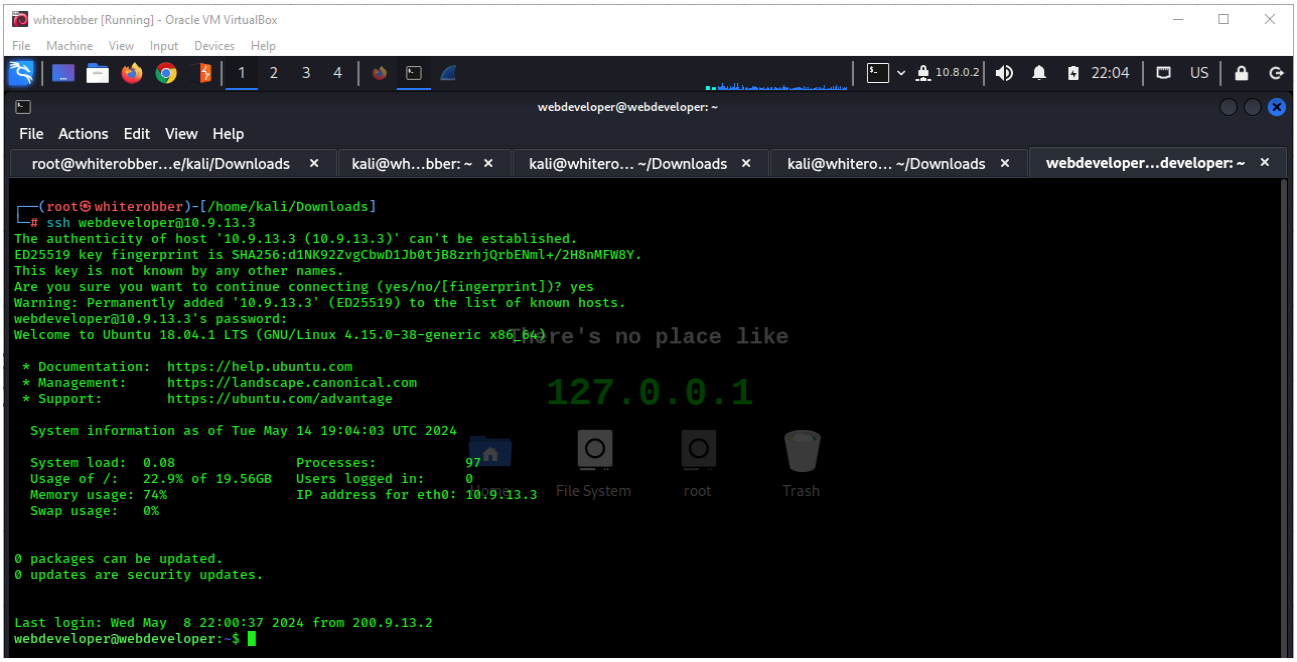


Рисунок 3.5.54 – Виконуємо підключення по SSH, підключення успішне

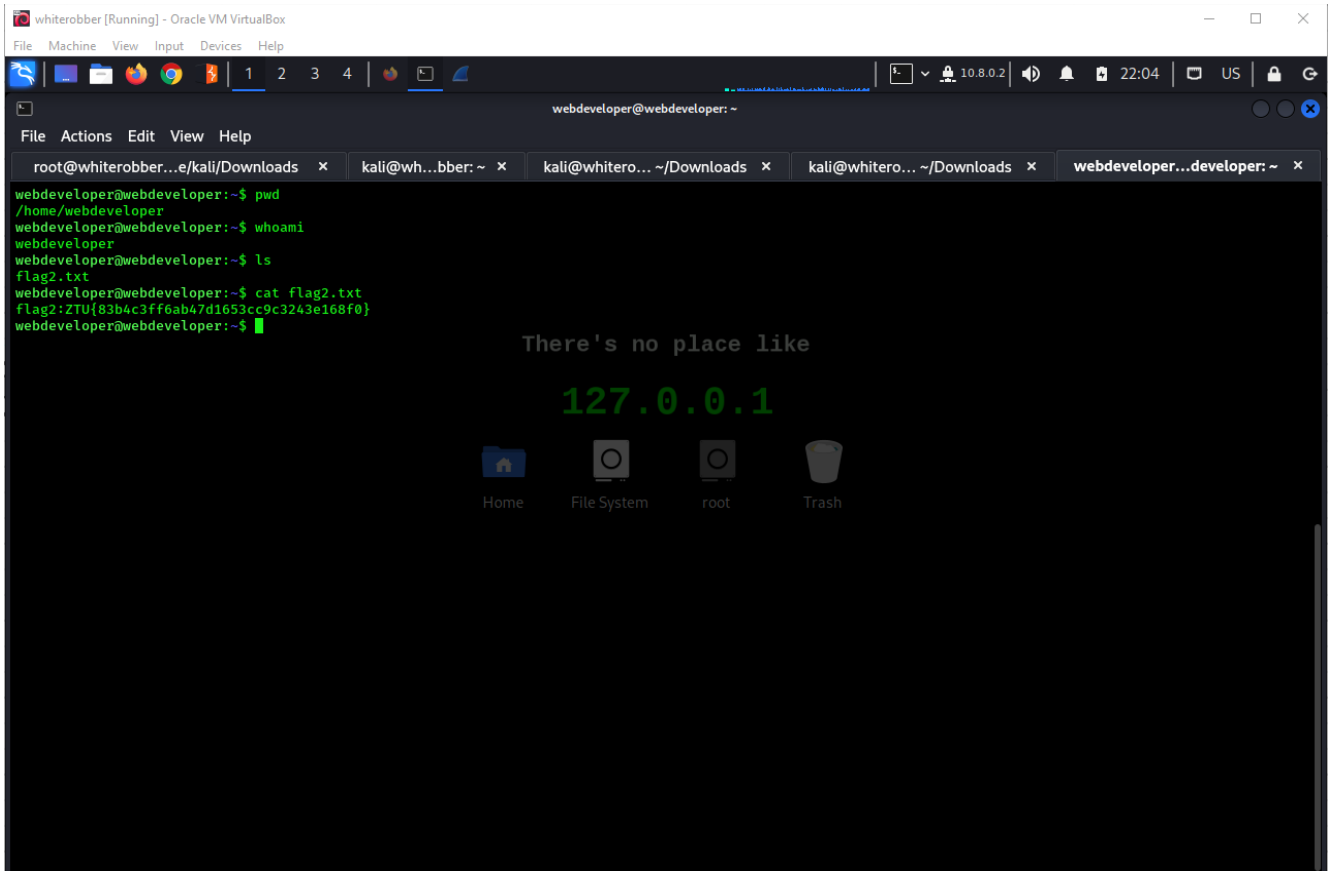


Рисунок 3.5.55 – Другий прапор успішно знайдено

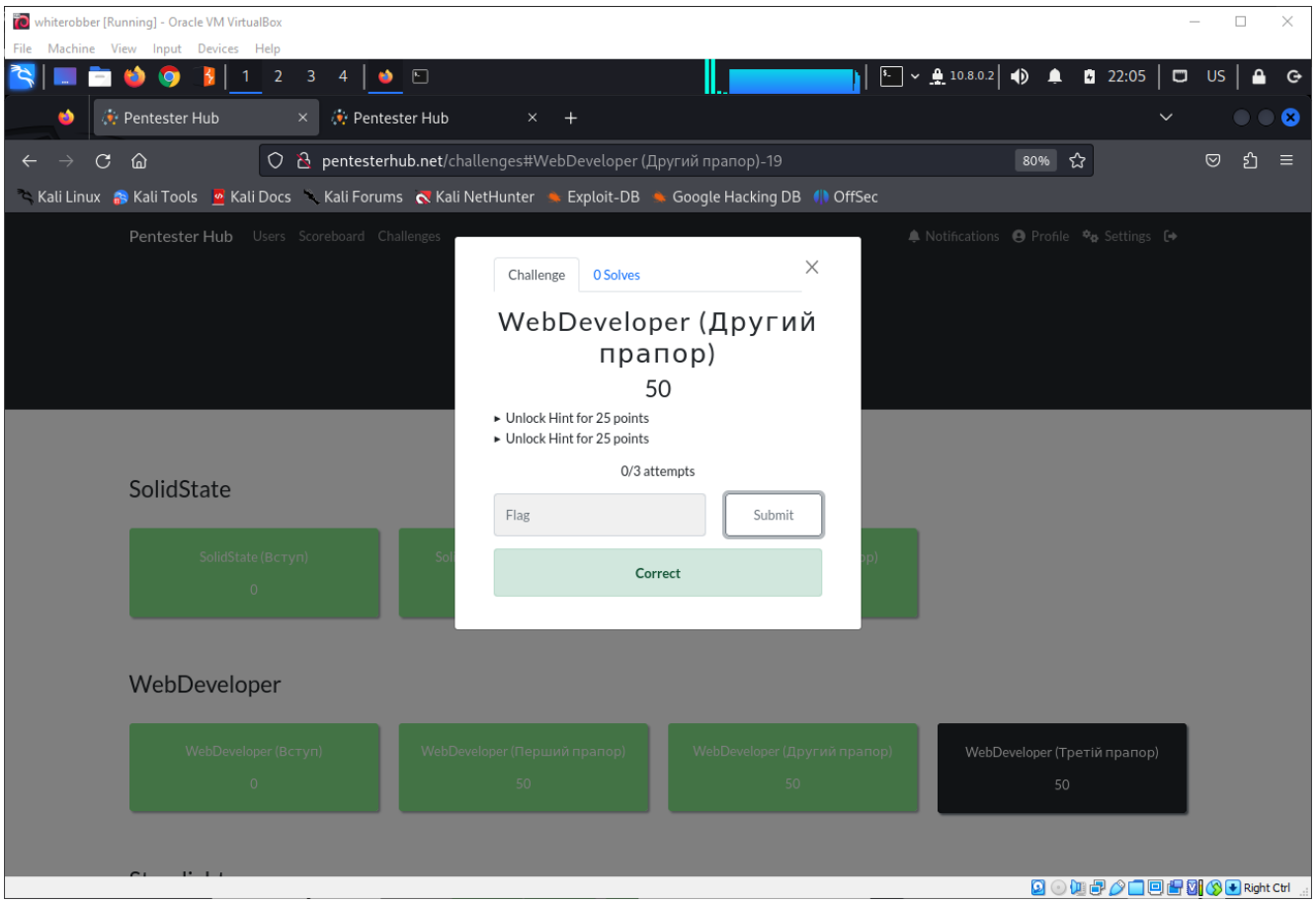


Рисунок 3.5.56 – Прапор успішно надіслано, завдання зараховано

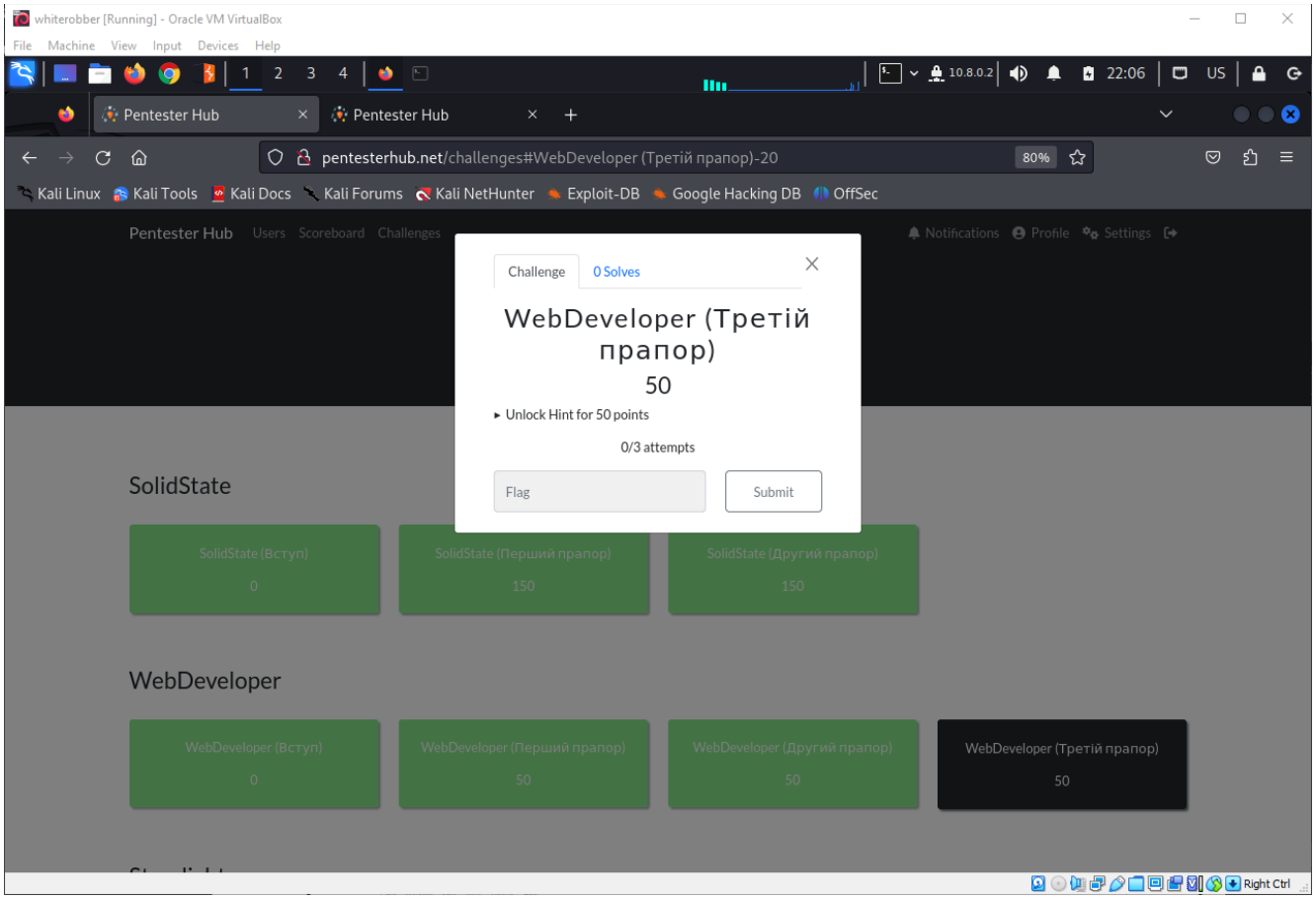


Рисунок 3.5.57 – Четверте/фінальне завдання WebDeveloper (Третій прапор)

Успішне підключення по SSH дає змогу перевірити, які команди користувач webdeveloper може запускати від імені root.

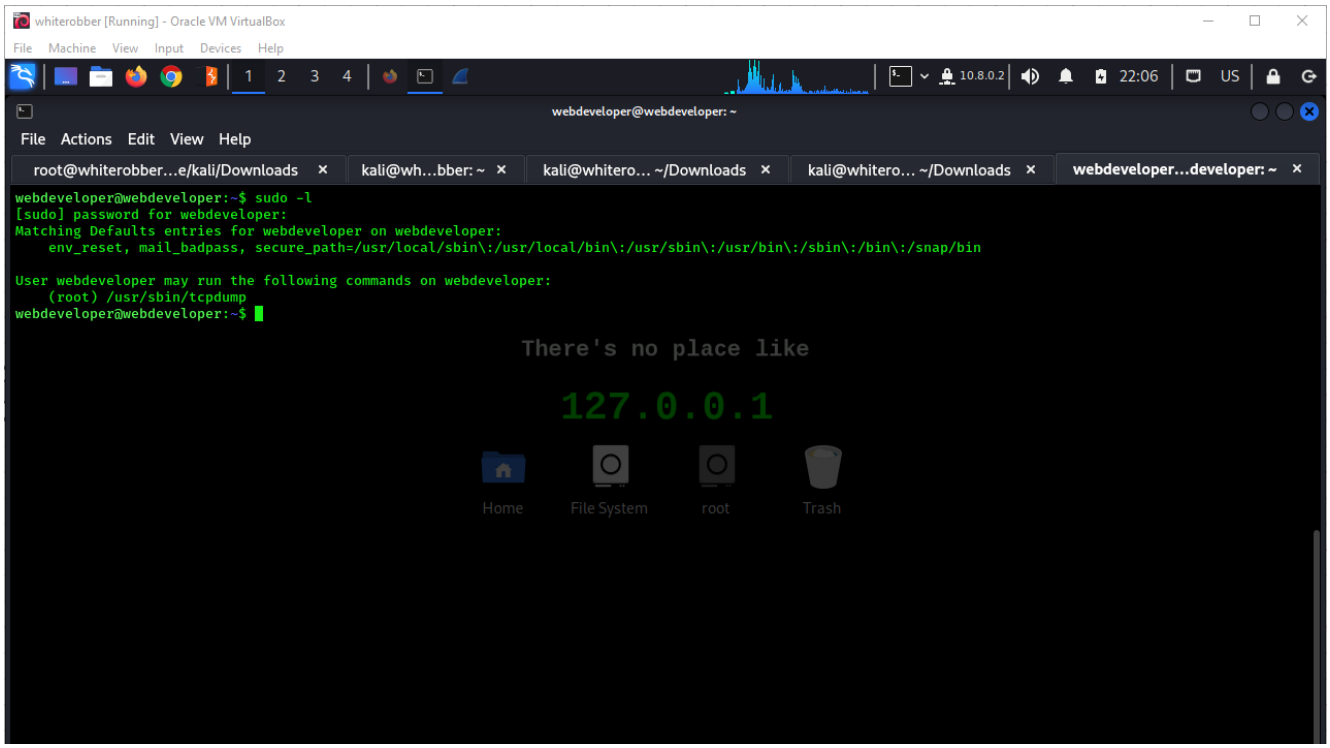


Рисунок 3.5.58 – Виконання команди sudo -l

Вивід sudo -l показав, що можна використати tcpdump для підвищення привілеїв.

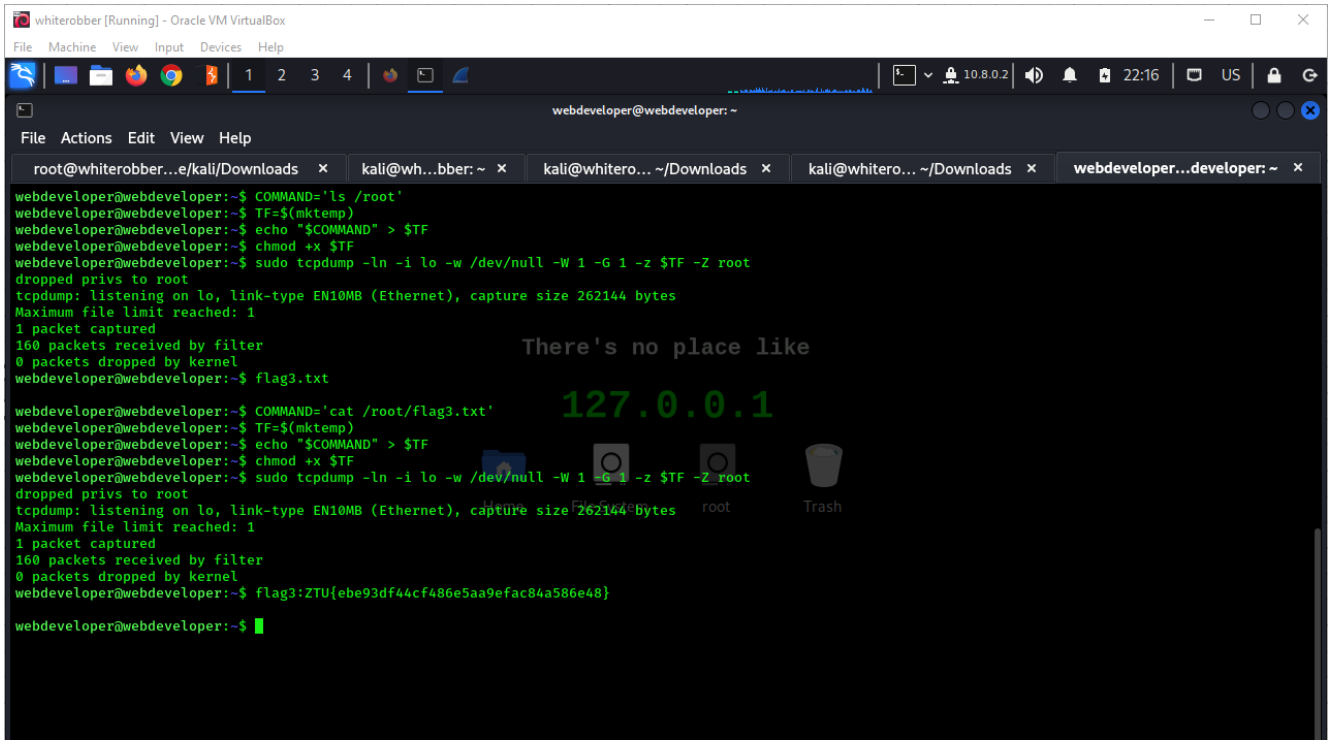


Рисунок 3.5.59 – Підвищення привілеїв успішне, третій прапор успішно знайдено

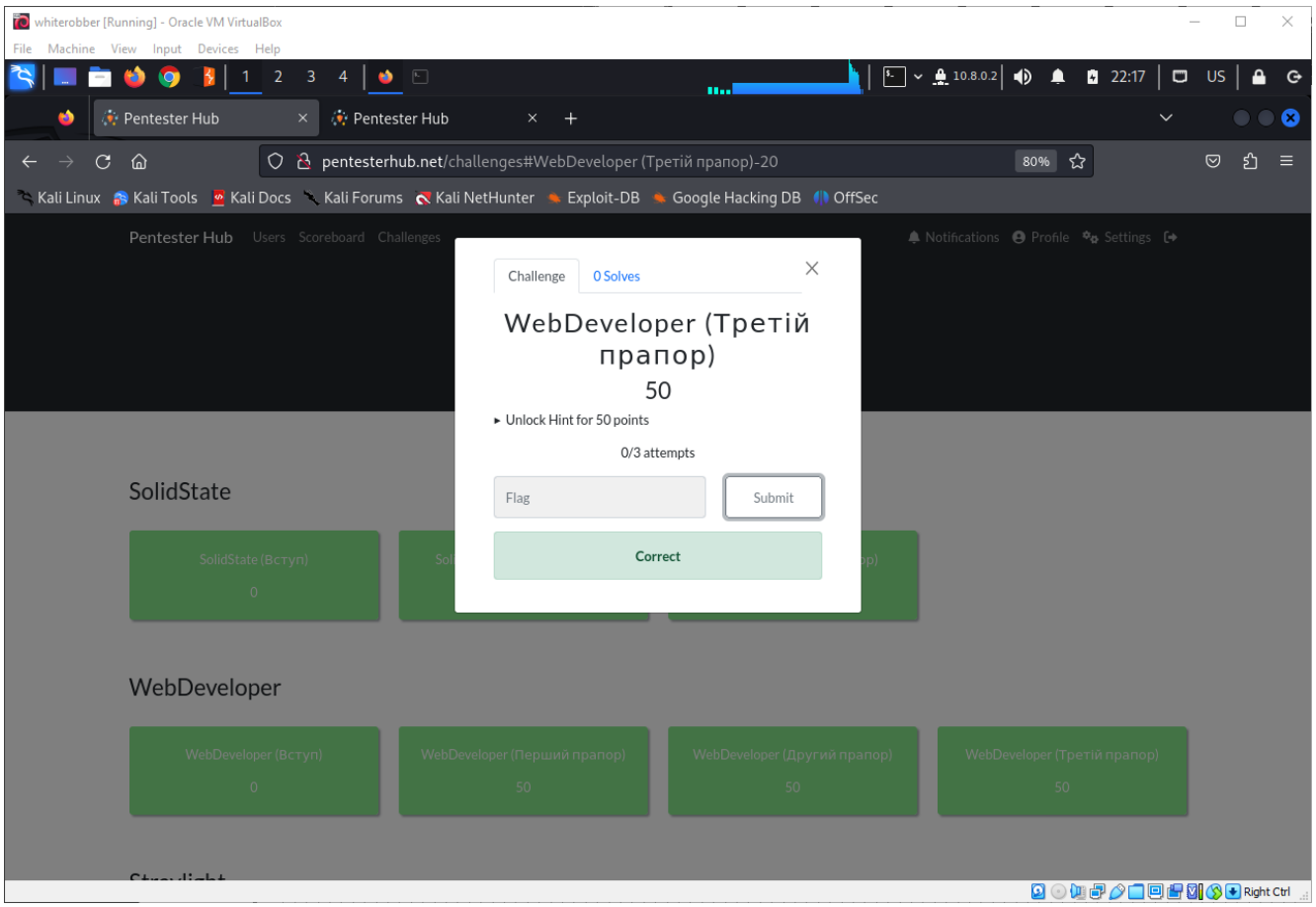


Рисунок 3.5.60 – Прапор успішно надіслано, завдання зараховано

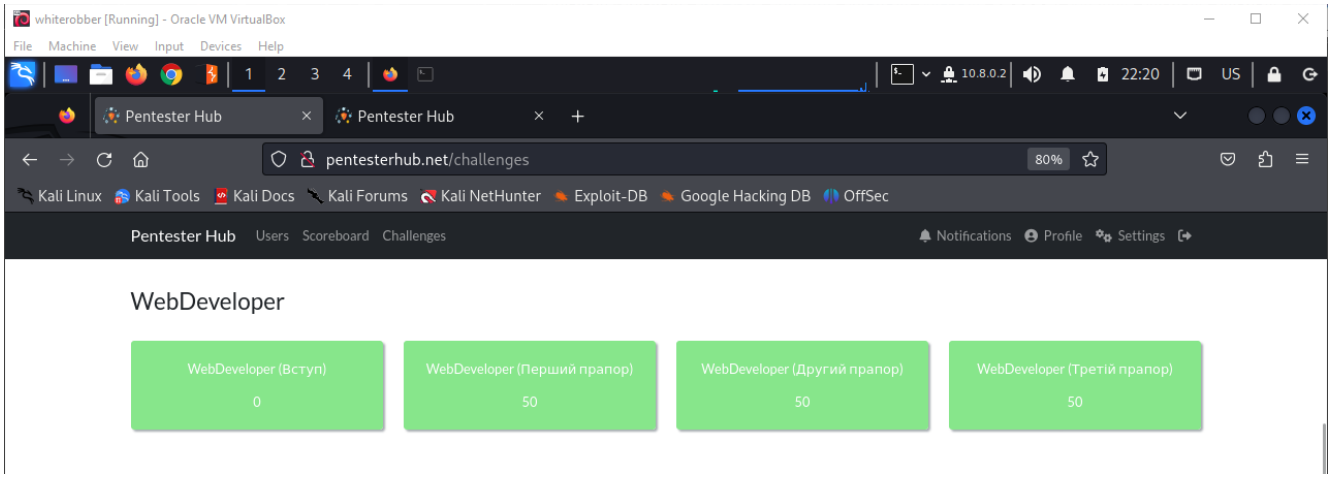


Рисунок 3.5.61 – Серія завдань WebDeveloper протестована (вирішена)

Розпочнемо вирішення серії завдань для машини Straylight.

Зазначимо, процес ознайомлення з описом завдання, виконання підключення до OpenVPN сервера, успішне відкриття веб-сайту машини Straylight та перегляд безкоштовної підказки для того, щоб надіслати прапор для виконання завдання та відкрити наступне завдання в серії аналогічний до SolidState.

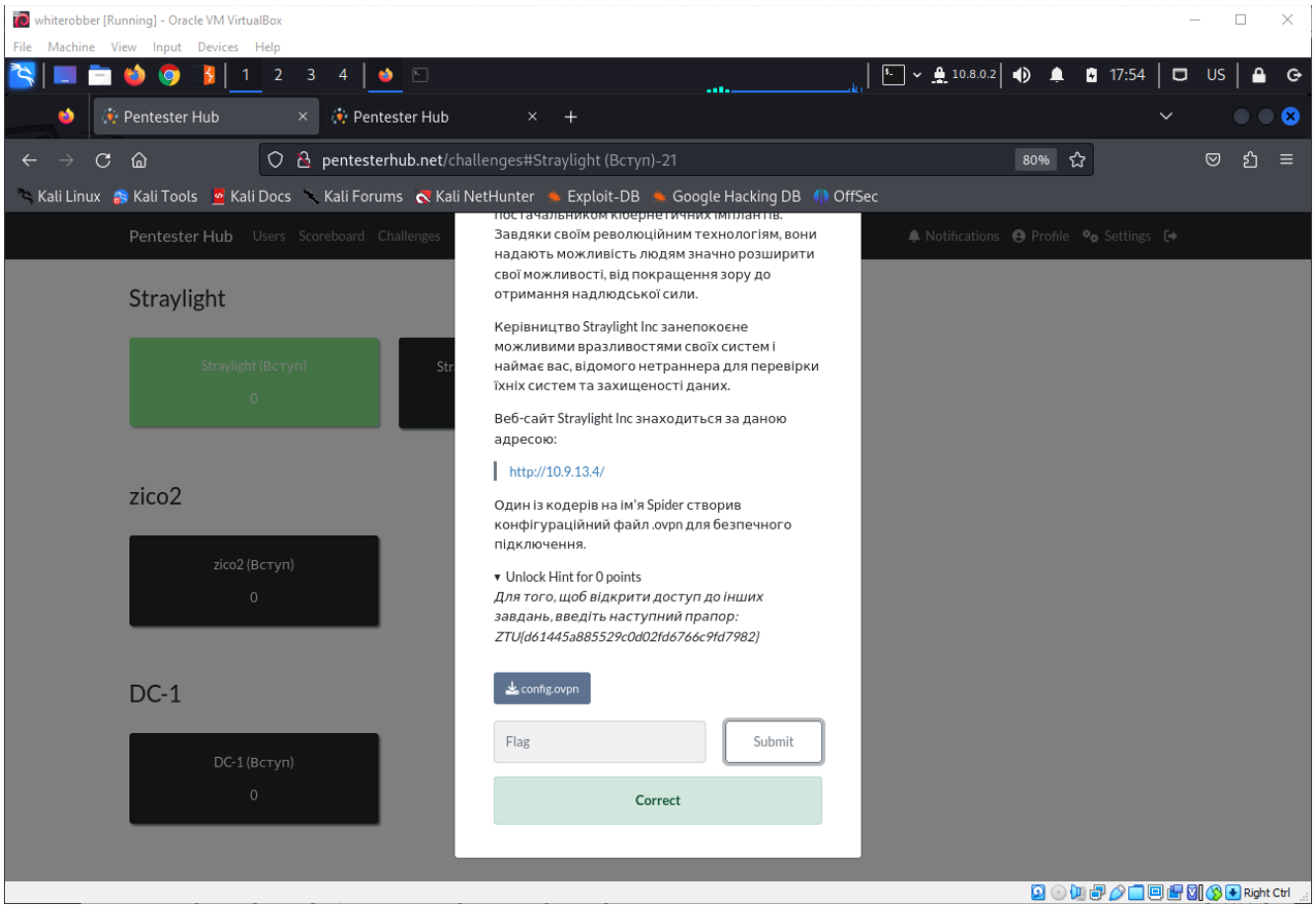


Рисунок 3.5.62 – Перше завдання Straylight (Вступ),
прапор успішно надіслано, завдання зараховано

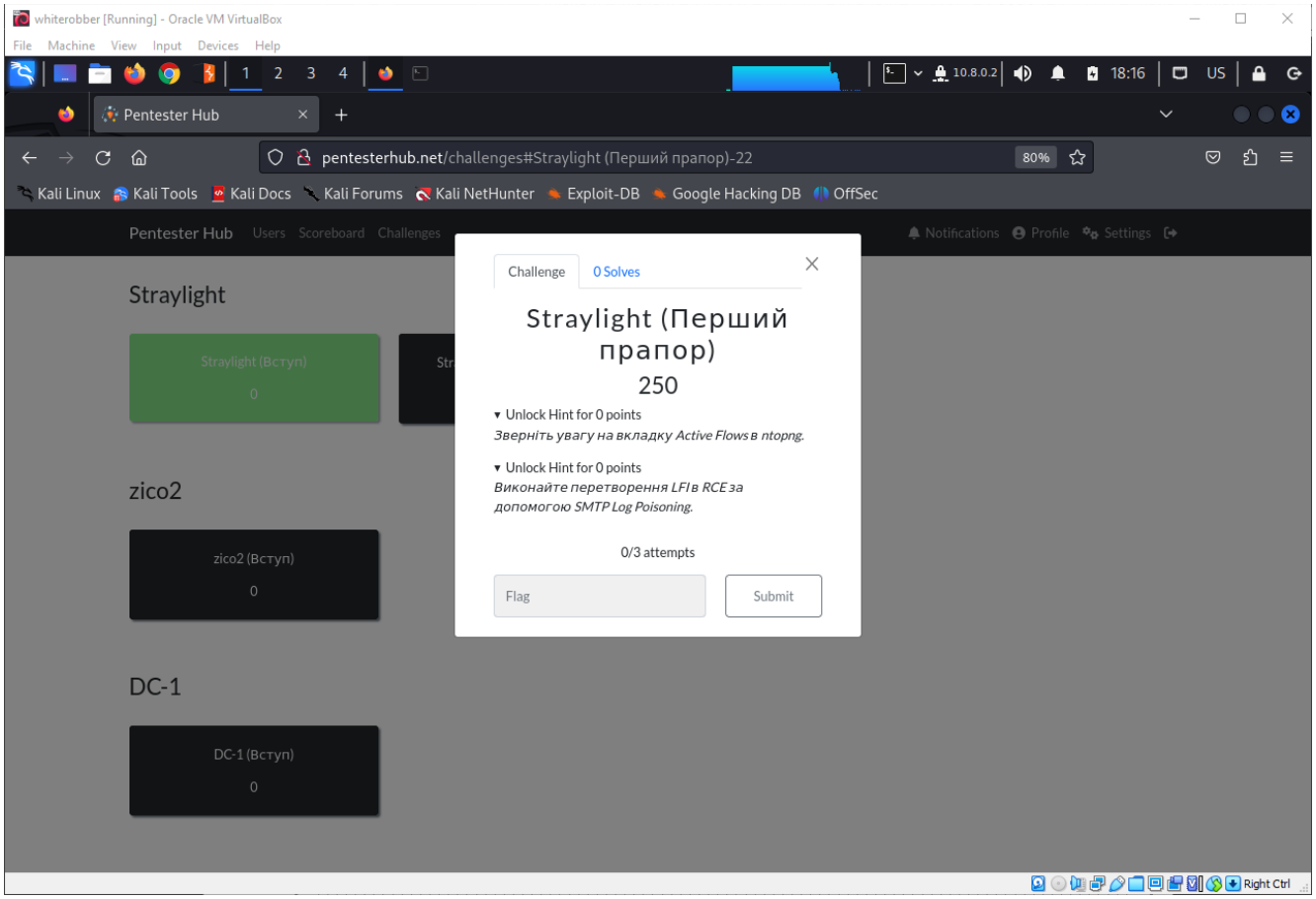


Рисунок 3.5.63 – Друге завдання Straylight (Перший прапор)

Виконаємо сканування відкритих портів.

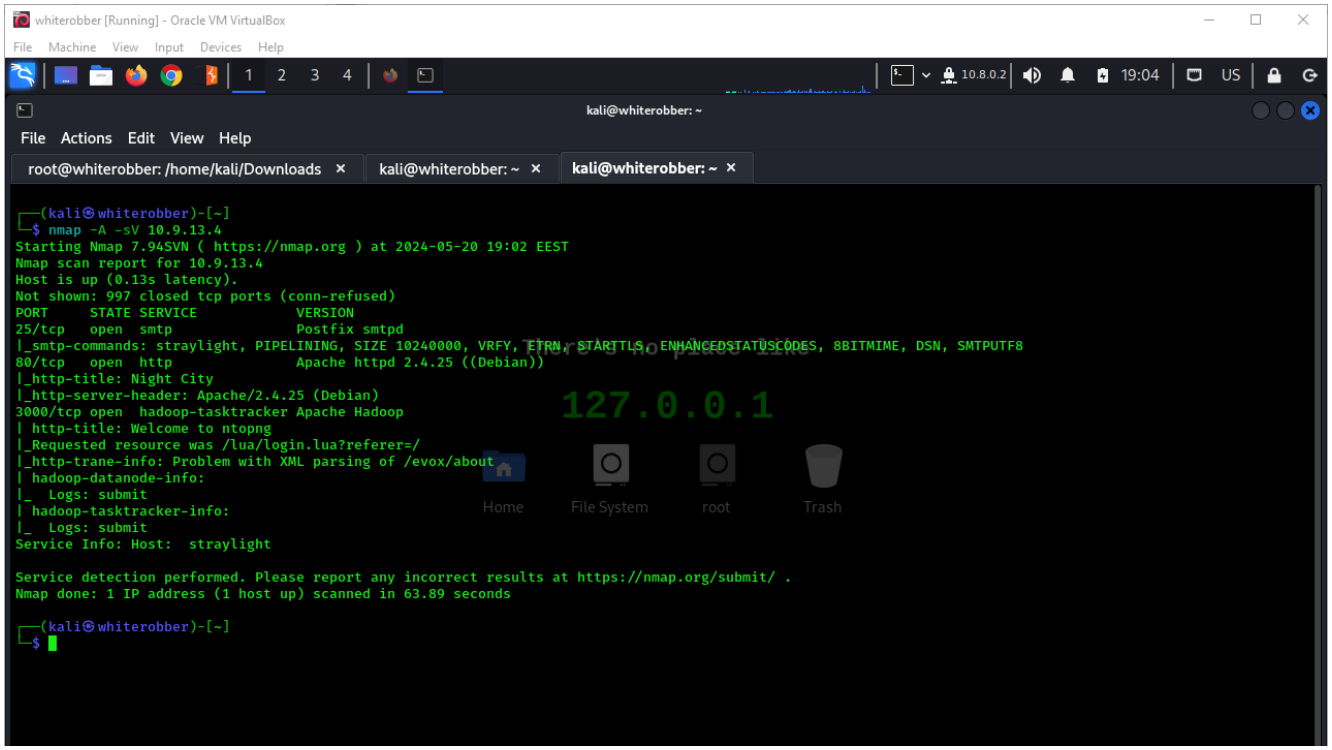


Рисунок 3.5.64 – Результат сканування відкритих портів за допомогою інструмента nmap

Звертаючи увагу на результат сканування, помічено цікавий порт 3000/tcp (ntopng), спробуємо перейти за адресою 10.9.13.4:3000 використовуючи пошуковий рядок браузера.

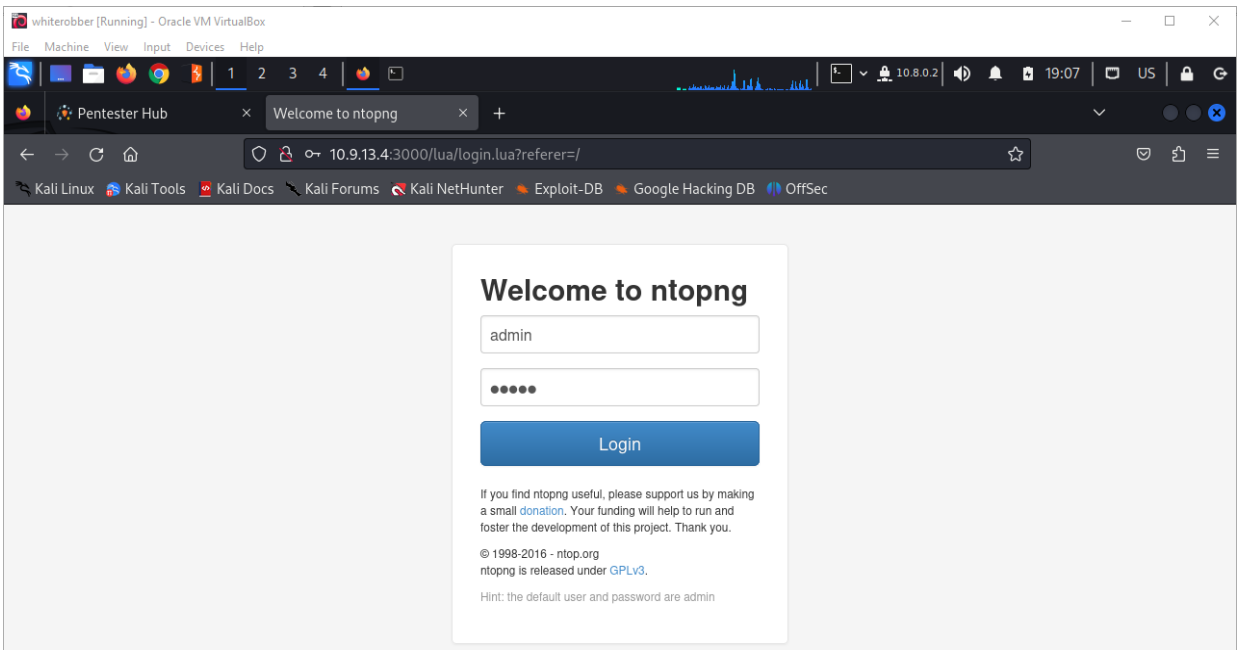


Рисунок 3.5.65 – Результат переходу за адресою 10.9.13.4:3000 використовуючи пошуковий рядок браузера, для виконання входу були використані облікові дані за замовчуванням admin:admin

Згідно вказаній інформації в першій безкоштовній підказці в ntopng потрібно знайти вкладку Active Flows.

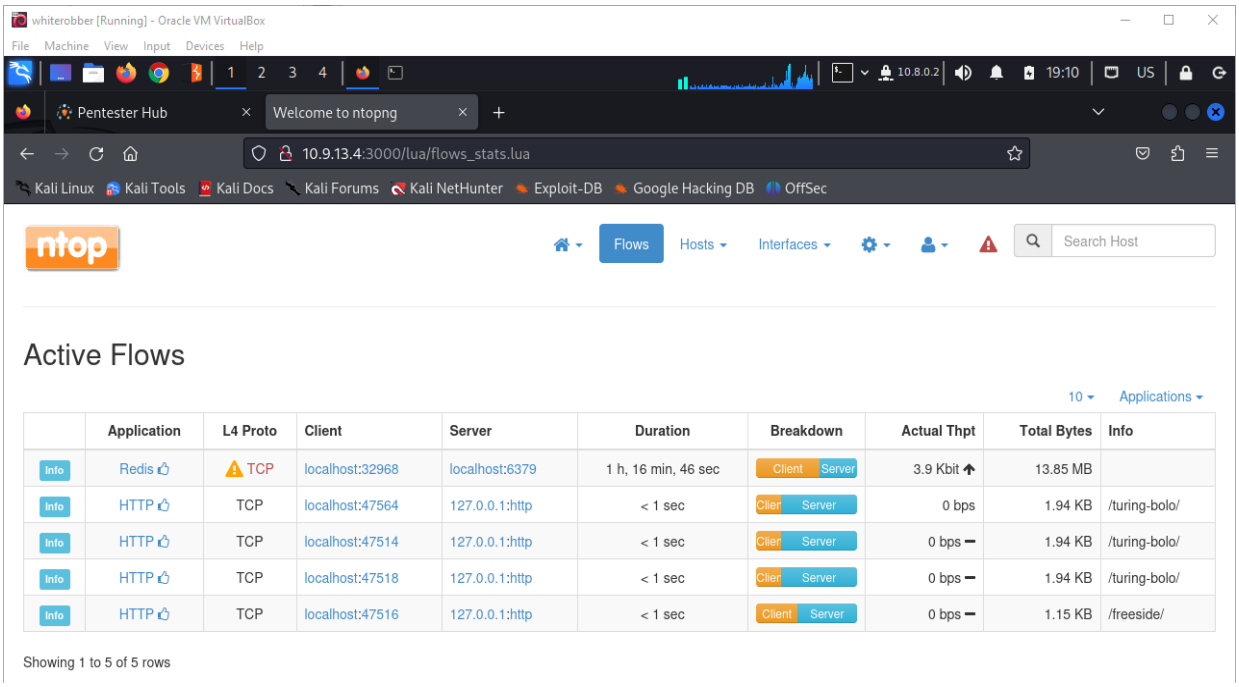


Рисунок 3.5.66 – Вкладка Active Flows

На вкладці Active Flows у стовпці Info зазначена цікава веб-сторінка turing-bolo, переглянемо її.

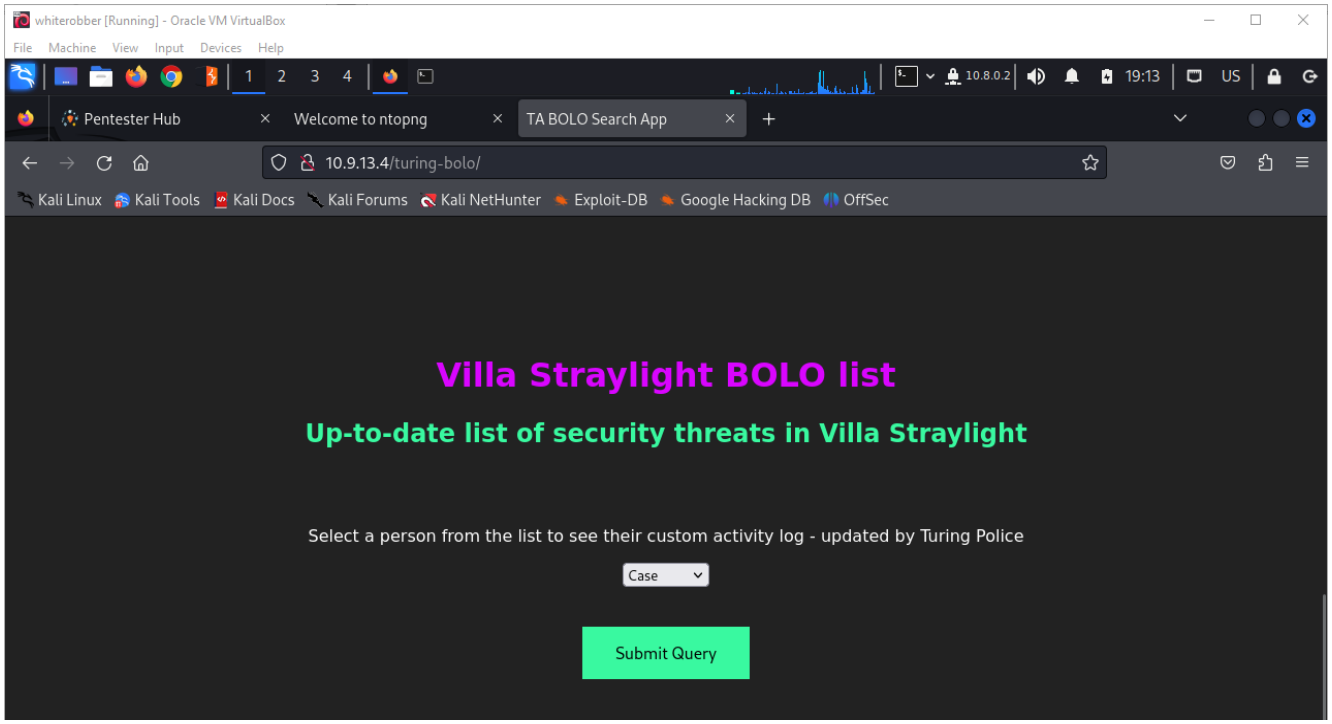


Рисунок 3.5.67 – Перегляд веб-сторінки turing-bolo

Веб-сторінка turing-bolo надає можливість переглядати журнал активності, залишаємо встановлений за замовчуванням параметр «Case» та надсилаємо запит.

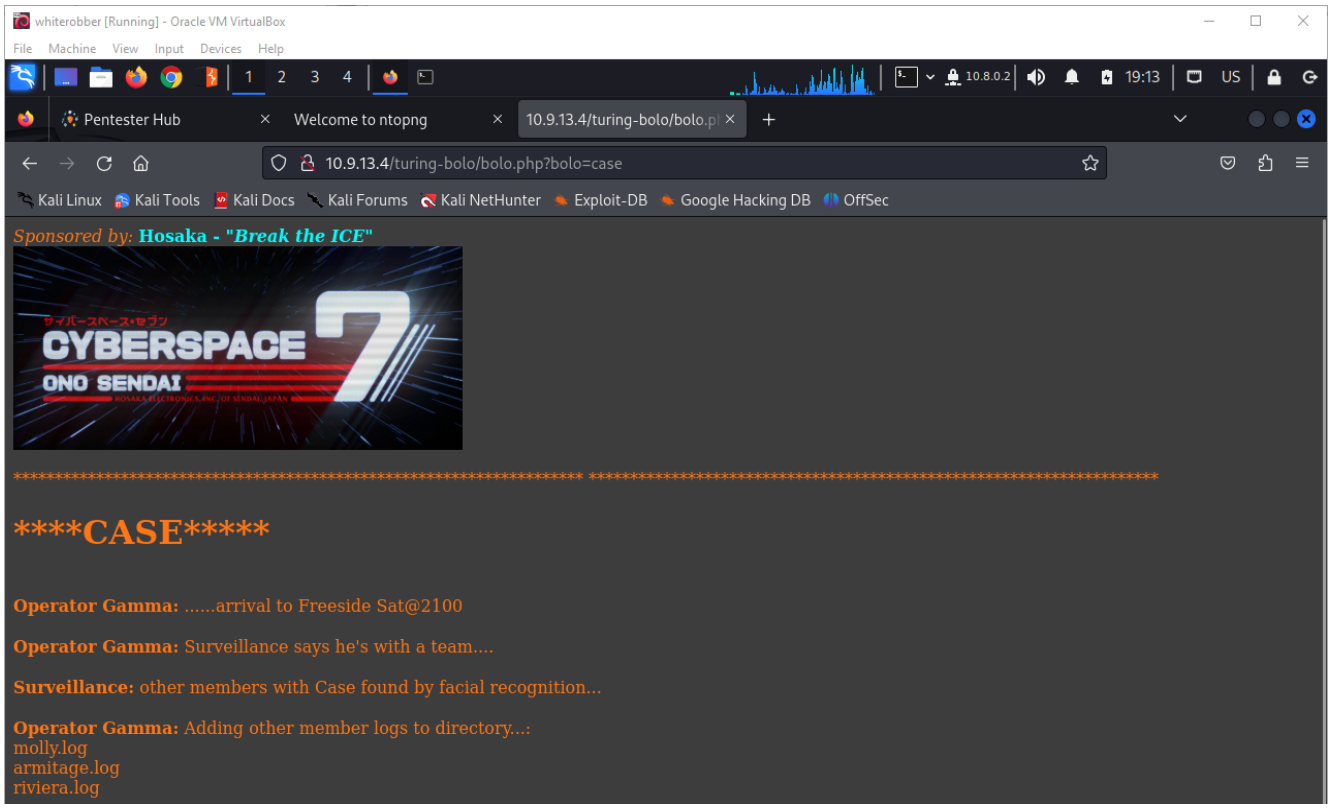


Рисунок 3.5.68 – Відбулося відкриття сторінки bolo.php

На сторінці bolo.php наявний параметр bolo здається, що він додає в кінець .log, а потім виводиться вміст журналу активності. Все це вказує на те, що можливо використати LFI.

Повертаючись до результату сканування відкритих портів звертаємо увагу, на порт 25/tcp (smtp), спробуємо використовуючи LFI вивести вміст журналу активності пошти.

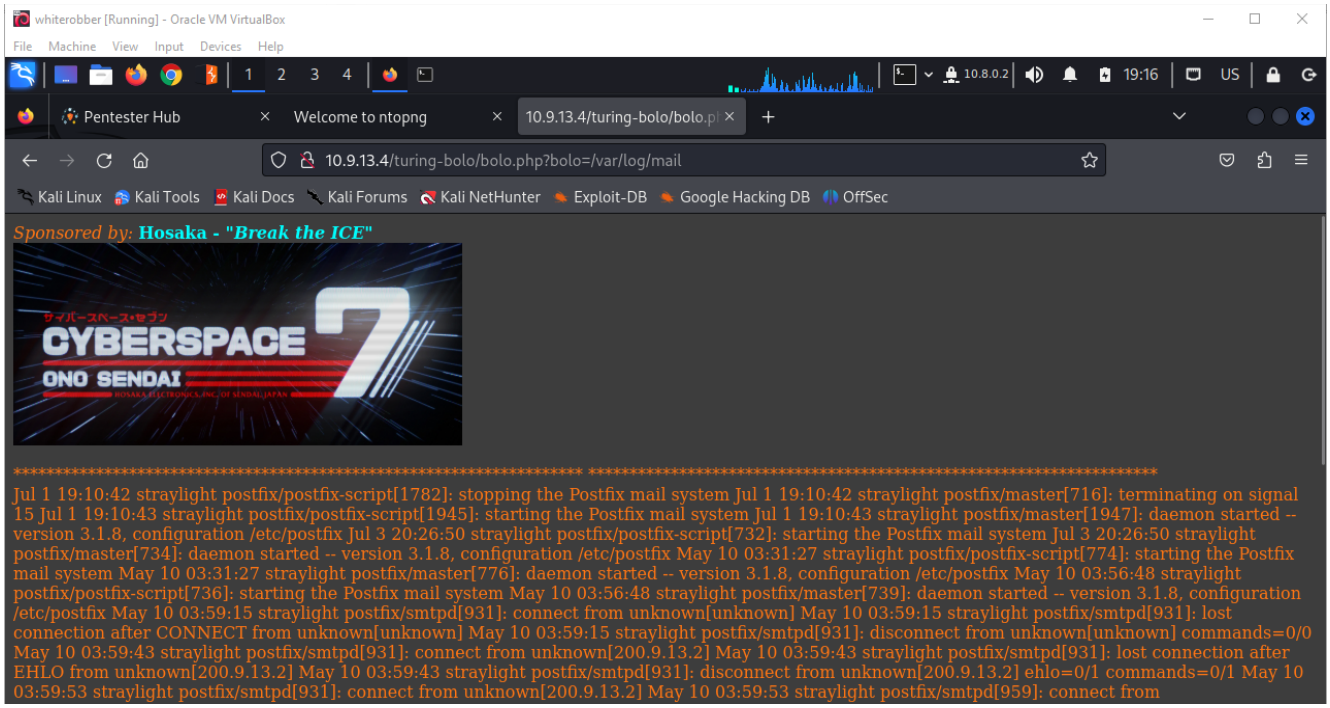


Рисунок 3.5.69 – Вміст журналу активності пошти

Згідно вказаній інформації в другій безкоштовній підказці виконаємо перетворення LFI в RCE за допомогою SMTP Log Poisoning, для цього потрібно підключитися до порту 25 використовуючи telnet.

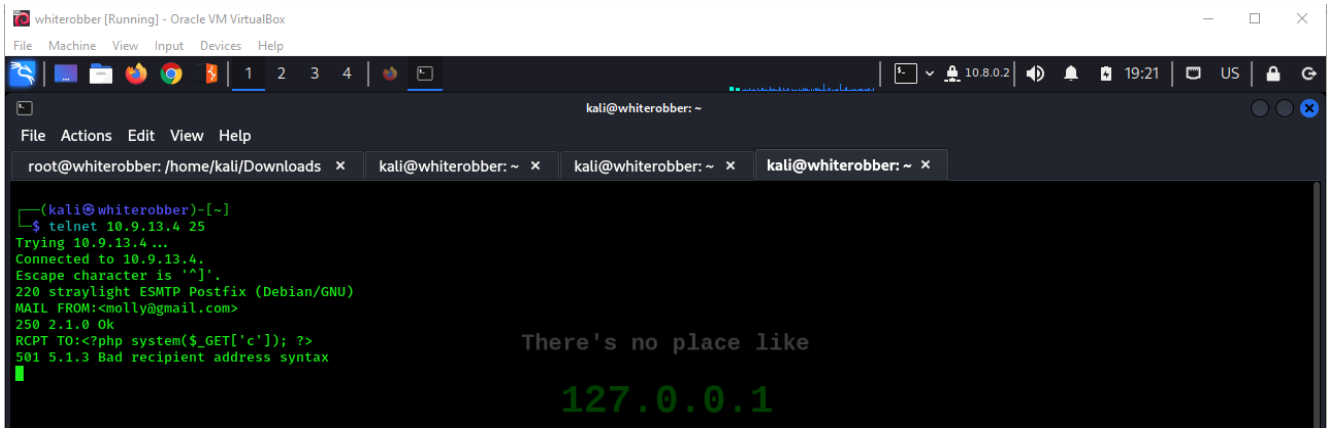


Рисунок 3.5.70 – Підключення до порту 25 використовуючи telnet, перетворення LFI в RCE за допомогою SMTP Log Poisoning

LFI успішно перетворено в RCE для того, щоб перевірити працездатність після var/log/mail додамо &c=.

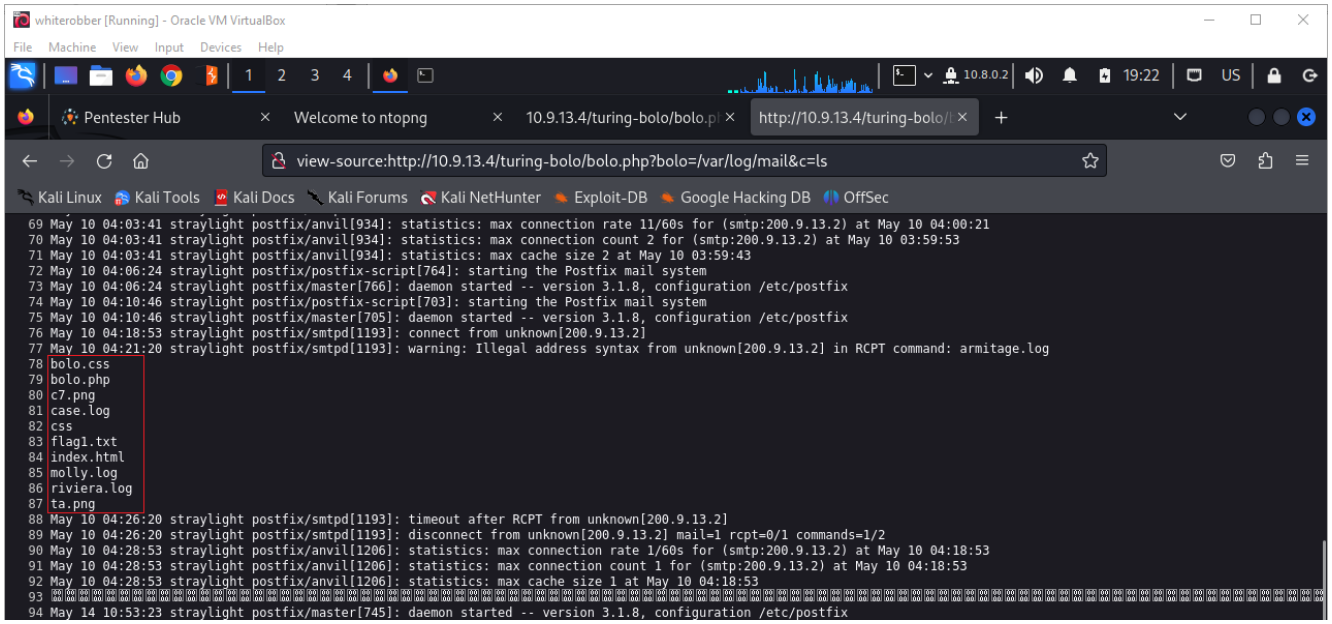


Рисунок 3.5.71 – Виконання команди ls

Виконавши команду ls ми успішно вивели наявні файли та директорії, серед переліку було помічено файл flag1.txt, переглянемо його зміст.

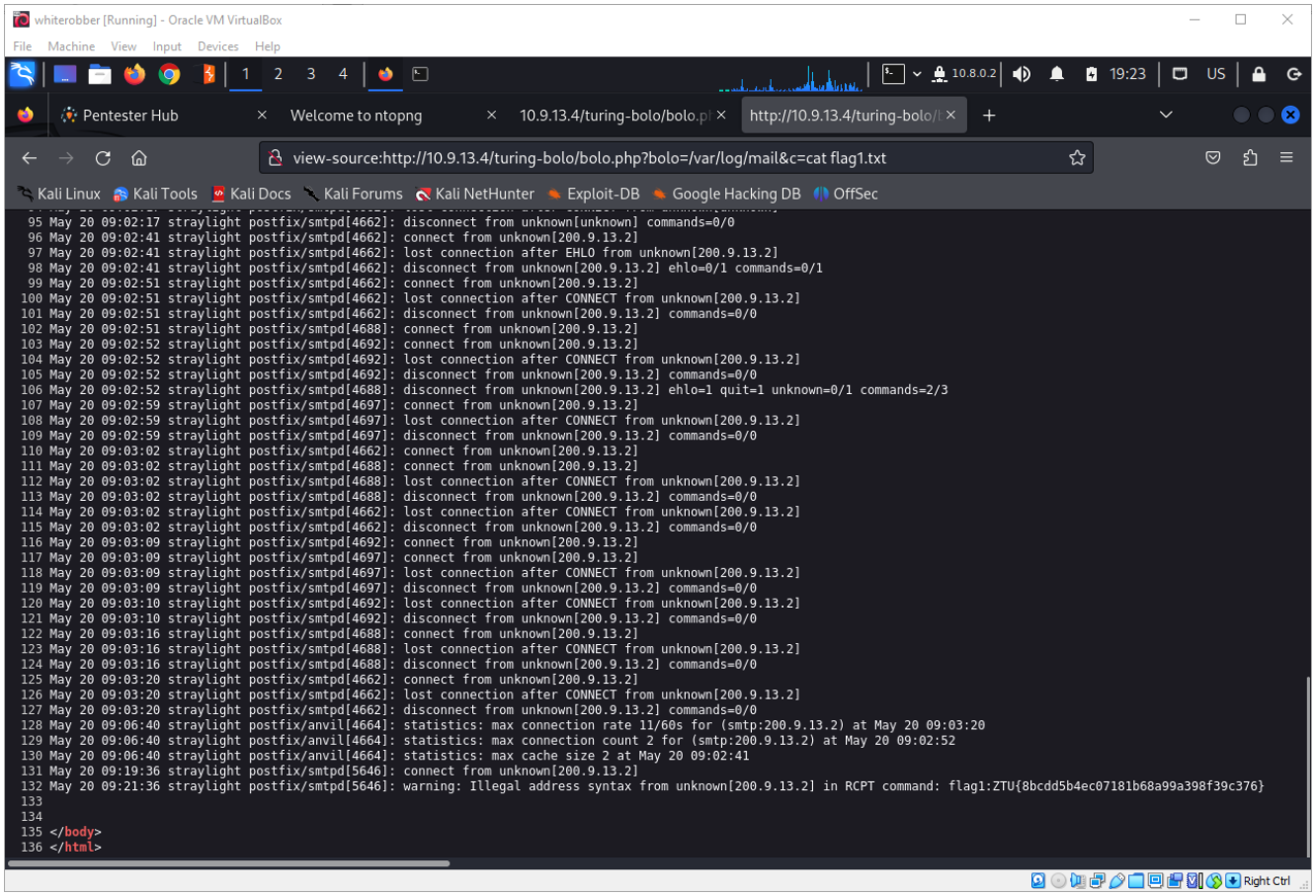


Рисунок 3.5.72 – Перший прапор успішно знайдено

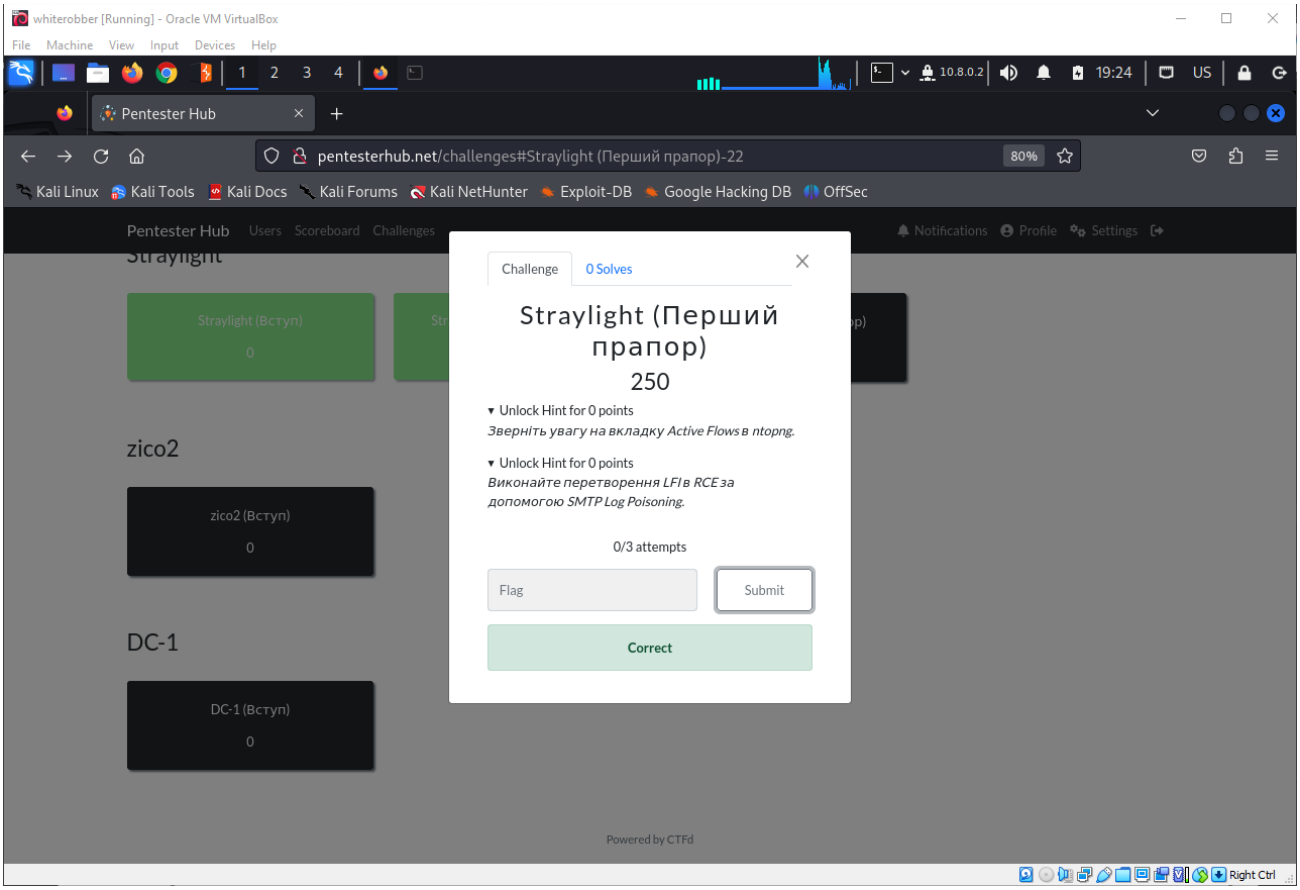


Рисунок 3.5.73 – Прапор успішно надіслано, завдання зараховано

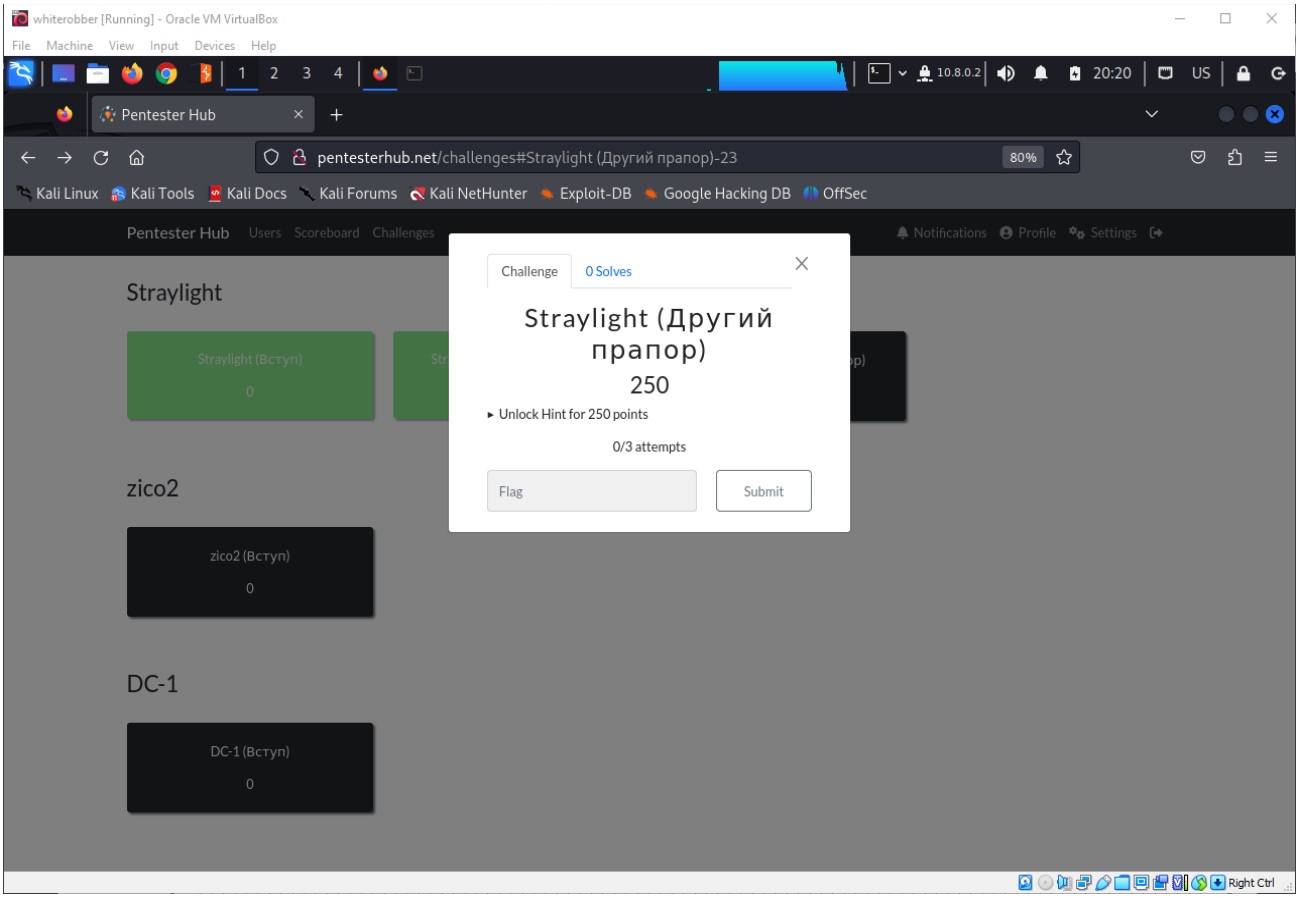


Рисунок 3.5.74 – Третє/фінальне завдання Straylight (Другий прапор)

В завдані зазначено заблоковану підказку, для перевірки функціоналу розблокуємо її.

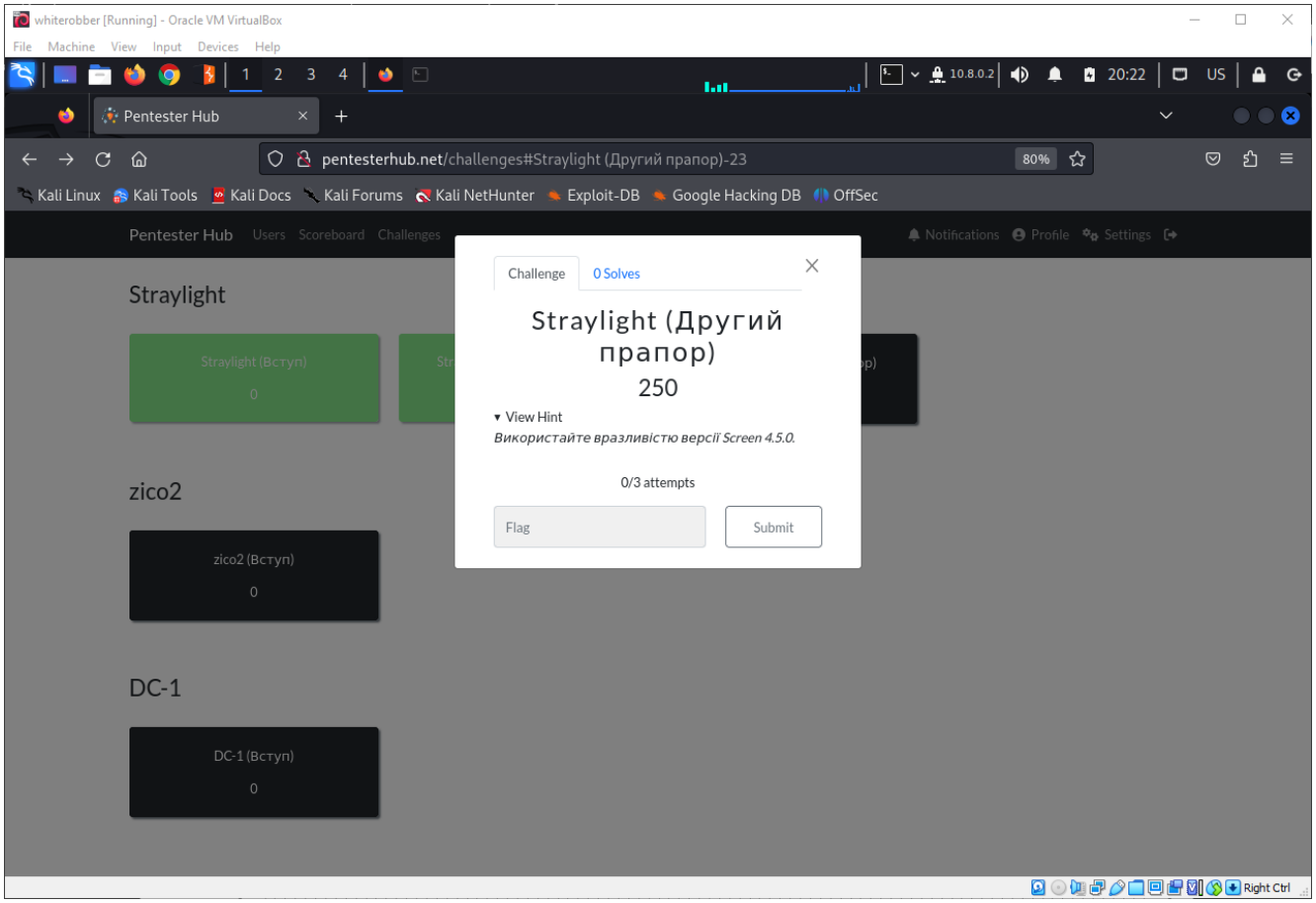


Рисунок 3.5.75 – Розблоковано підказку

Спробуємо додати зворотну оболонку.

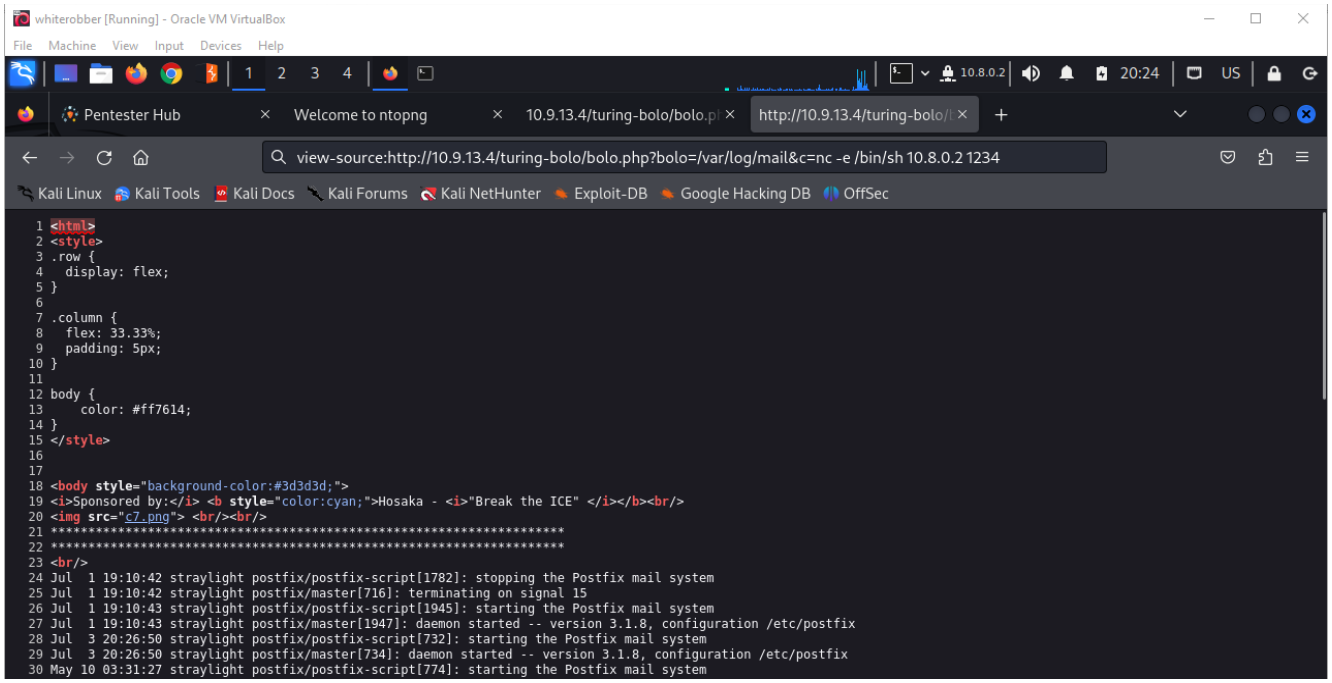


Рисунок 3.5.76 – Додання зворотної оболонки

Зворотну оболонку успішно додано, створимо нове вікно терміналу та почнемо прослуховувати встановлений порт 1234.

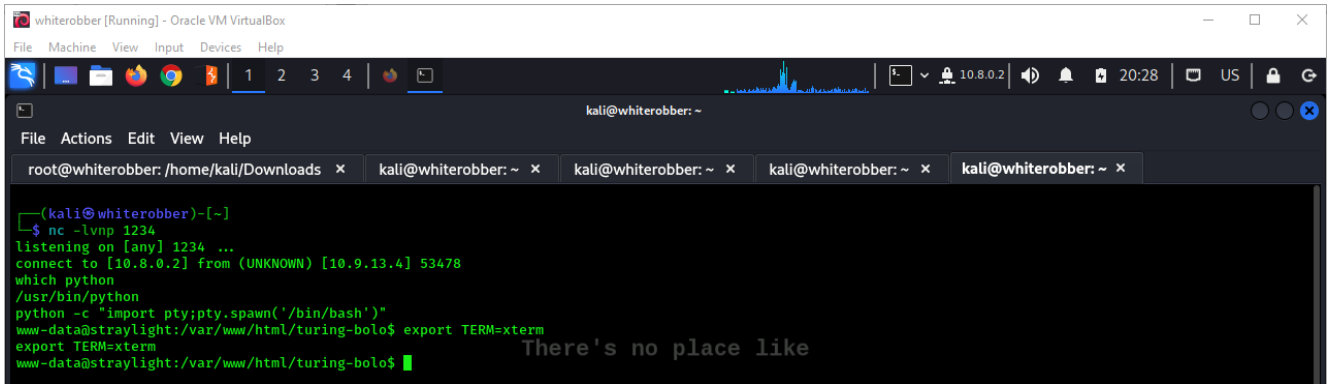


Рисунок 3.5.77 – Зворотна оболонка спрацювала , підключення успішне

Згідно вказаній інформації в розблокованій підказці почнемо пошук вразливості для версії Screen 4.5.0 за допомогою утиліти searchsploit.

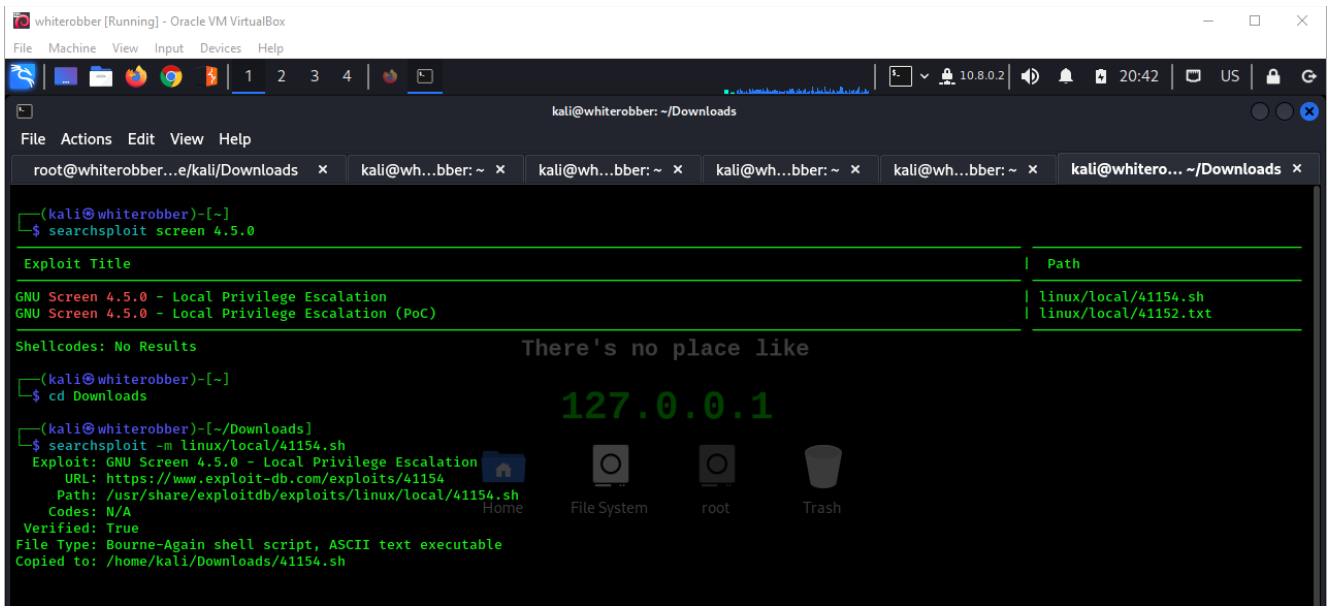


Рисунок 3.5.78 – Знайдено вразливість та завантажено скрипт 41154.sh

Завантажимо даний скрипт на машини Straylight за допомогою Python SimpleHTTPServer та каталогу tmp.

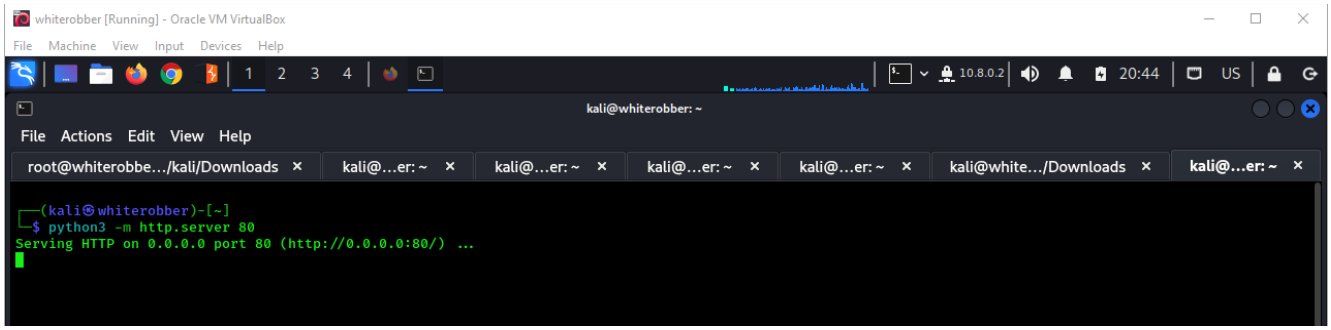


Рисунок 3.5.79 – Запускаємо Python SimpleHTTPServer на робочій станції

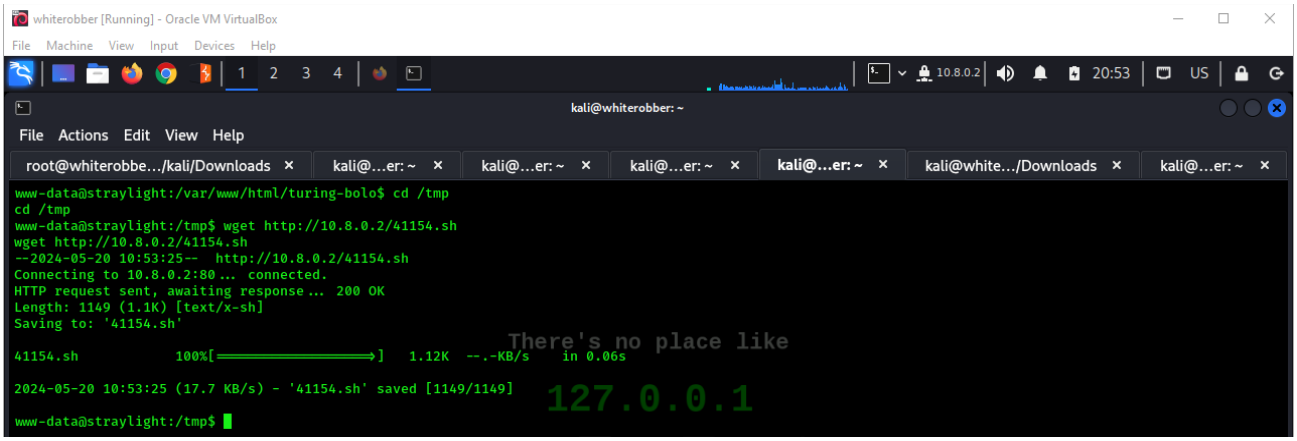


Рисунок 3.5.80 – Переходимо до каталогу tmp та використовуючи wget завантажуємо скрипт 41154.sh на машину Straylight

Встановлюємо права для виконання та запускаємо даний скрипт.

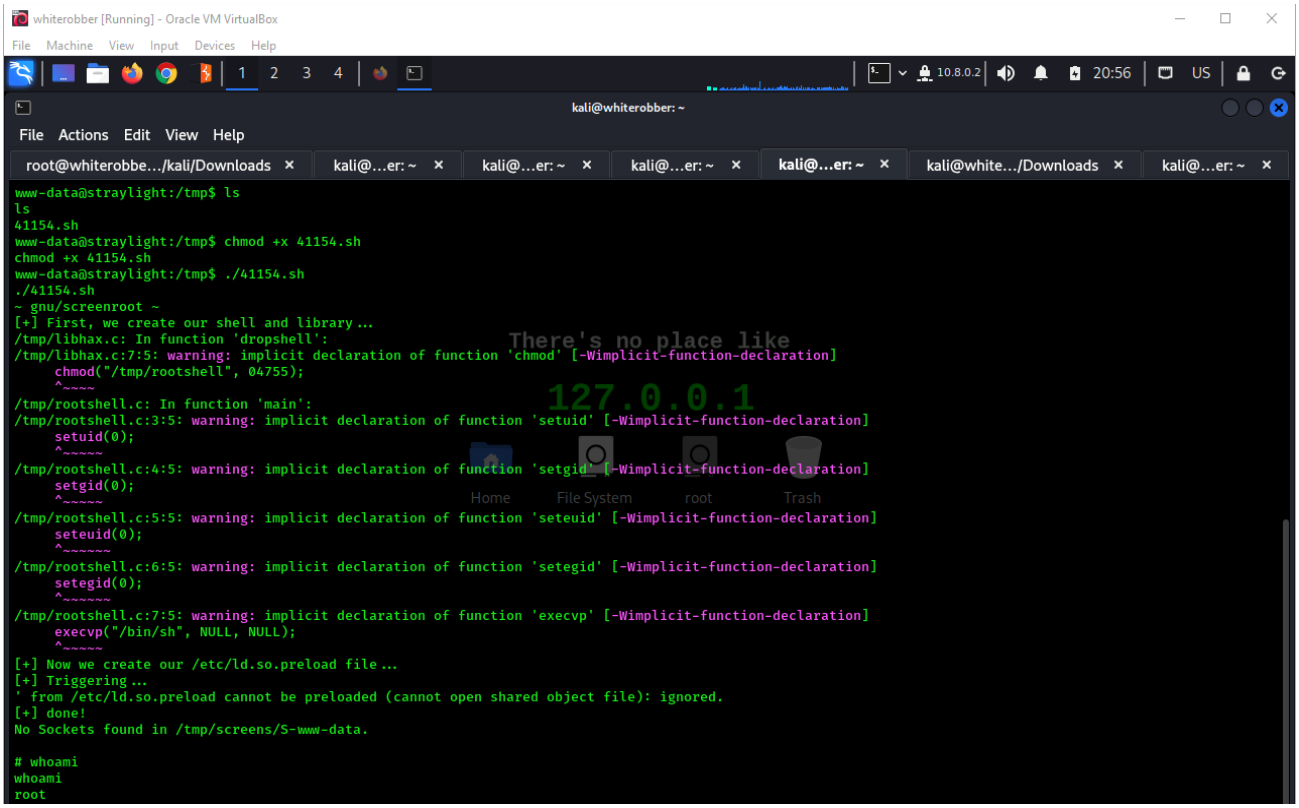


Рисунок 3.5.81 – Виконання скрипта

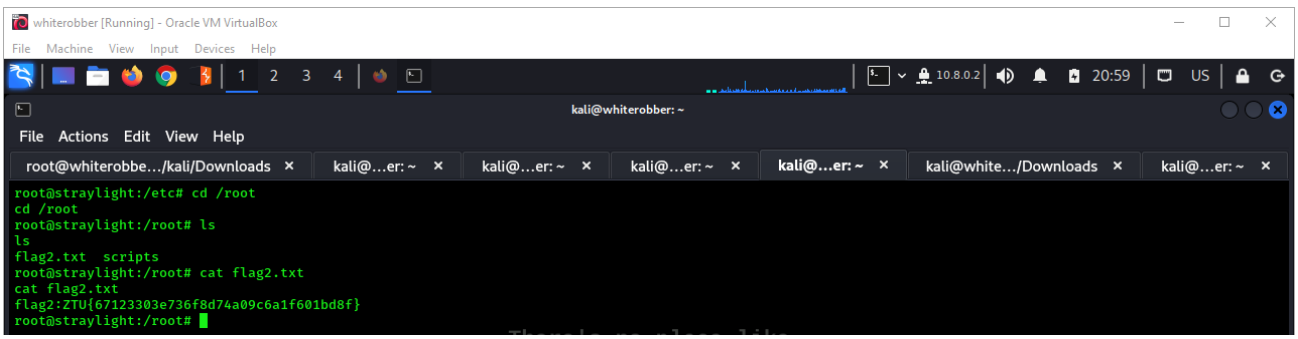


Рисунок 3.5.82 – Другий прапор успішно знайдено

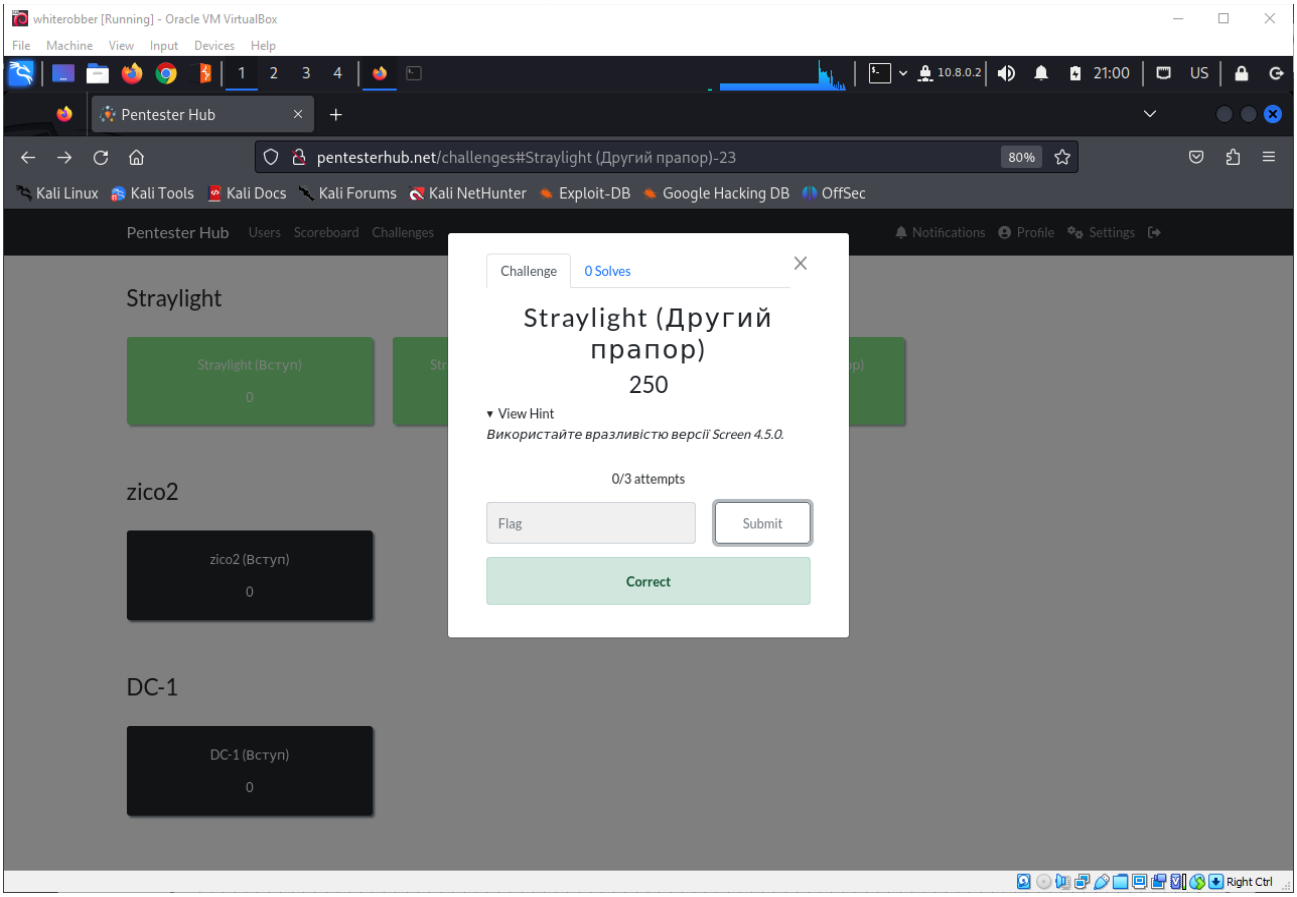


Рисунок 3.5.83 – Прапор успішно надіслано, завдання зараховано

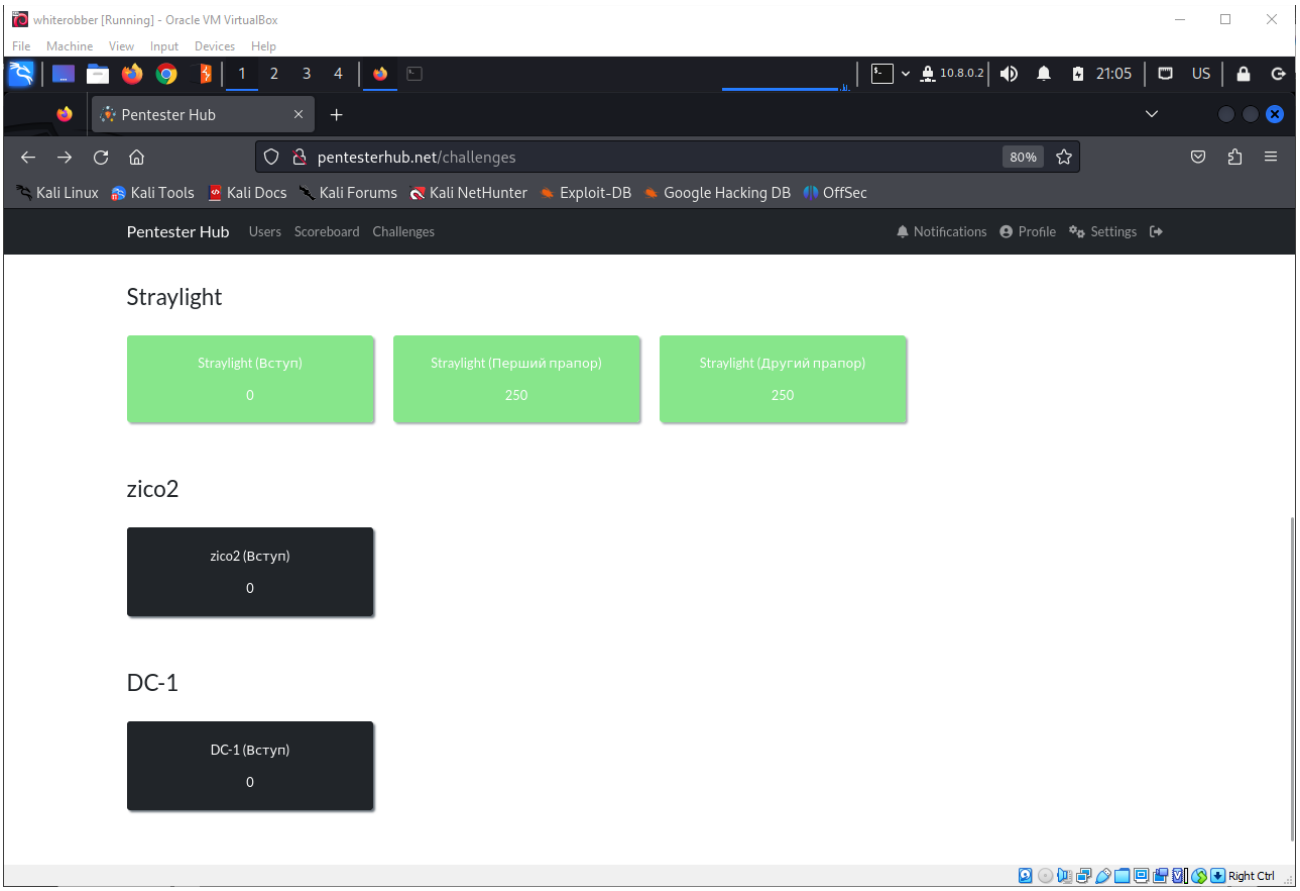


Рисунок 3.5.84 – Серія завдань Straylight успішно протестована (вирішена)

Розпочнемо вирішення серії завдань для машини zico2.

Зазначимо, процес ознайомлення з описом завдання, виконання підключення до OpenVPN сервера, успішне відкриття веб-сайту машини zico2 та перегляд безкоштовної підказки для того, щоб надіслати прапор для виконання завдання та відкрити наступне завдання в серії аналогічний до SolidState.

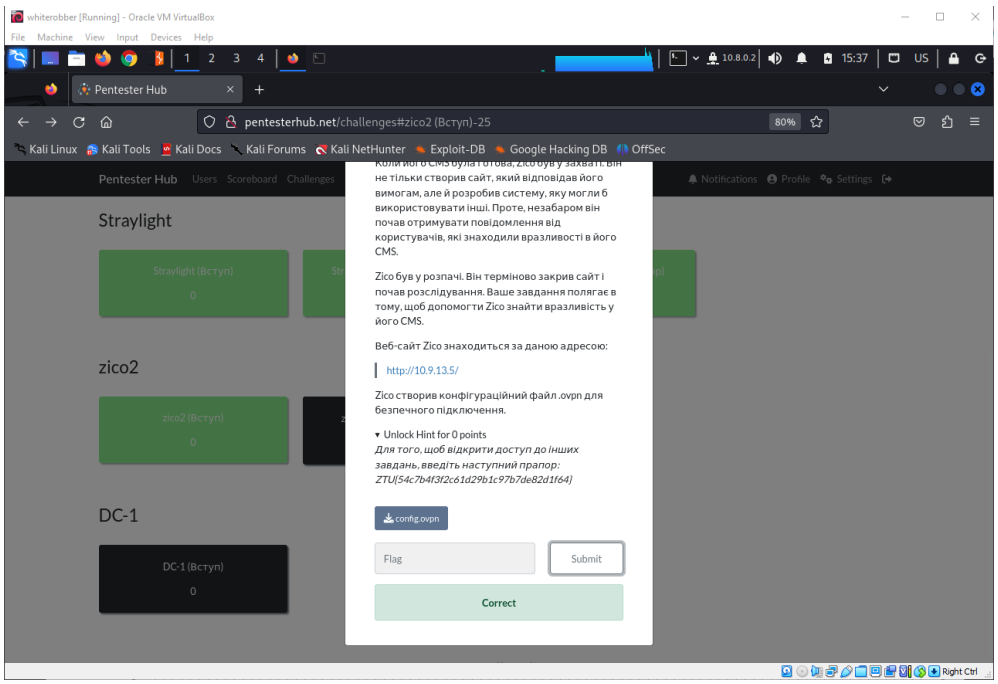


Рисунок 3.5.85 – Перше завдання zico2 (Вступ),
прапор успішно надіслано, завдання зараховано

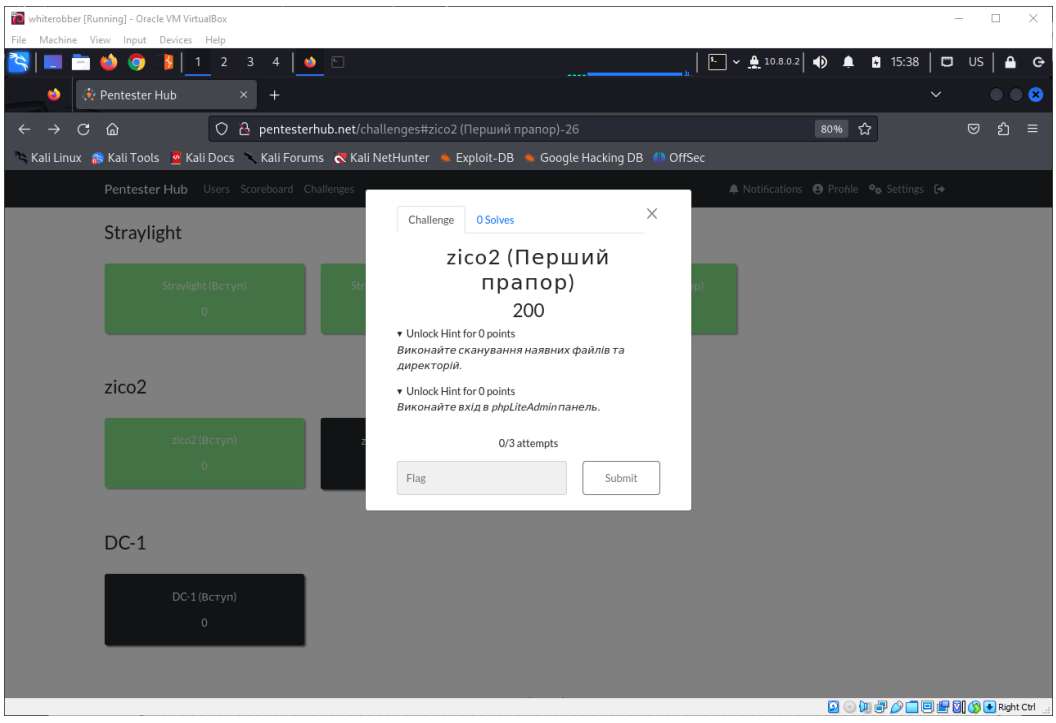


Рисунок 3.5.86 – Друге завдання zico2 (Перший прапор)

Звертаючи увагу на першу безкоштовну підказку можна зробити висновок, що потрібно виконати сканування сканування наявних файлів та директорій.

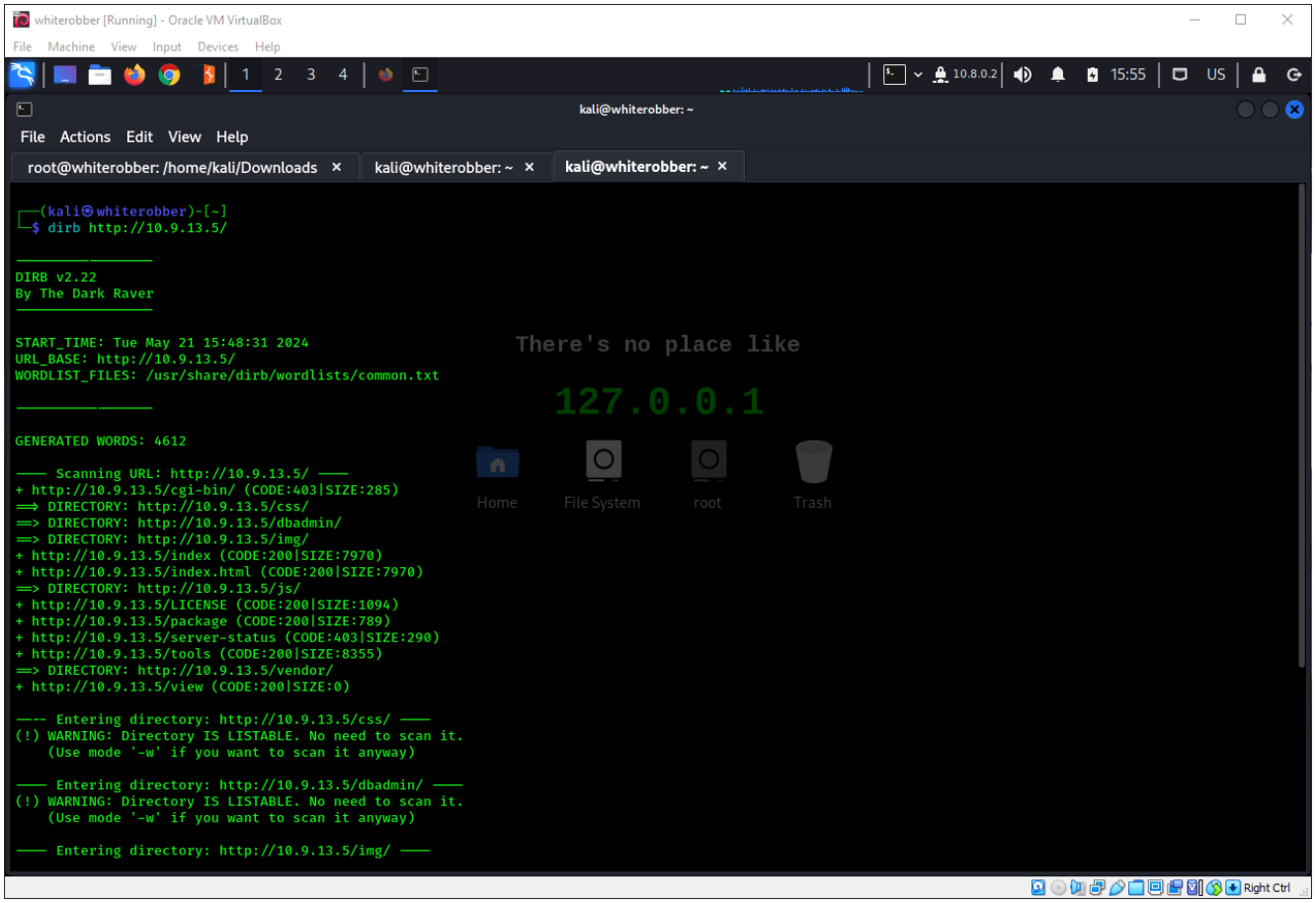


Рисунок 3.5.87 – Результат сканування наявних файлів та директорій за допомогою інструмента dirb

В результаті сканування було виявлено декілька директорій, але найбільший інтерес викликає директорія dbadmin, спробуємо її переглянути.

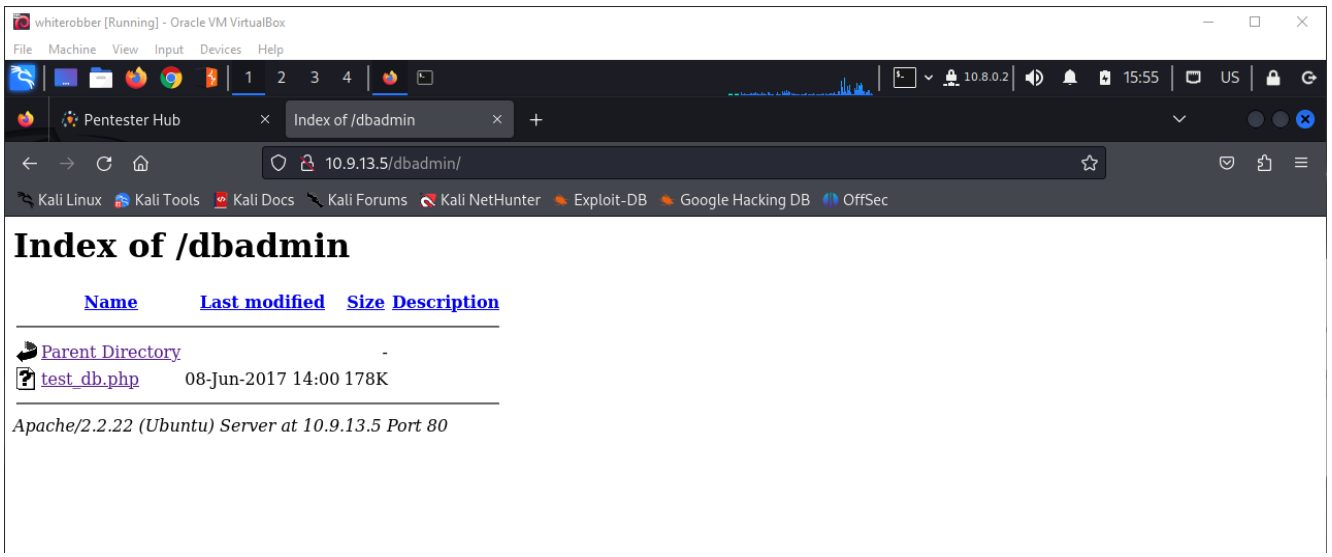


Рисунок 3.5.88 – Перегляд директорії dbadmin

Всередині директорії dbadmin знаходиться файл test_db.php, спробуємо його відкрити.

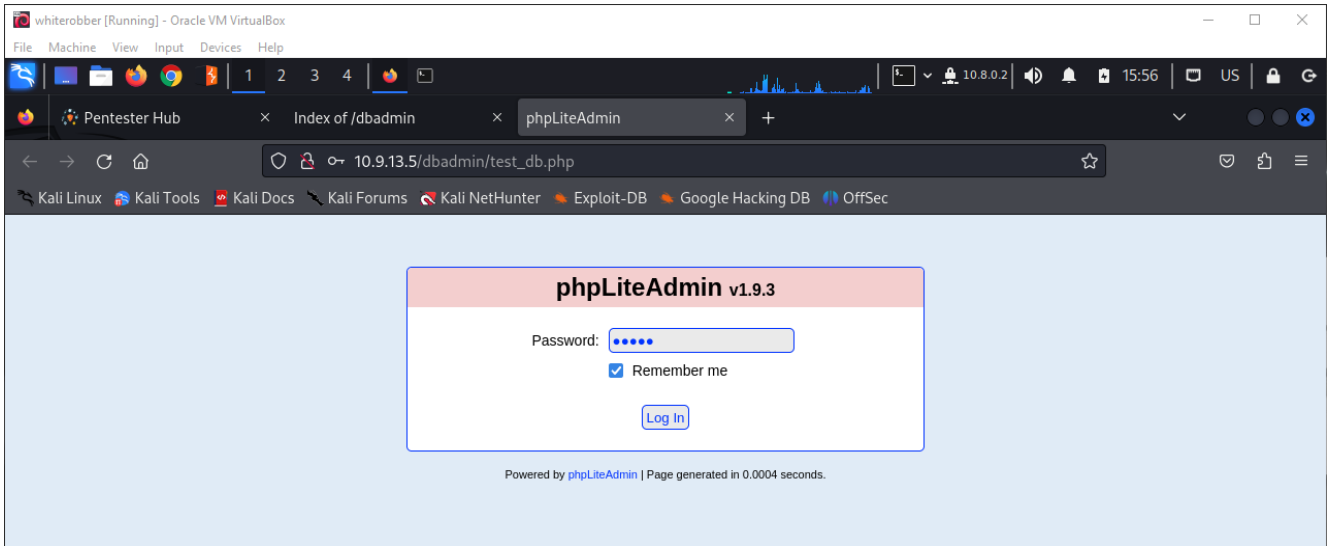


Рисунок 3.5.89 – phpLiteAdmin для виконання входу

було використано пароль за замовчуванням admin

Згідно вказаній інформації в другій безкоштовній підказці, було здійснено вхід в phpLiteAdmin панель.

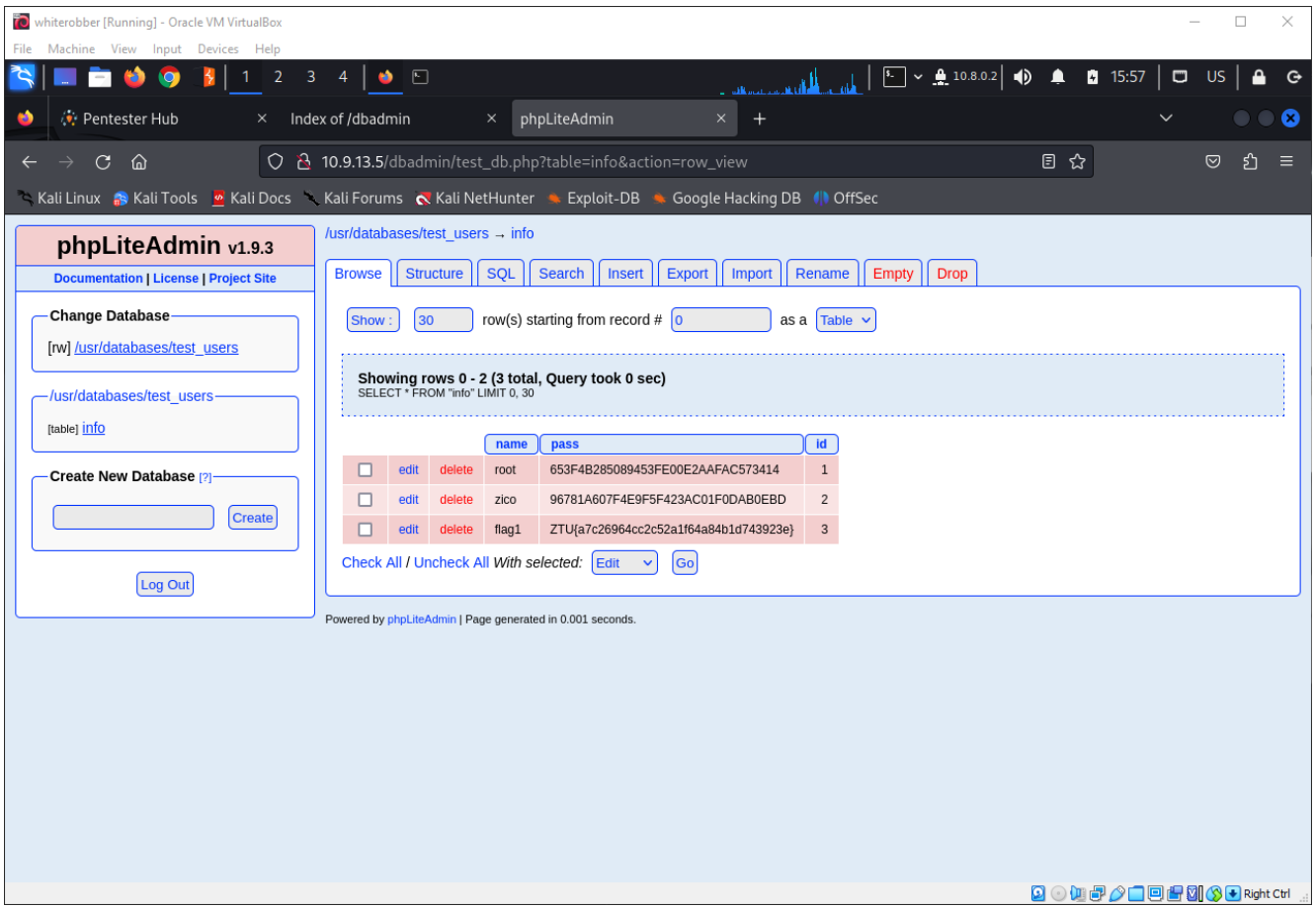


Рисунок 3.5.90 – Перегляд таблиці info, перший прапор успішно знайдено

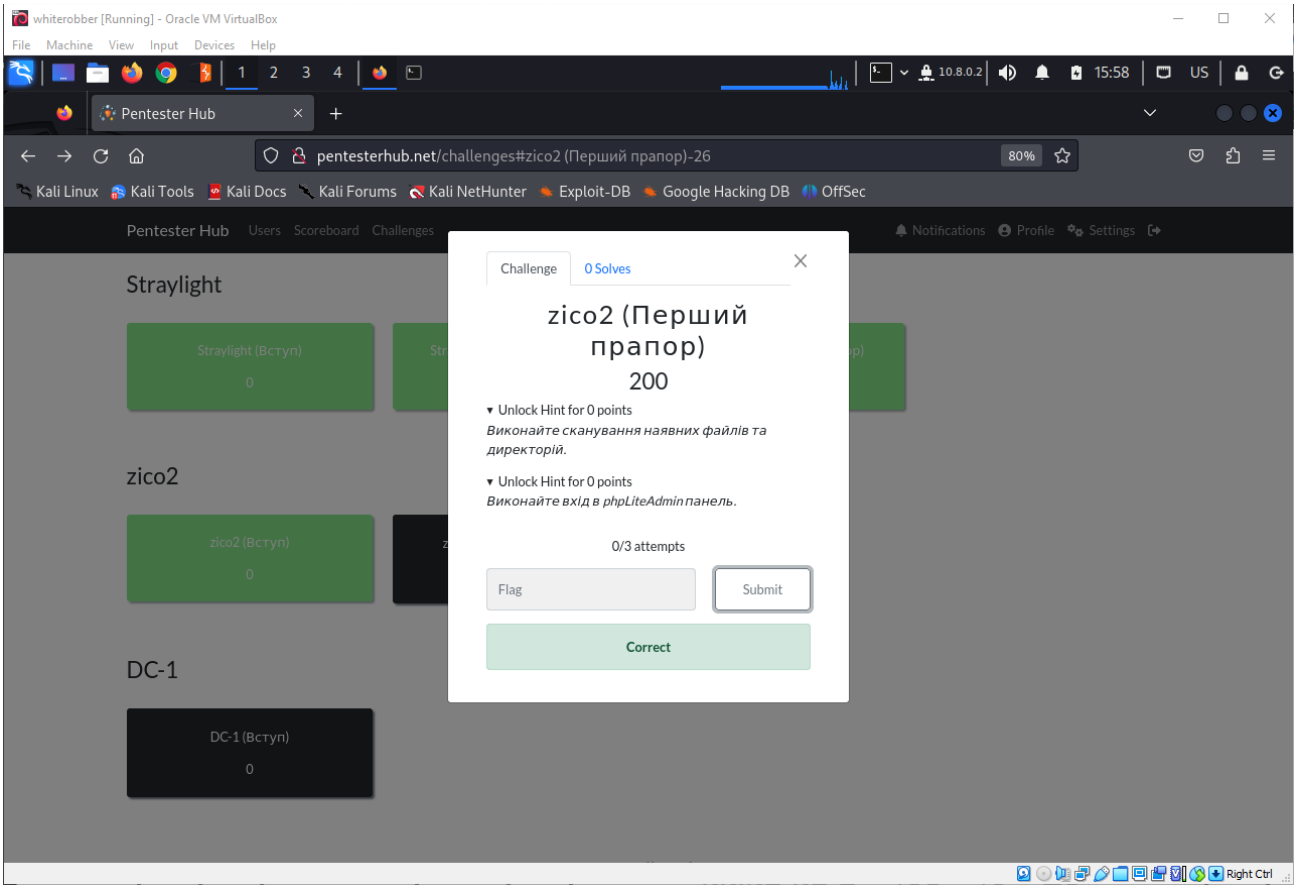


Рисунок 3.5.91 – Прапор успішно надіслано, завдання зараховано

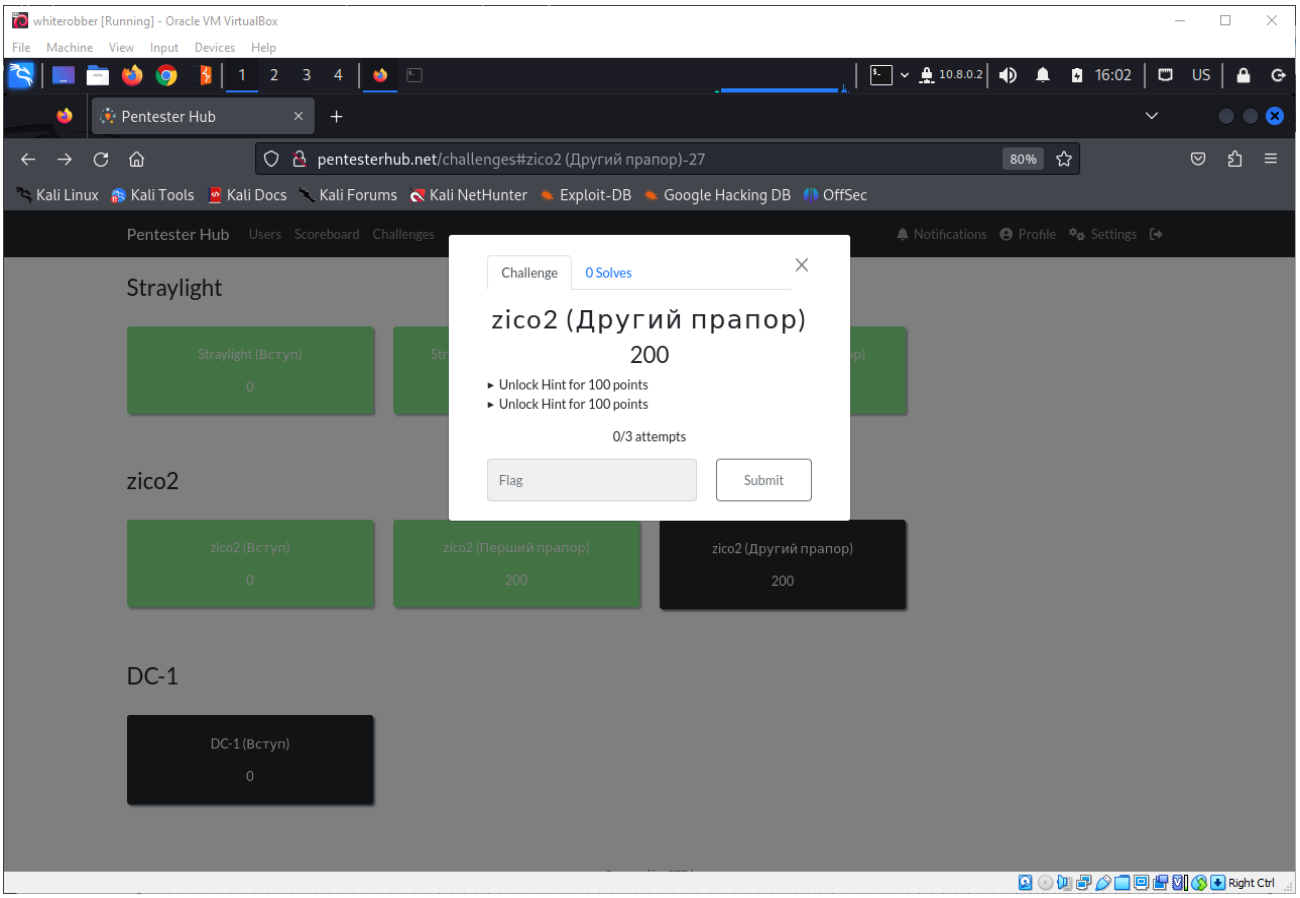


Рисунок 3.5.92 – Третє завдання zico2 (Другий прапор)

Два md5 хеши, які також були зазначені в таблиці info успішно зламані.

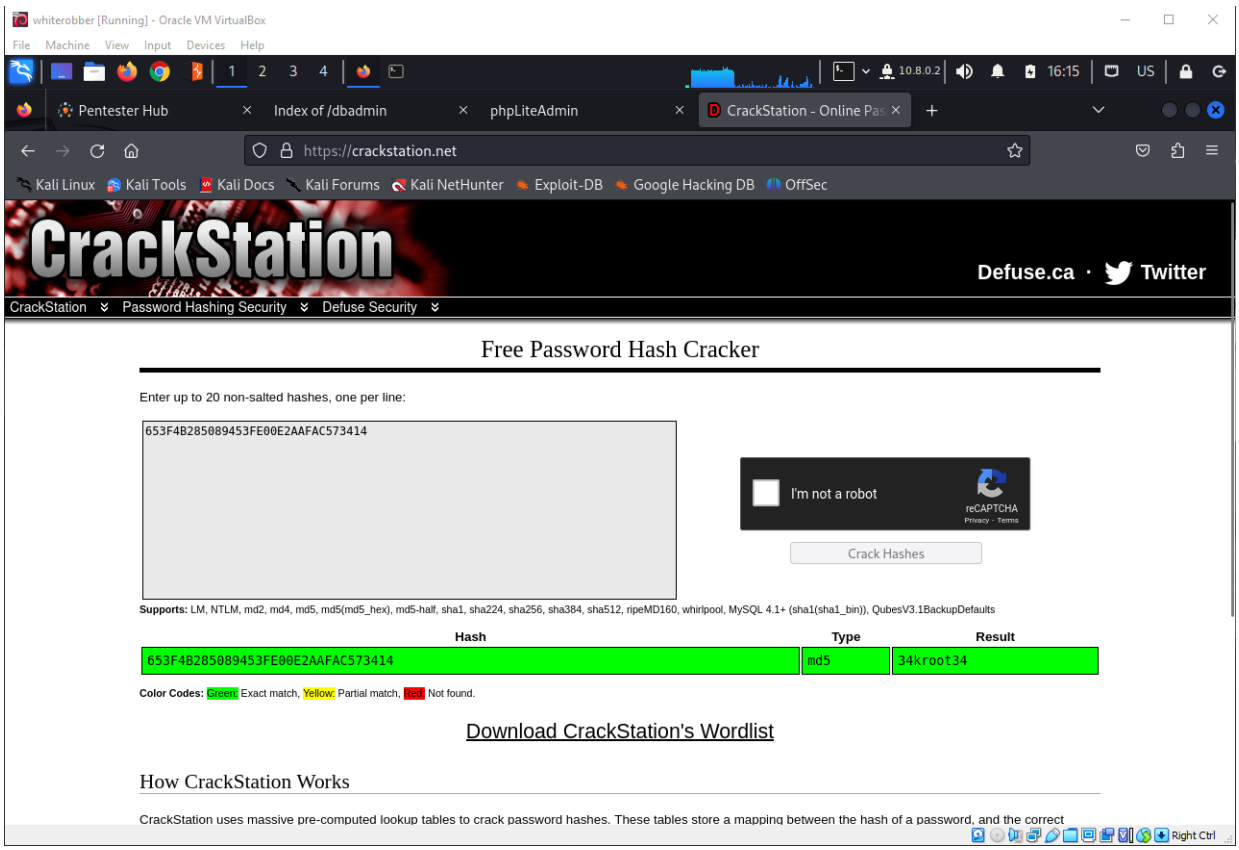


Рисунок 3.5.93 – Успішно зламаний хеш 653F4B285089453FE00E2AAFAC573414

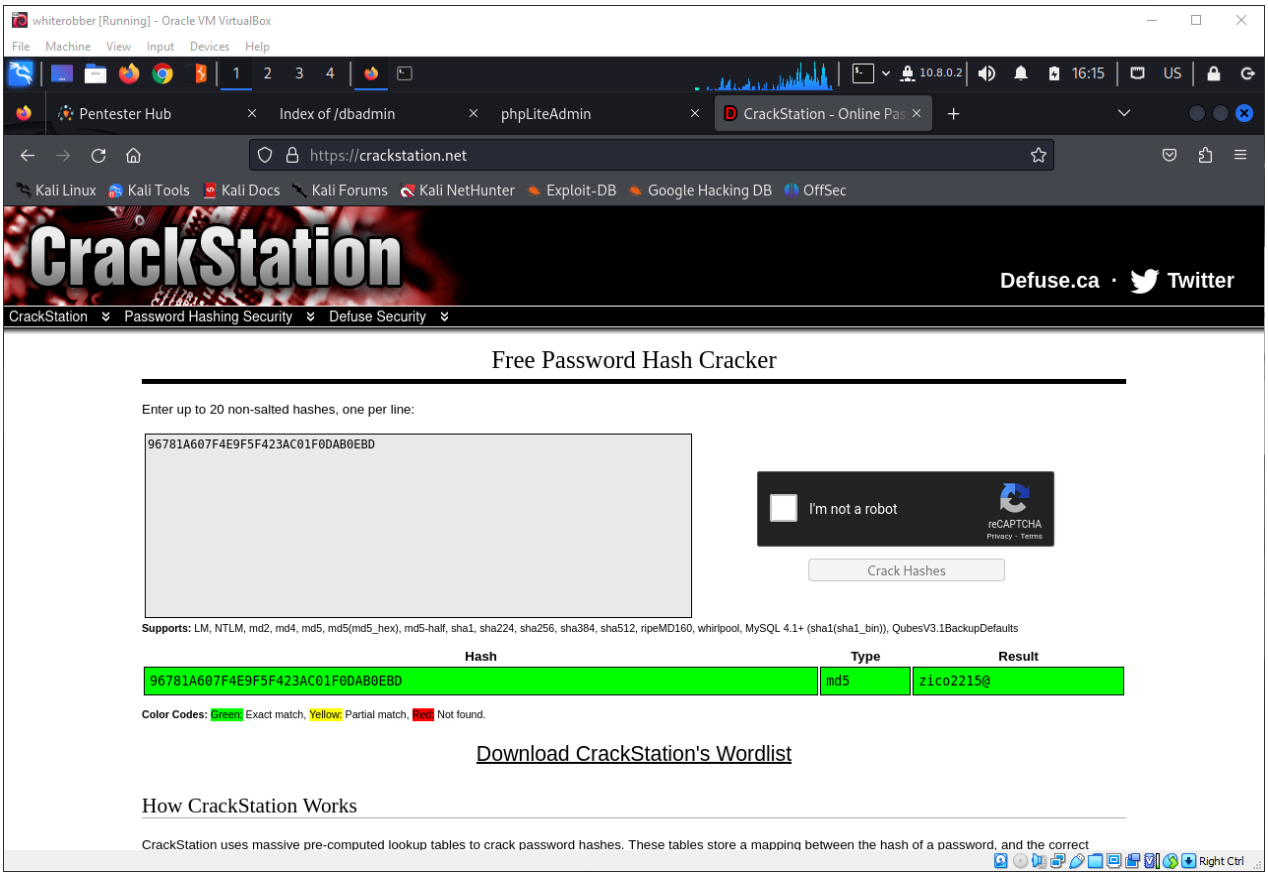


Рисунок 3.5.94 – Успішно зламаний хеш 96781A607F4E9F5F423AC01F0DAB0EBD

Нажаль використовуючи дані облікові дані не вдалось виконати підключення по SSH, тому наступний кроком є пошук вразливості для версії phpLiteAdmin 1.9.3.

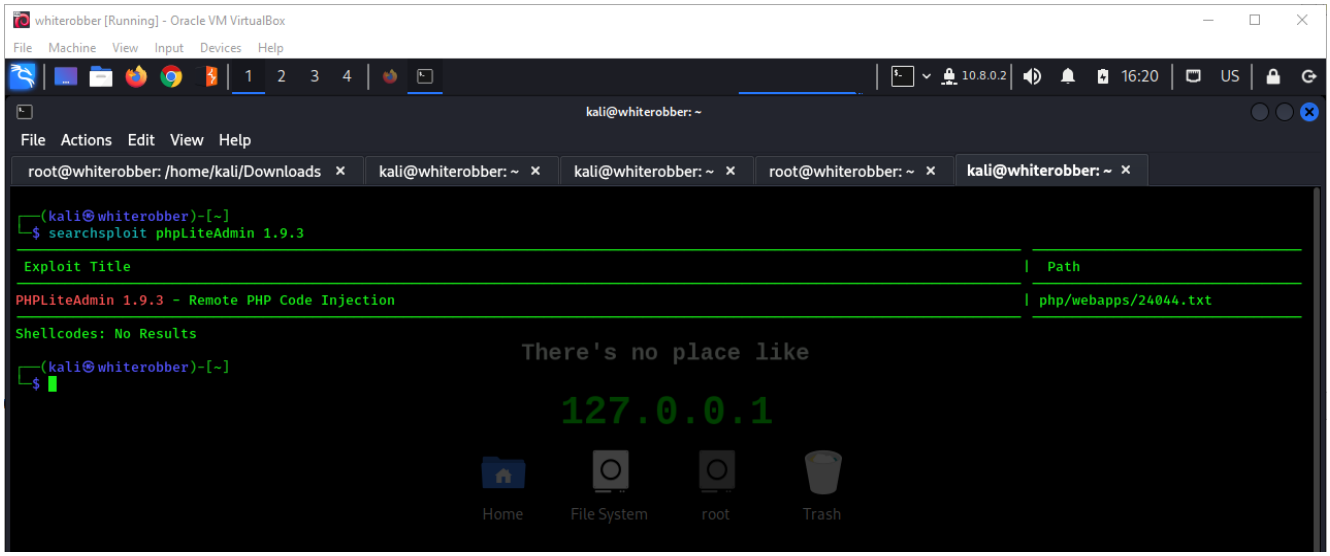


Рисунок 3.5.95 – Знайдено вразливість

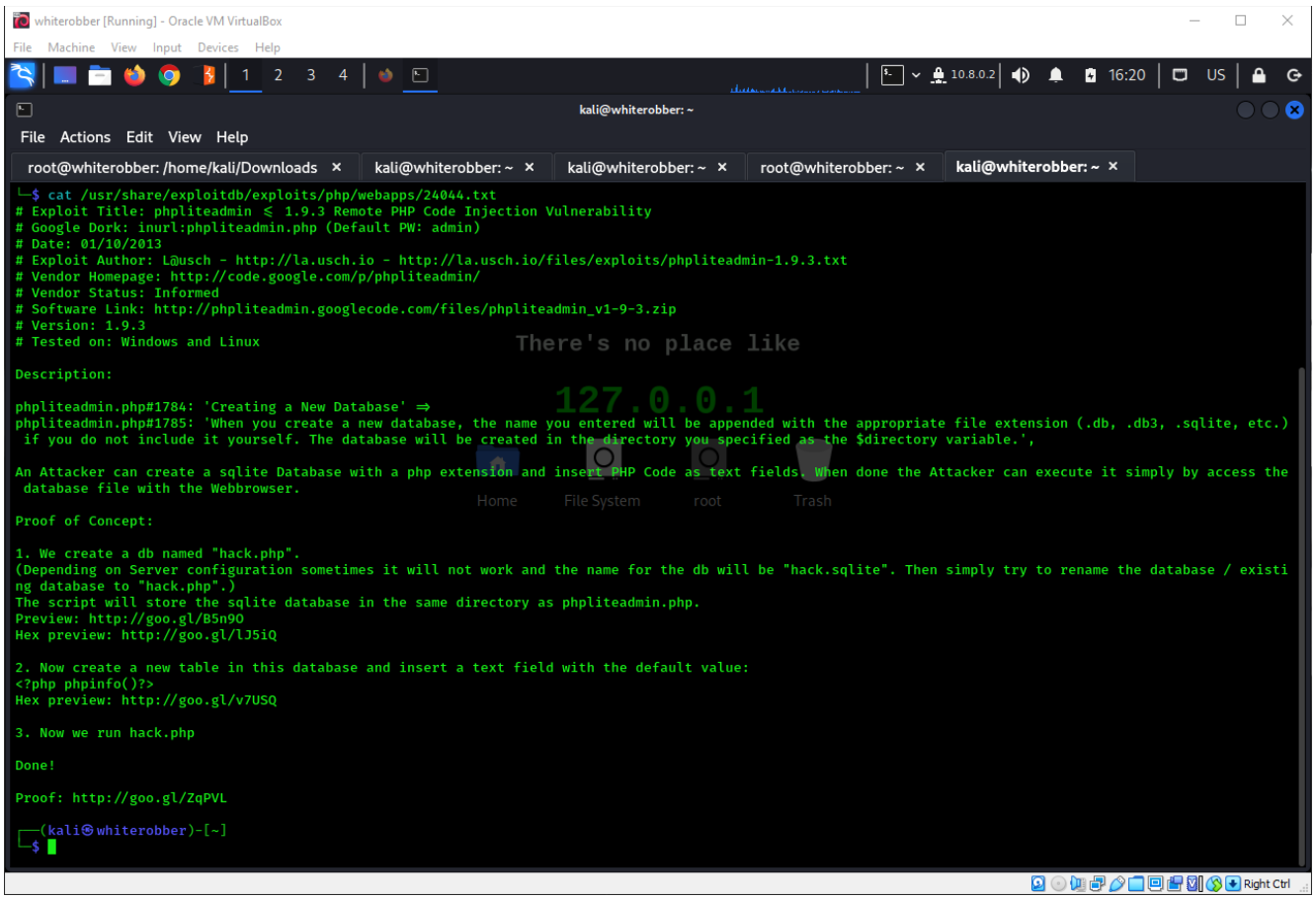


Рисунок 3.5.96 – Перегляд інформації про вразливість

Проаналізувавши інформацію про вразливість стало зрозуміло, що потрібно створити нову базу даних hack.php всередині цієї бази даних створити таблицю exploit з текстовим типом даних в якій потрібно внести RCE payload.

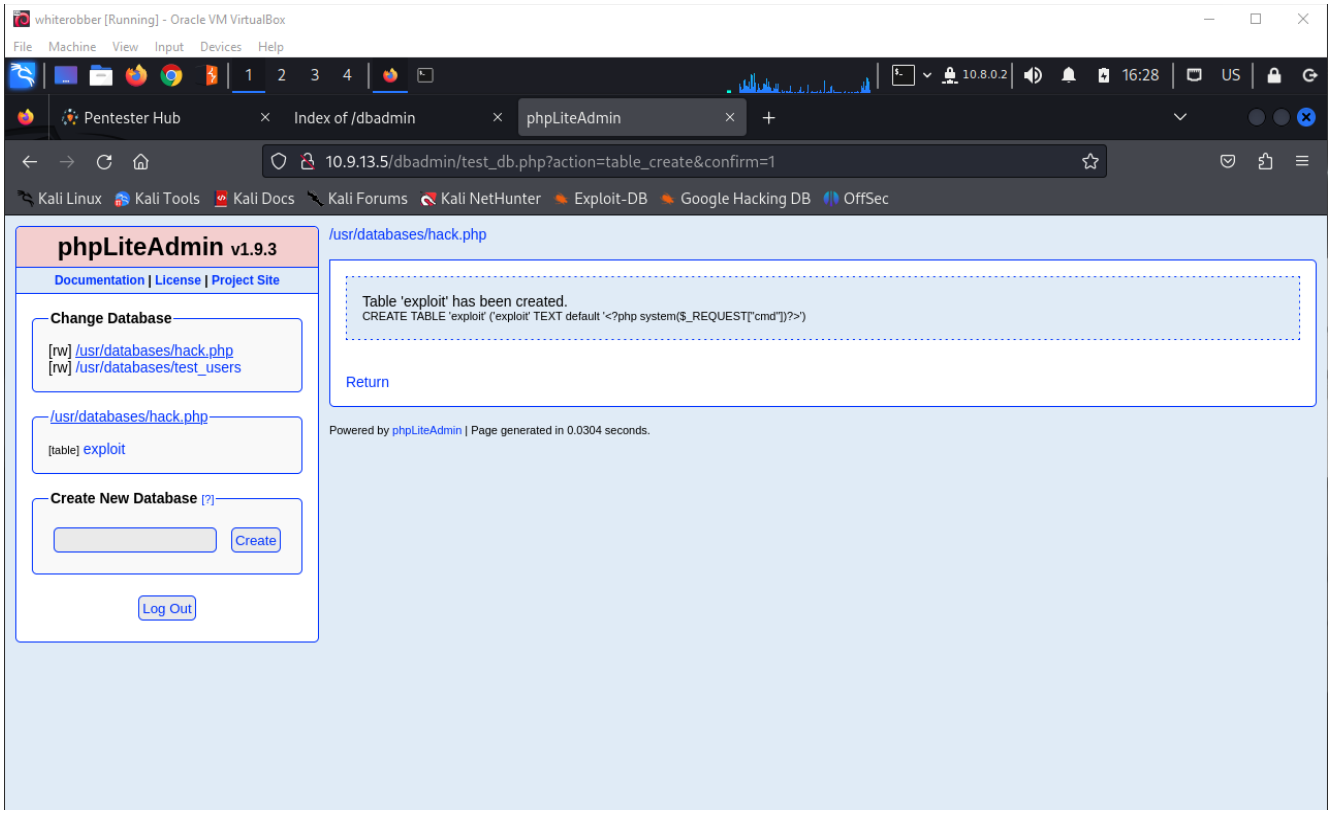


Рисунок 3.5.97 – Створення бази даних hack.php, таблиці exploit з текстовим типом даних в якій внесено RCE payload

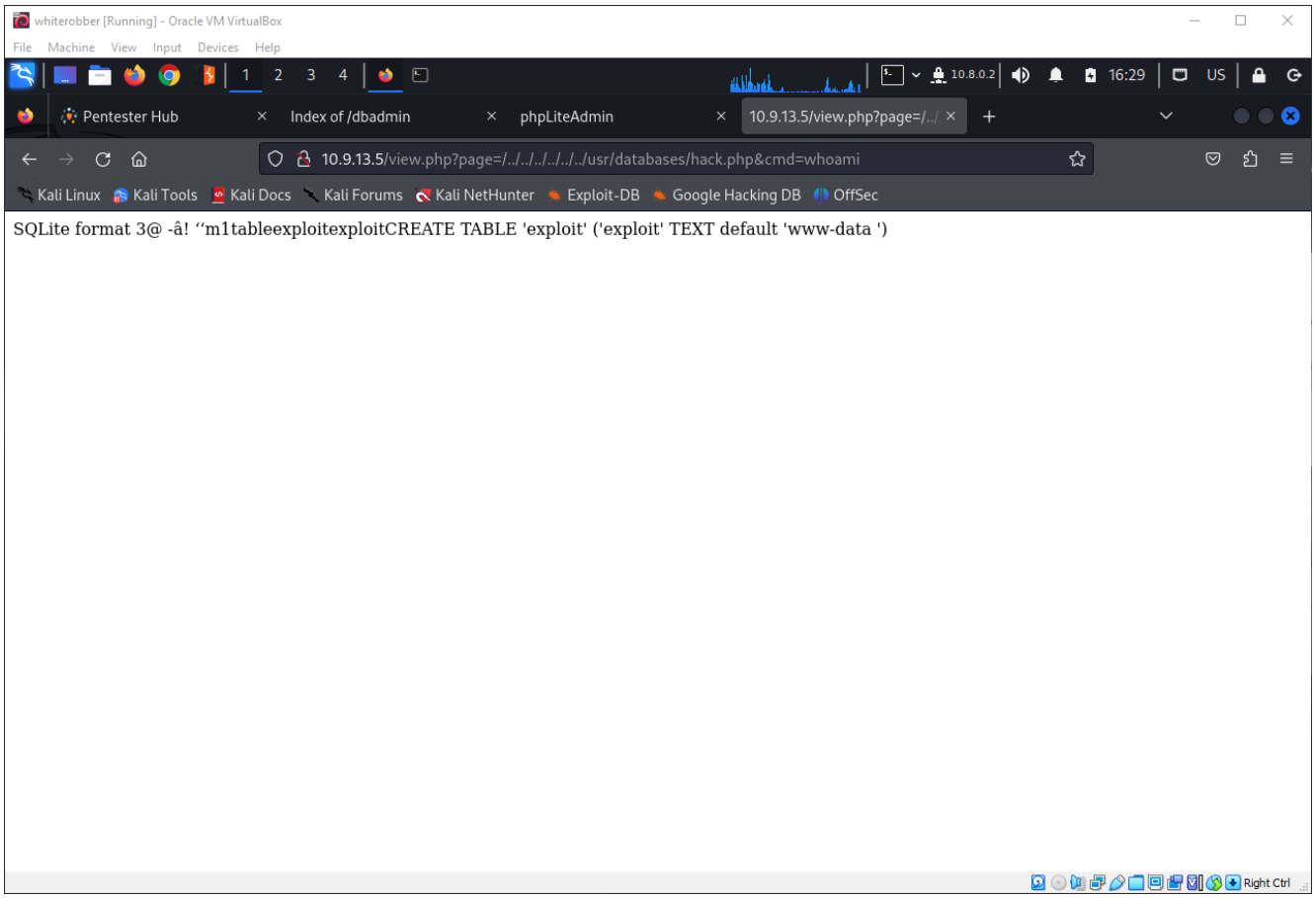


Рисунок 3.5.98 – Перевірка працездатності RCE

Наступним кроком є створення файлу зворотної оболонки shell.php

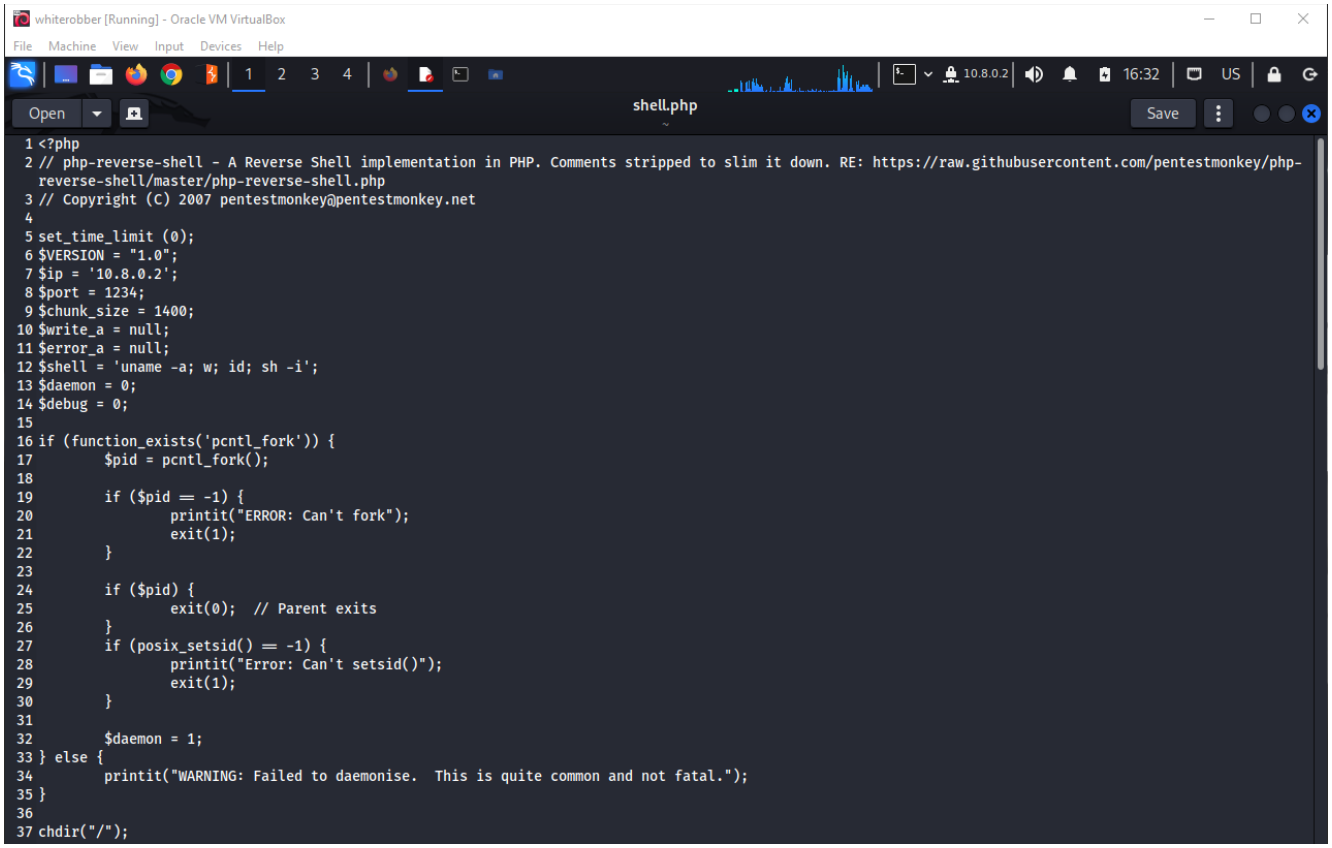


Рисунок 3.5.99 – Файл зворотної оболонки shell.php

Завантажимо файл зворотної оболонки на машини zico2 за допомогою Python SimpleHTTPServer та каталогу tmp.

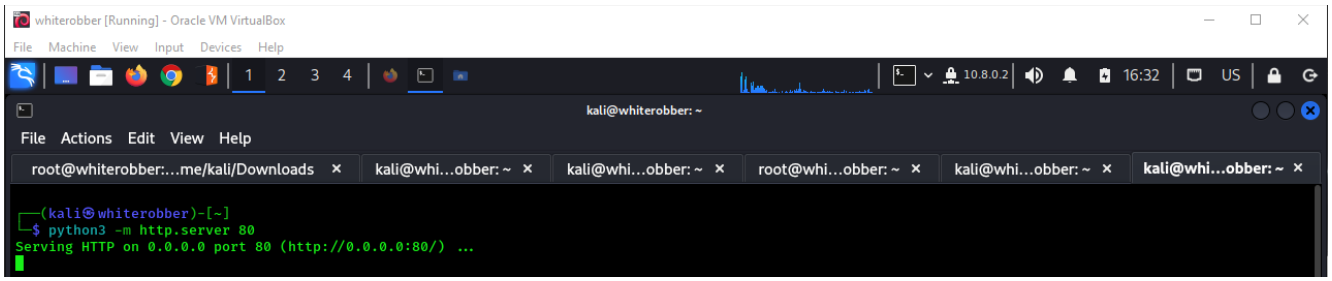


Рисунок 3.5.100 – Запускаємо Python SimpleHTTPServer на робочій станції

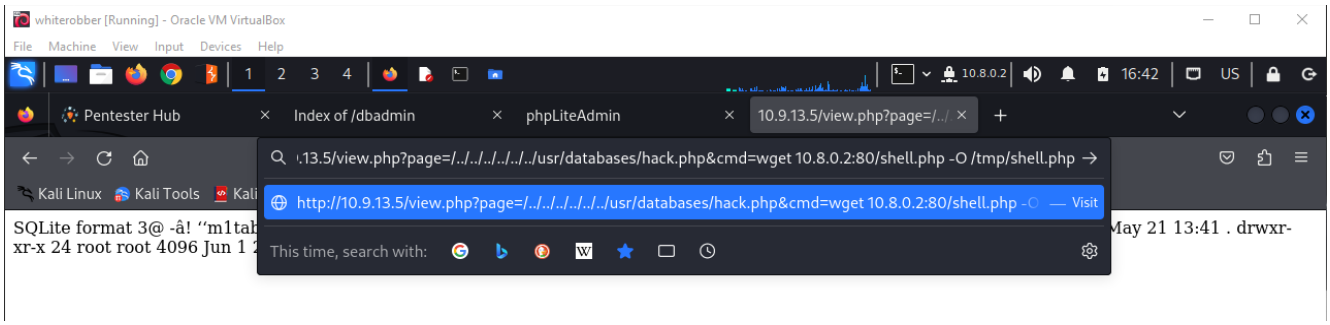


Рисунок 3.5.101 – Використовуючи wget завантажуємо файл зворотної оболонки shell.php на машину zico2 в каталог tmp

Зворотну оболонку успішно завантажено на машину zico2, створимо нове вікно терміналу та почнемо прослуховувати встановлений порт 1234, для активації зворотної оболонки виконаємо наступне у пошуковому рядку браузера:

– php /tmp/shell.php;

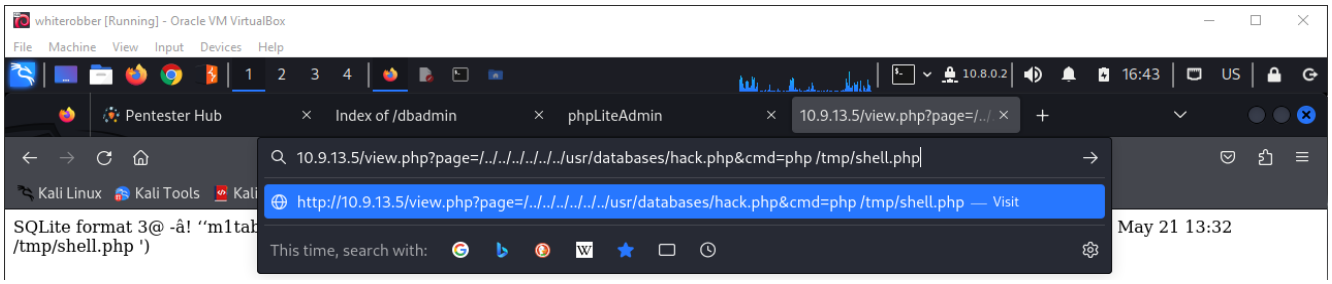


Рисунок 3.5.102 – Активація зворотної оболонки

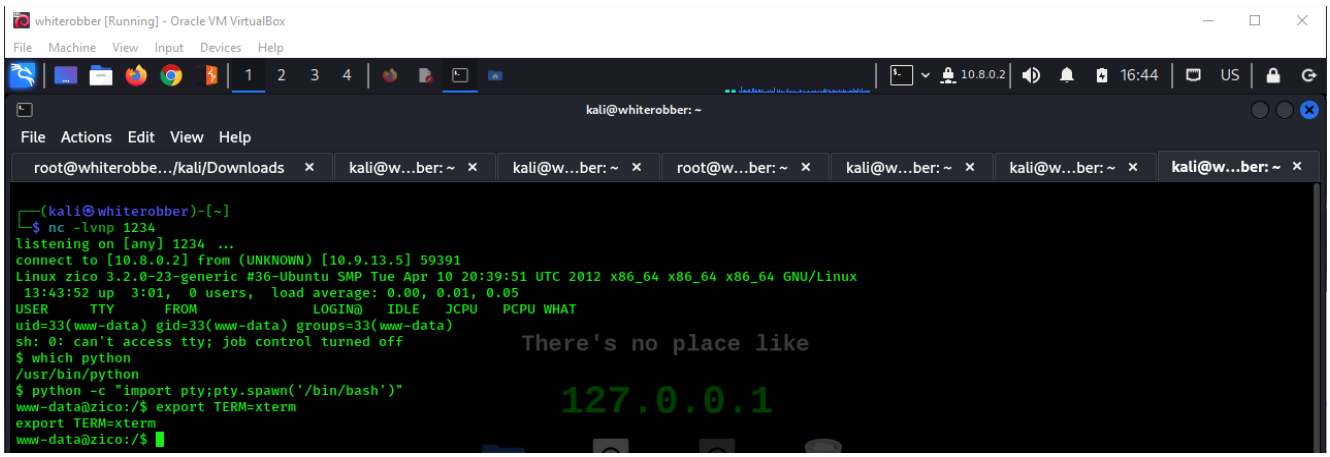


Рисунок 3.5.103 – Зворотна оболонка спрацювала, підключення успішне

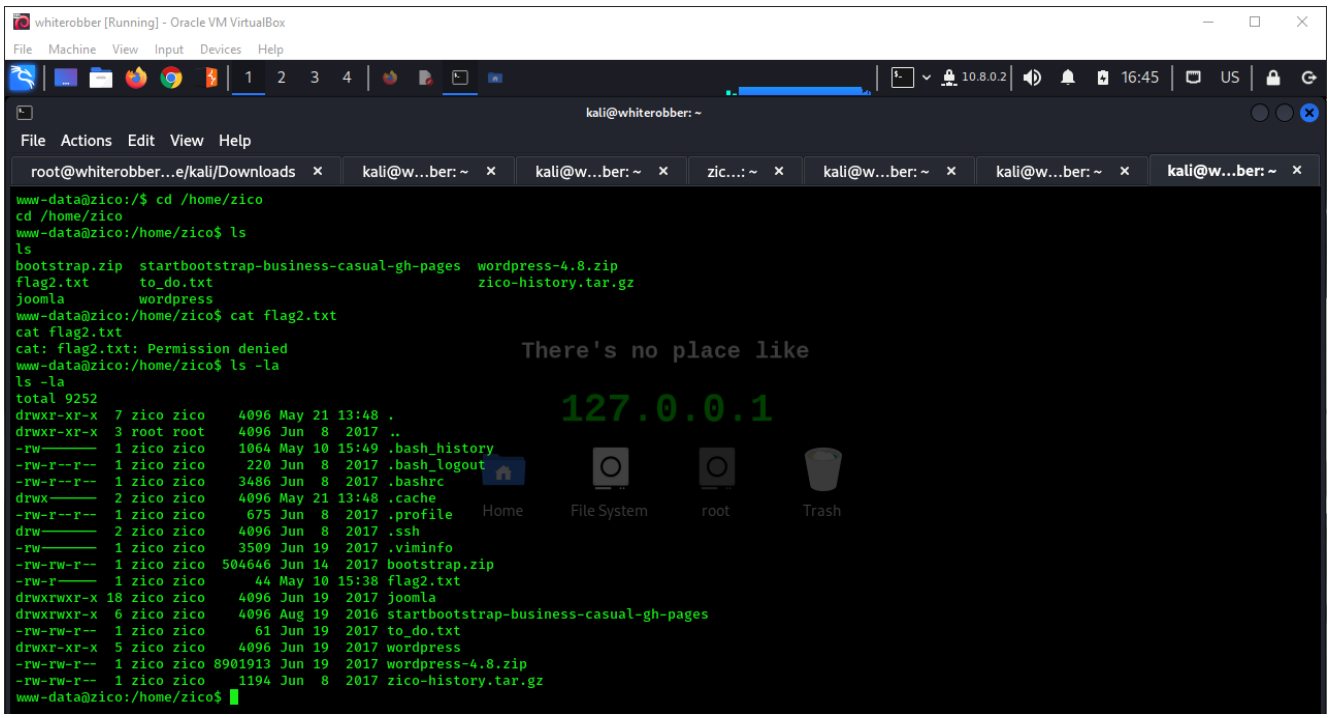


Рисунок 3.5.104 – Неможливо переглянути файл flag2.txt, відмовлено у доступі

Повідомлення відмовлено у доступі отримано через те, що доступ до файлу має користувач zico, на даний момент ми є користувачем www-data, серед переліку файлів та директорій наявна директорія wordpress, спробуємо переглянути вміст файлу wp-config.php та знайти облікові дані користувача zico.

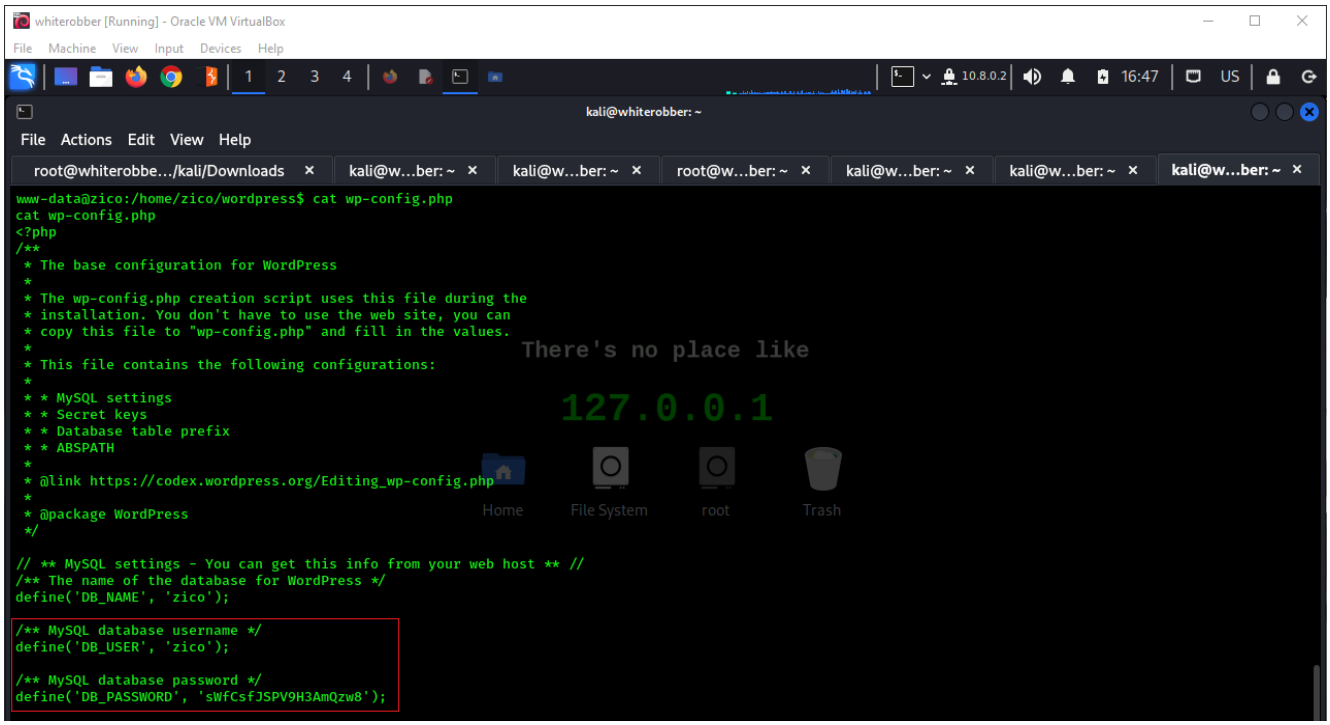


Рисунок 3.5.105 – Знайдено облікові дані MySQL користувача zico

Незважаючи на те, що були знайдені облікові дані MySQL, спробуємо виконати підключення по SSH використовуючи ці облікові дані.

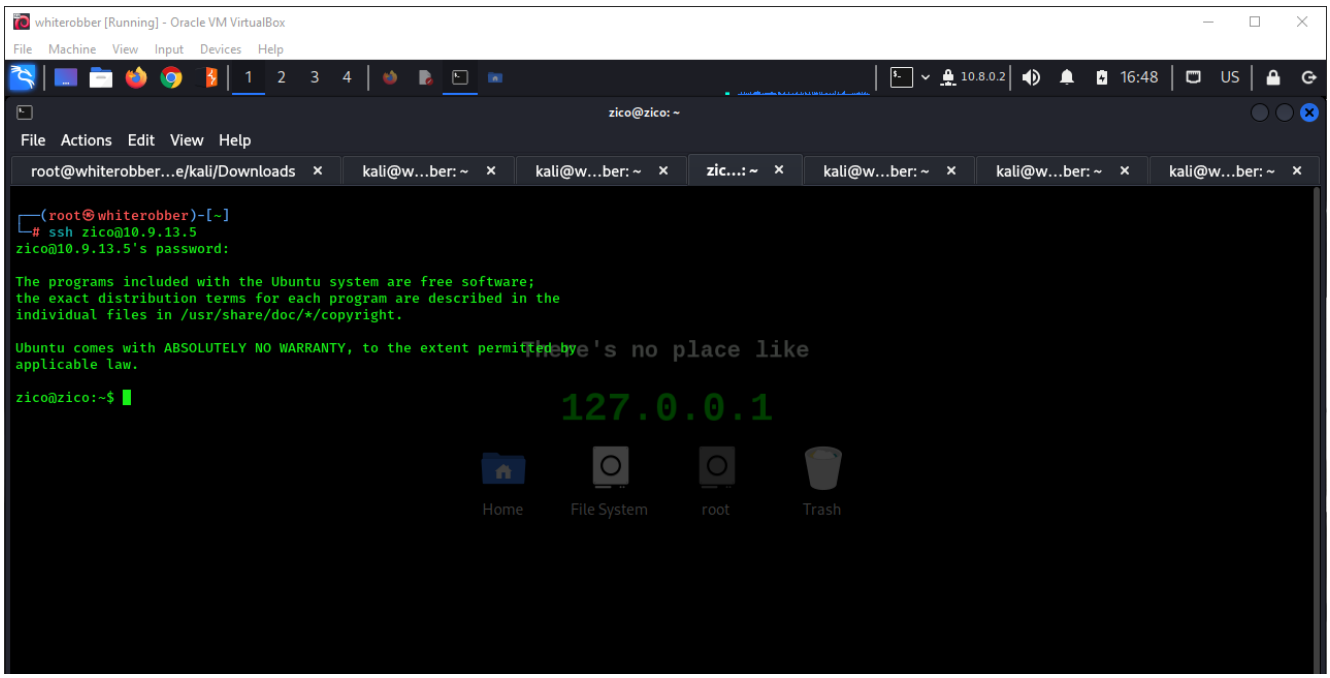


Рисунок 3.5.106 – Виконуємо підключення по SSH, підключення успішне

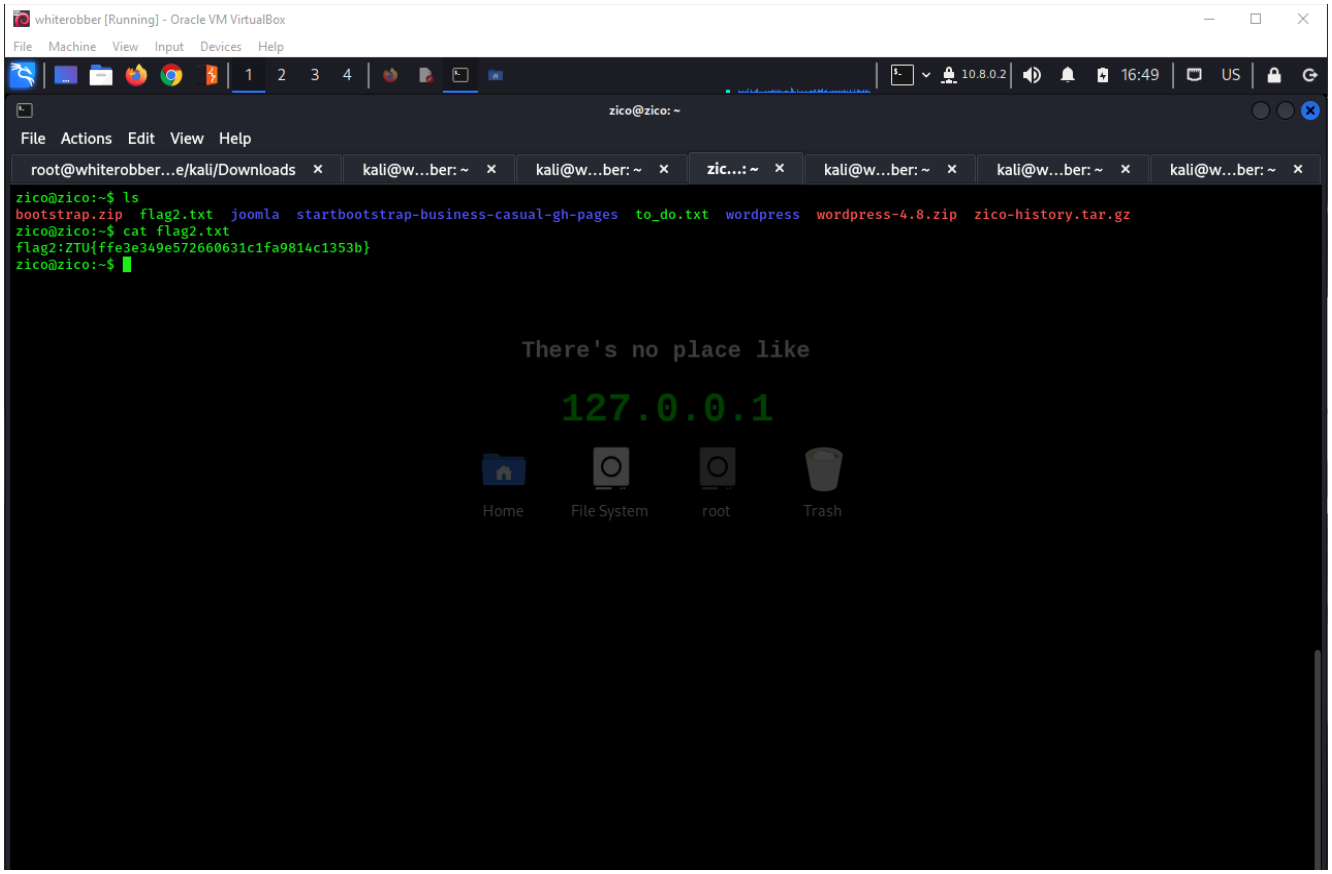


Рисунок 3.5.107 – Другий прапор успішно знайдено

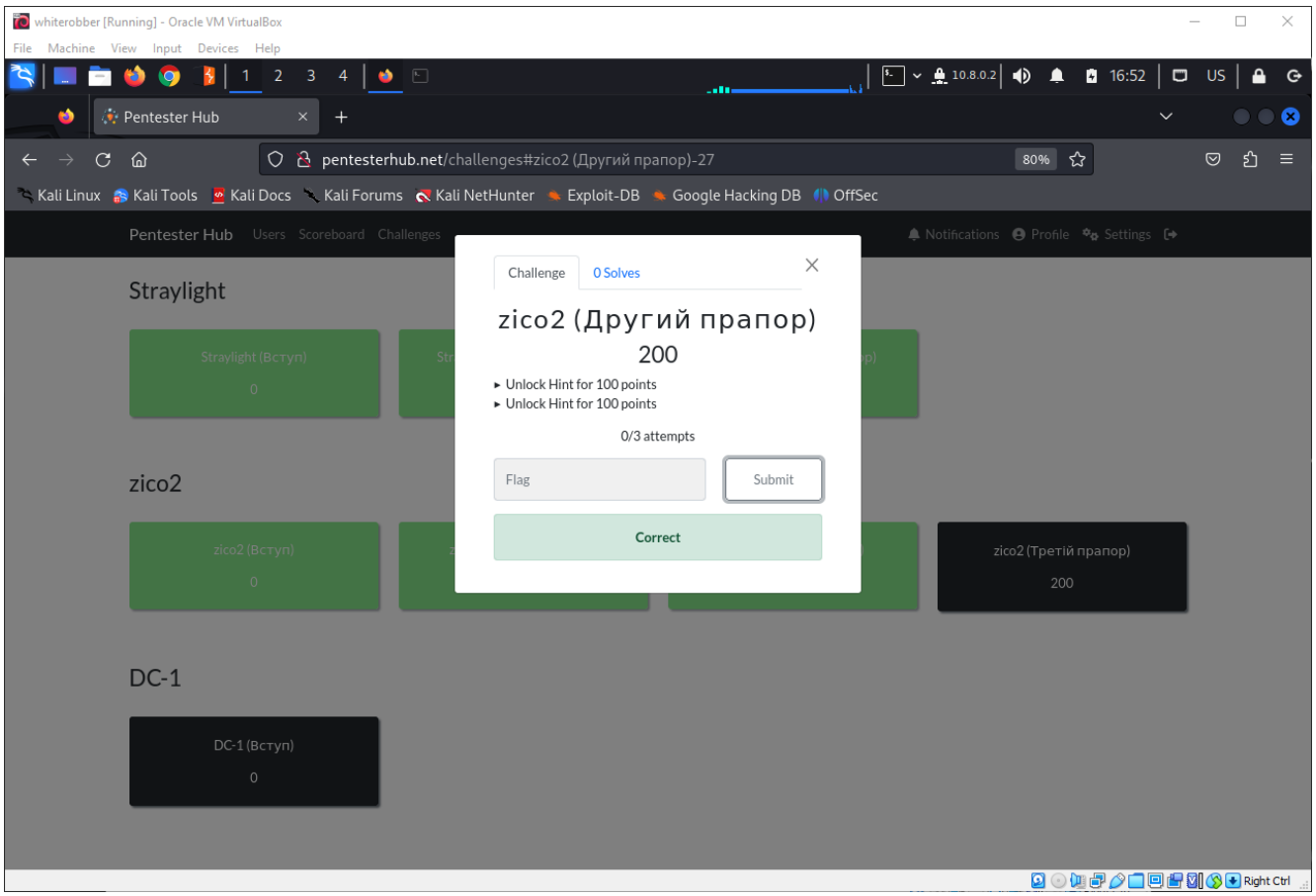


Рисунок 3.5.108 – Прапор успішно надіслано, завдання зараховано

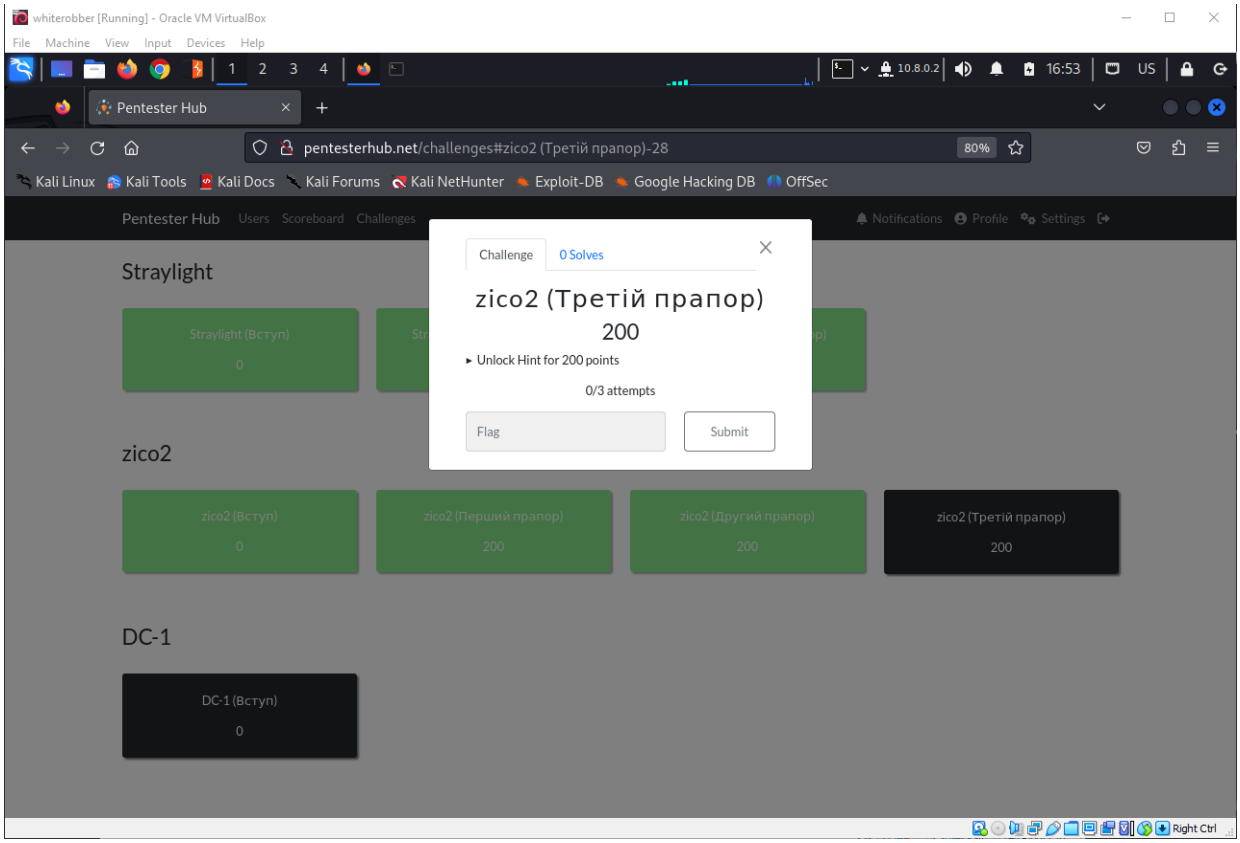


Рисунок 3.5.109 – Четверте/фінальне завдання zico2 (Третій прапор)

Успішне підключення по SSH дає змогу перевірити, які команди користувач zico2 може запускати від імені root.

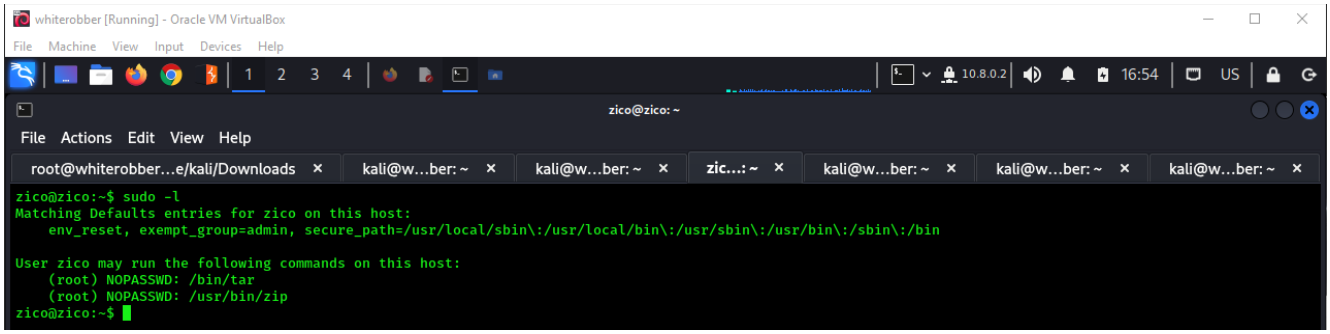


Рисунок 3.5.110 – Виконання команди sudo -l

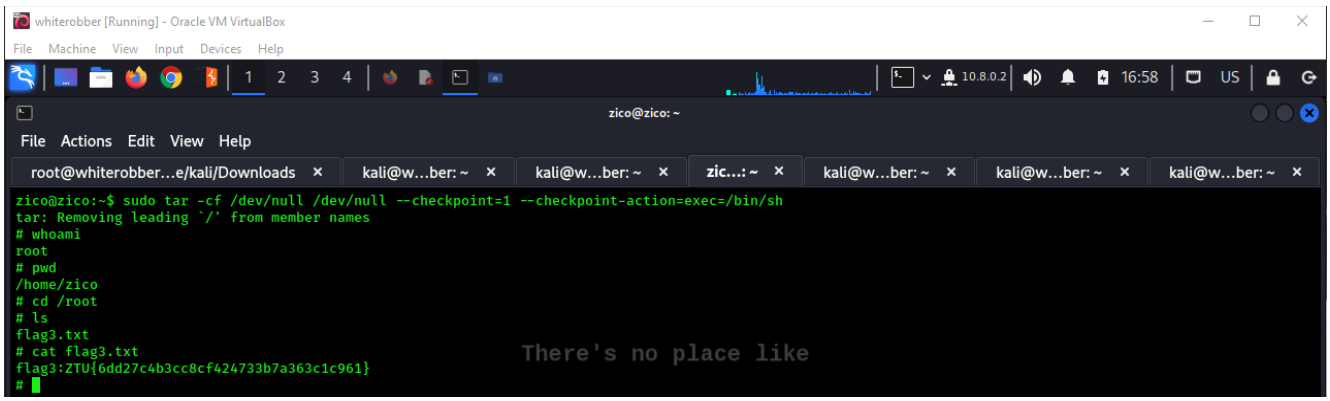


Рисунок 3.5.111 – Підвищення привілеїв успішне, третій прапор успішно знайдено

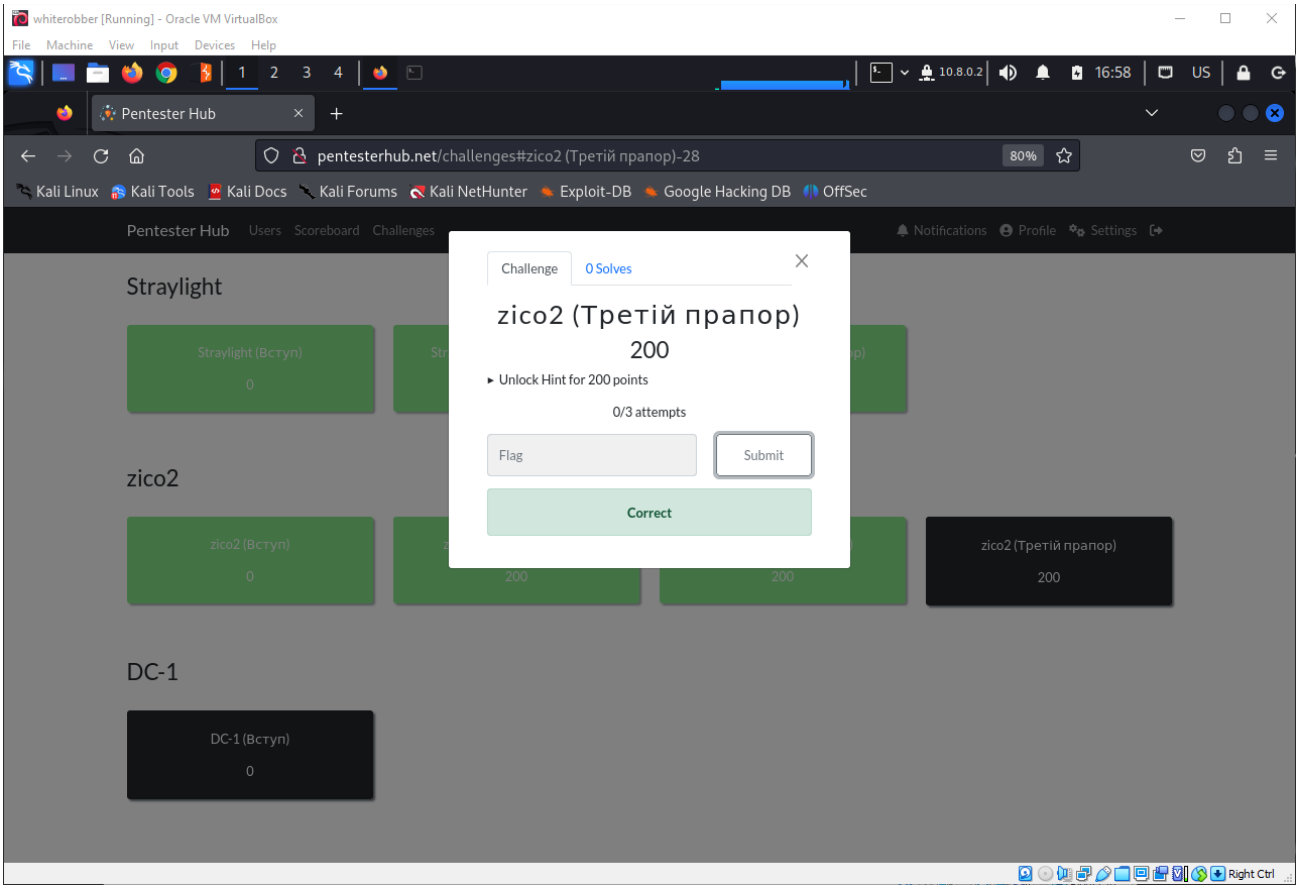


Рисунок 3.5.112 – Прапор успішно надіслано, завдання зараховано

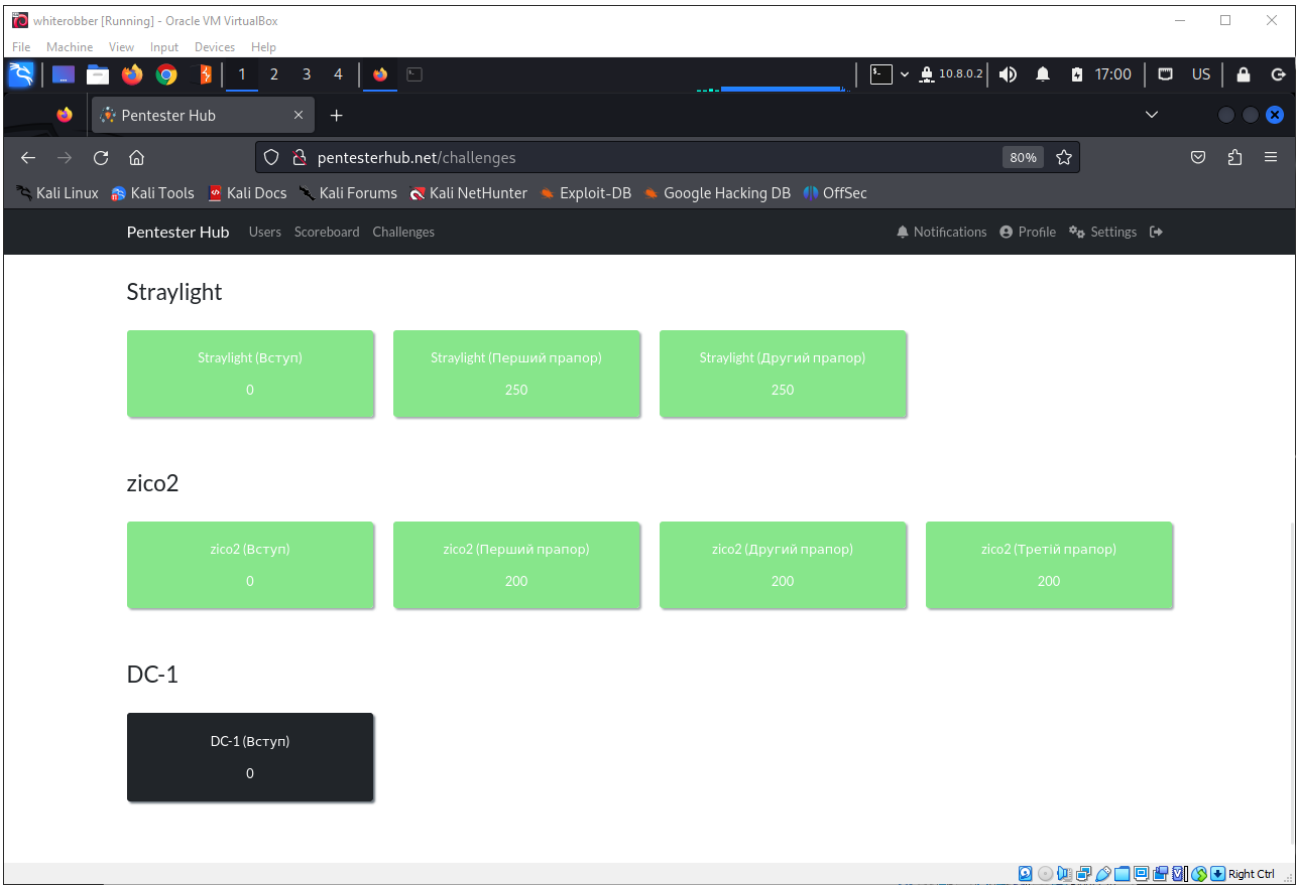


Рисунок 3.5.113 – Серія завдань zico2 успішно протестована (вирішена)

Розпочнемо вирішення серії завдань для машини DC-1.

Зазначимо, процес ознайомлення з описом завдання, виконання підключення до OpenVPN сервера, успішне відкриття веб-сайту машини DC-1 та перегляд безкоштовної підказки для того, щоб надіслати прапор для виконання завдання та відкрити наступне завдання в серії аналогічний до SolidState.

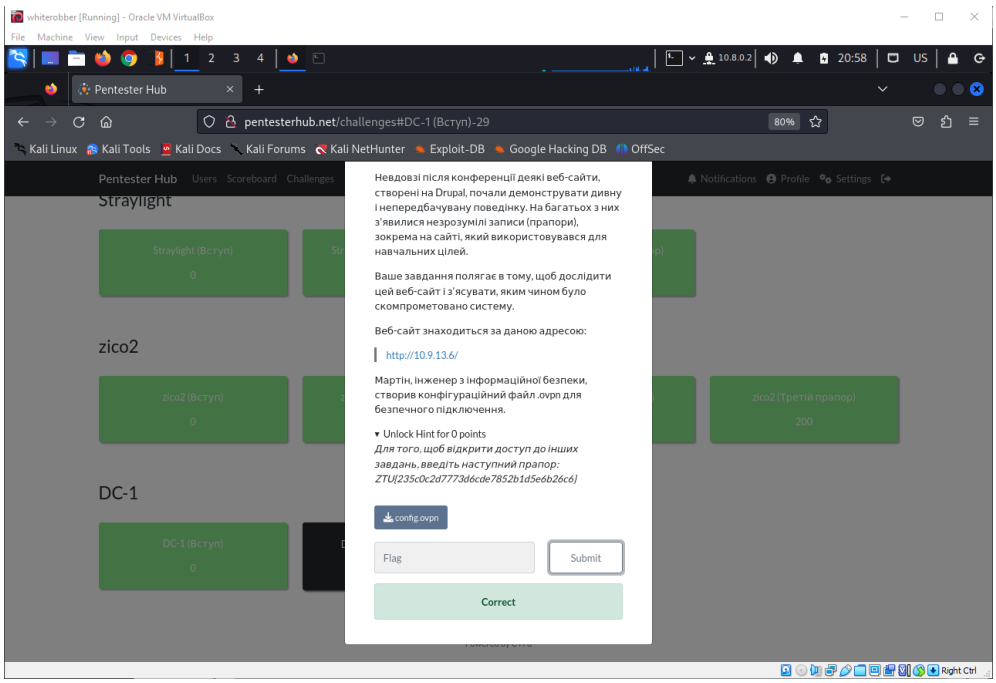


Рисунок 3.5.114 – Перше завдання DC-1 (Вступ), прапор успішно надіслано, завдання зараховано

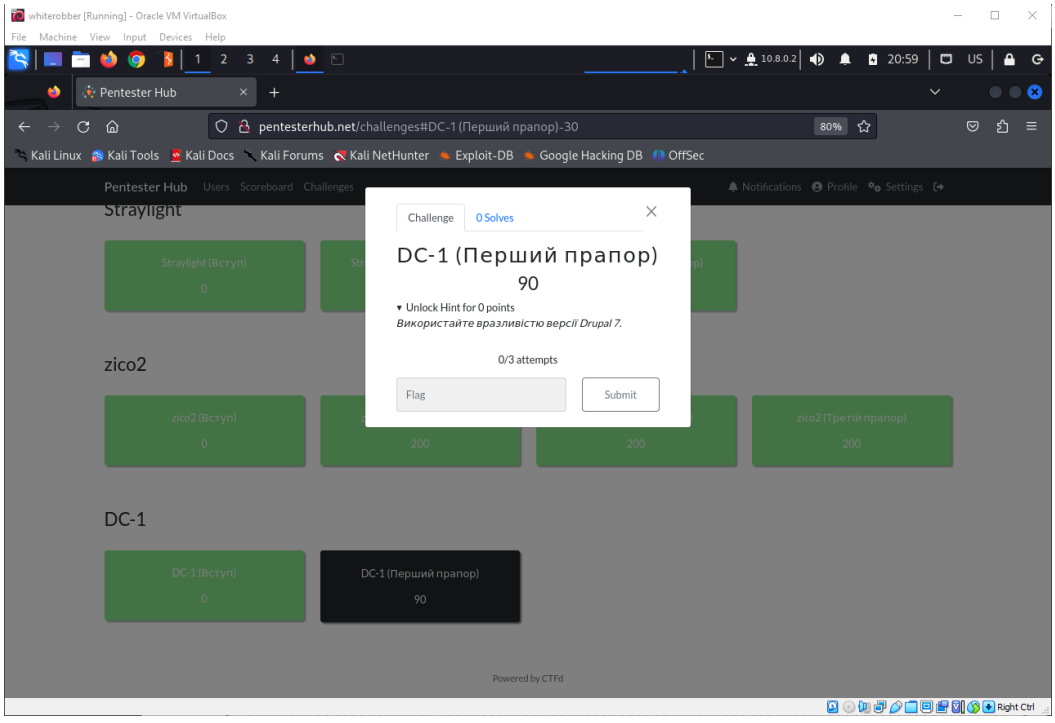


Рисунок 3.5.115 – Друге завдання DC-1 (Перший прапор)

Звертаючи увагу на безкоштовну підказку можна зробити висновок, що потрібно використати вразливість для Drupal 7, для цього використаємо утиліту msfconsole (Metasploit).

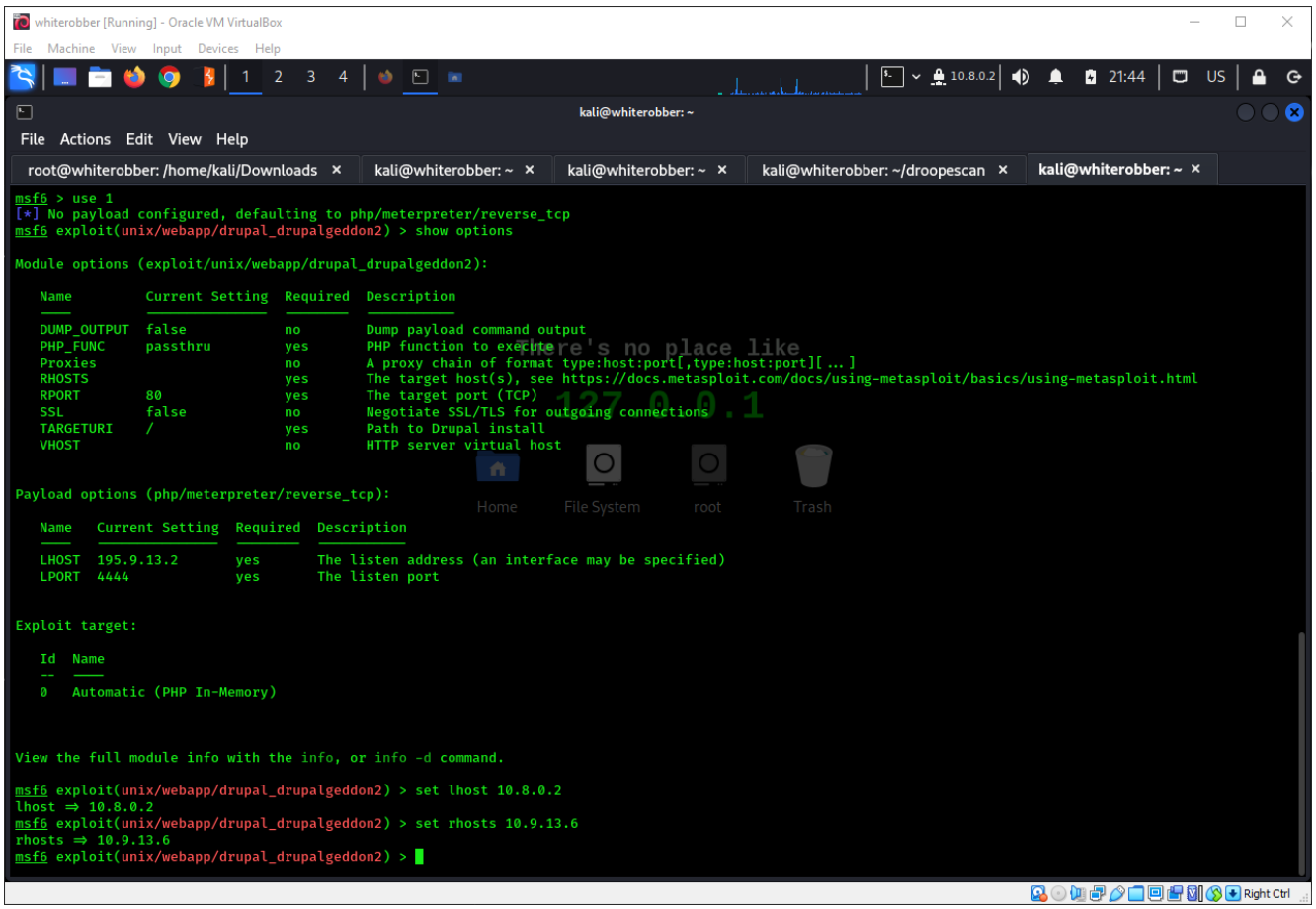


Рисунок 3.5.116 – Використано модуль unix/webapp/drupal_drupalgeddon2 в msfconsole (Metasploit)

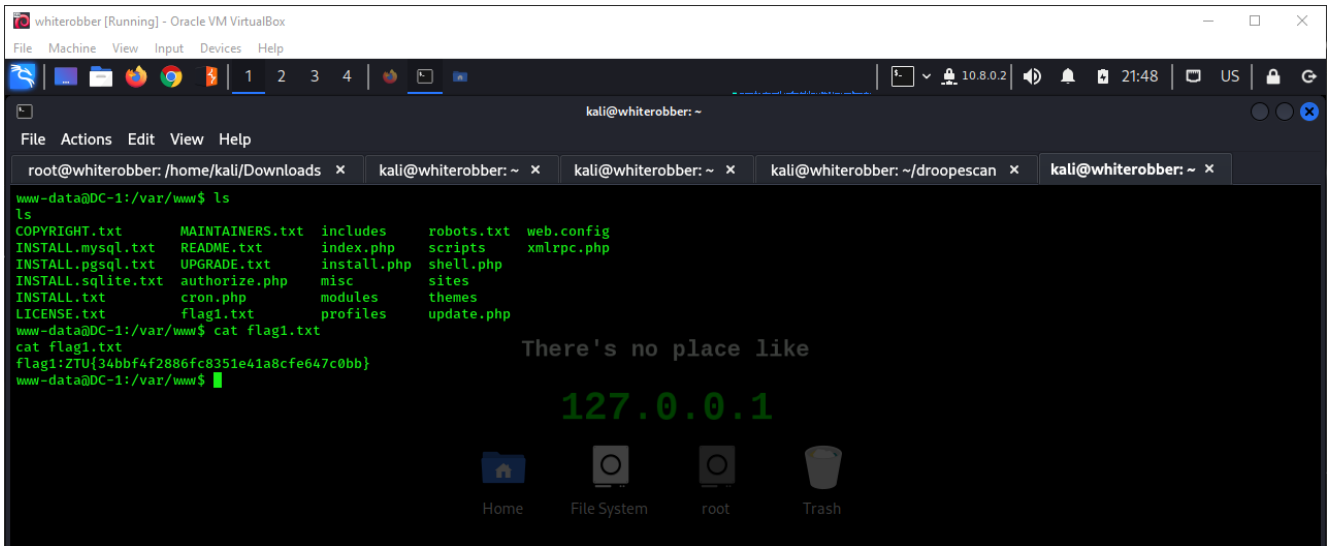


Рисунок 3.5.117 – Модуль спрацював, підключення успішне, перший прапор успішно знайдено

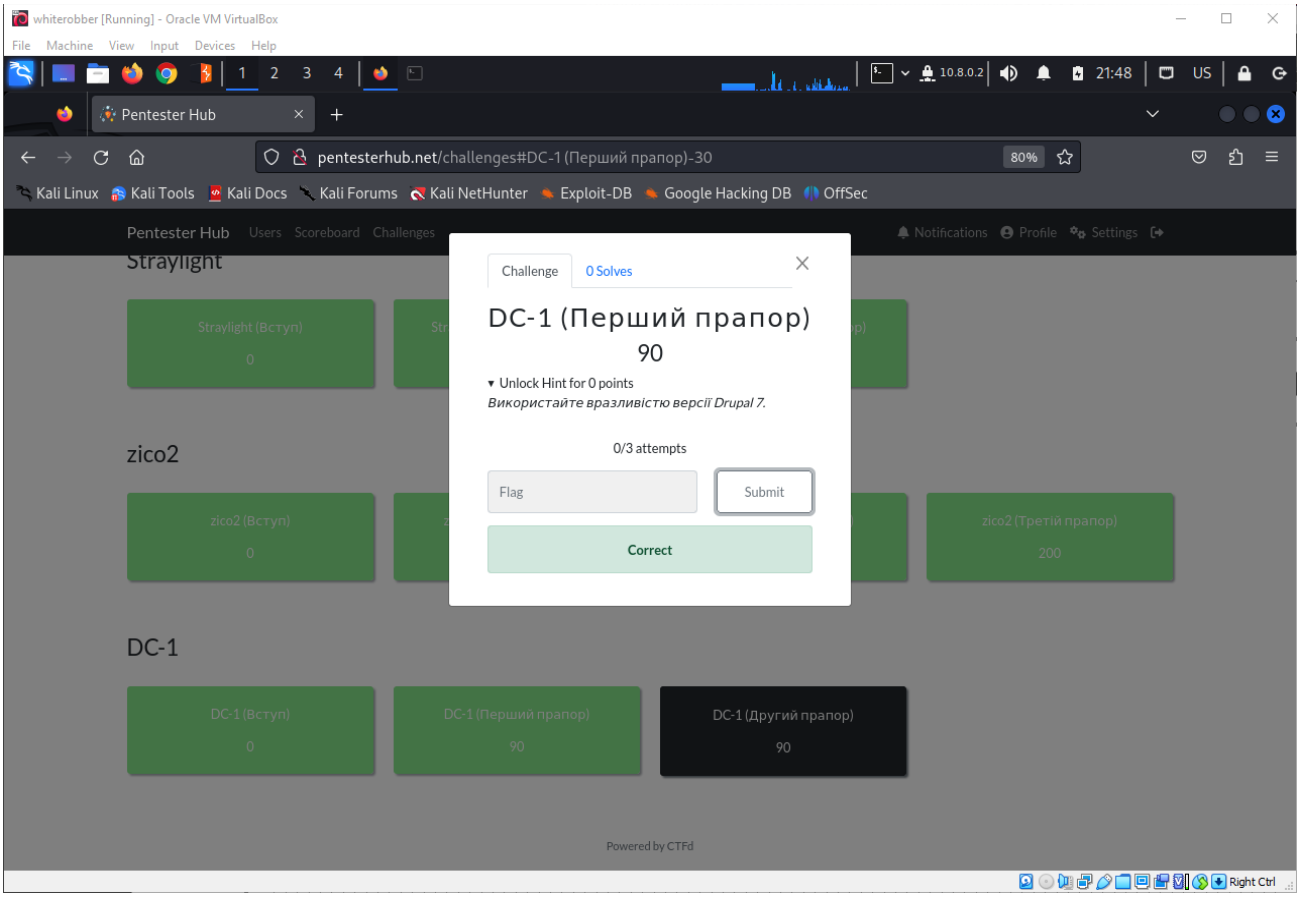


Рисунок 3.5.118 – Прапор успішно надіслано, завдання зараховано

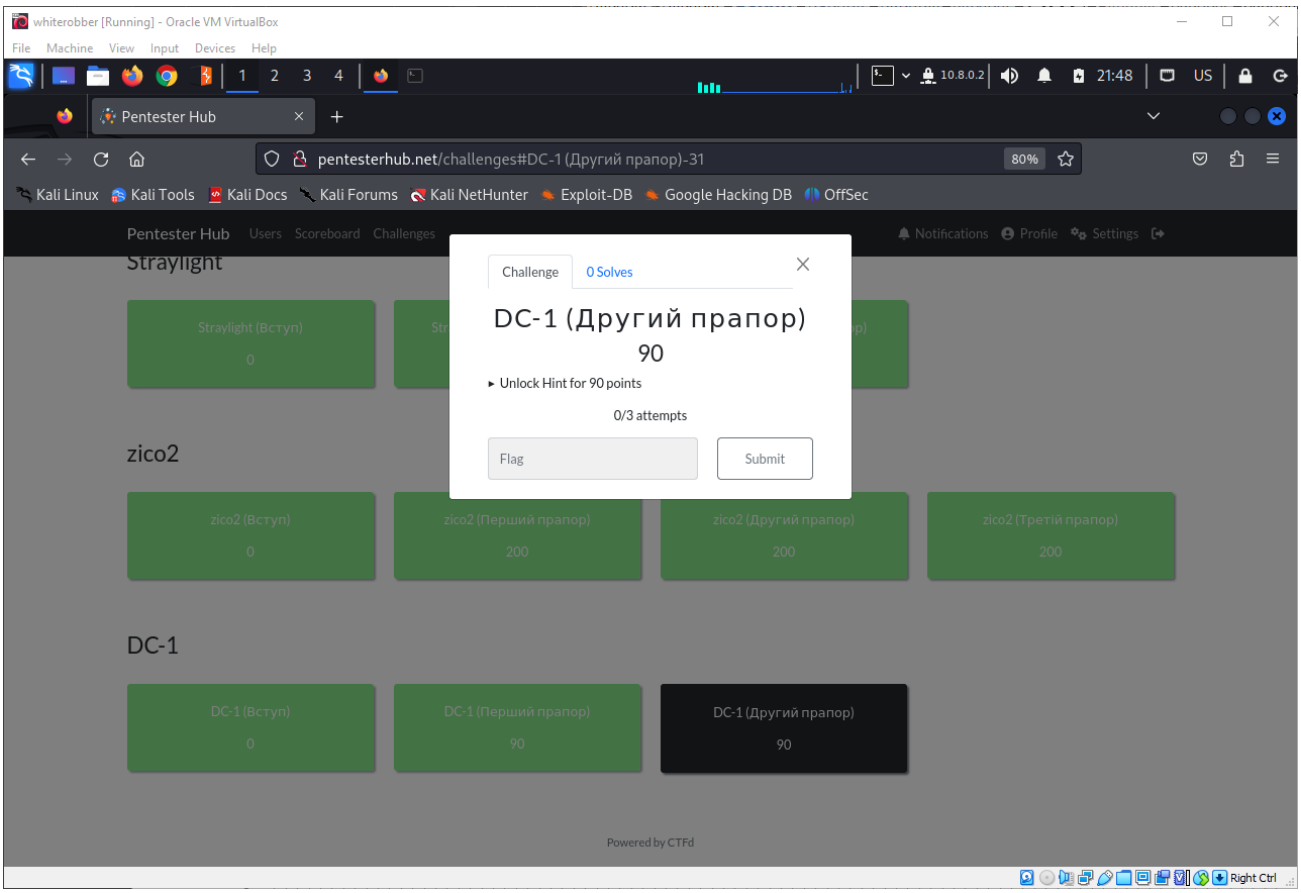


Рисунок 3.5.119 – Третє завдання DC-1 (Другий прапор)

Оскільки сайт використовує CMS Drupal, спробуємо переглянути файл конфігурації, який зазвичай знаходиться за наступним шляхом sites/default.

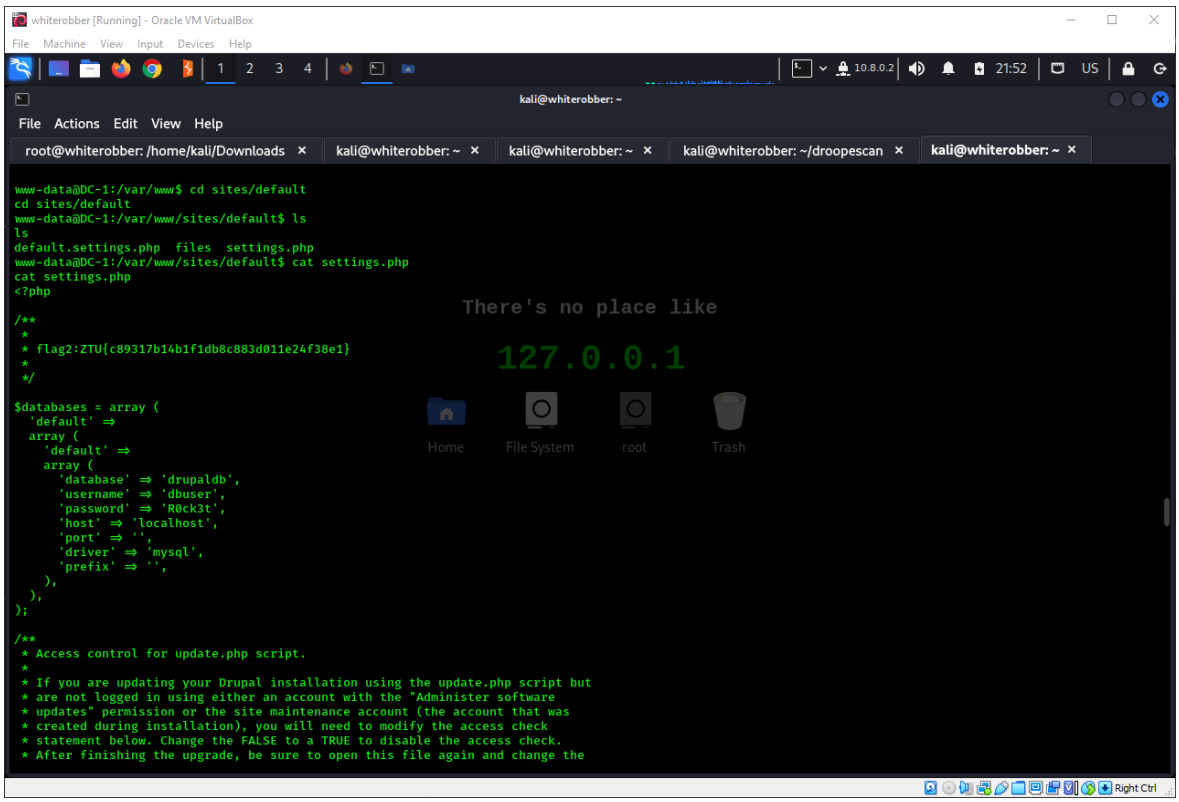


Рисунок 3.5.120 – Перегляд конфігураційного файлу settings.php, другий прапор успішно знайдено

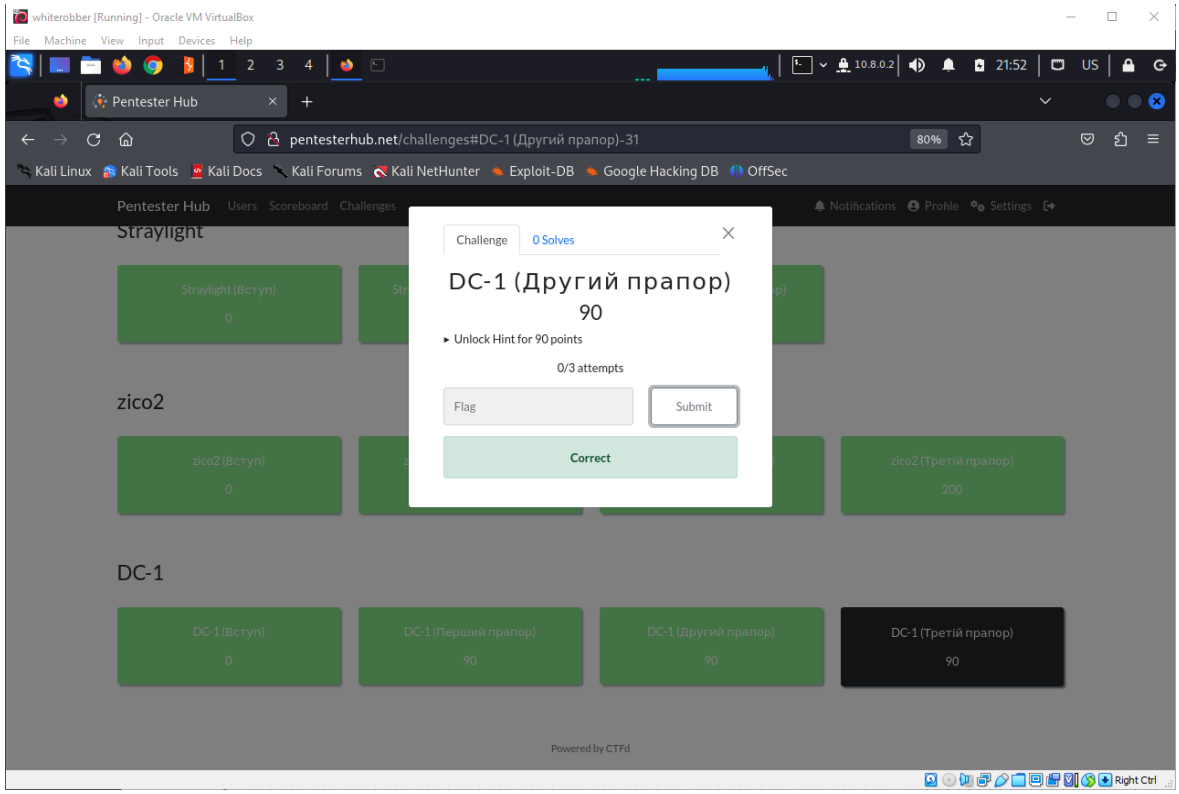


Рисунок 3.5.121 – Прапор успішно надіслано, завдання зараховано

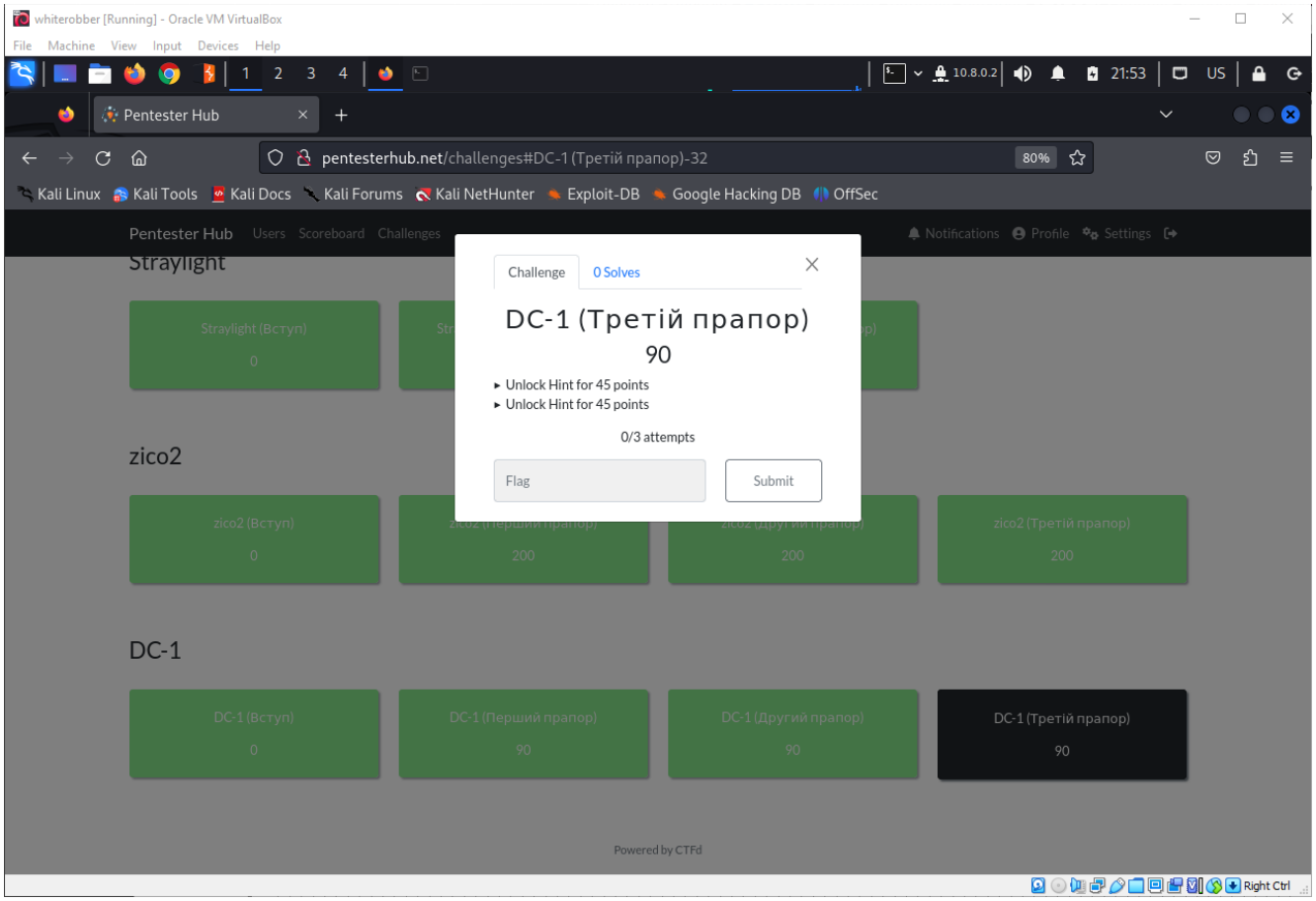


Рисунок 3.5.122 – Четверте завдання DC-1 (Третій прапор)

Оскільки було знайдено облікові дані, а виконати підключення по SSH не вдалось, спробуємо виконати вхід в MySQL.

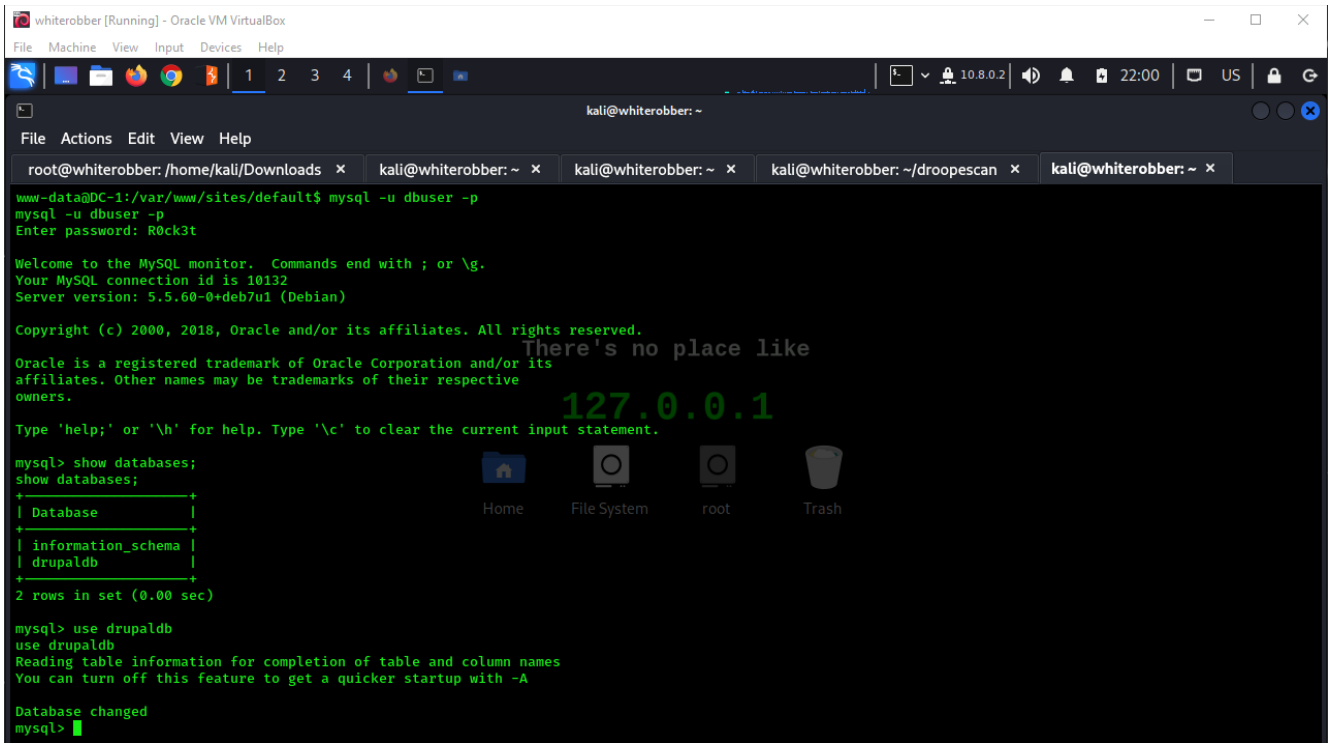


Рисунок 3.5.123 – Успішне підключення до MySQL

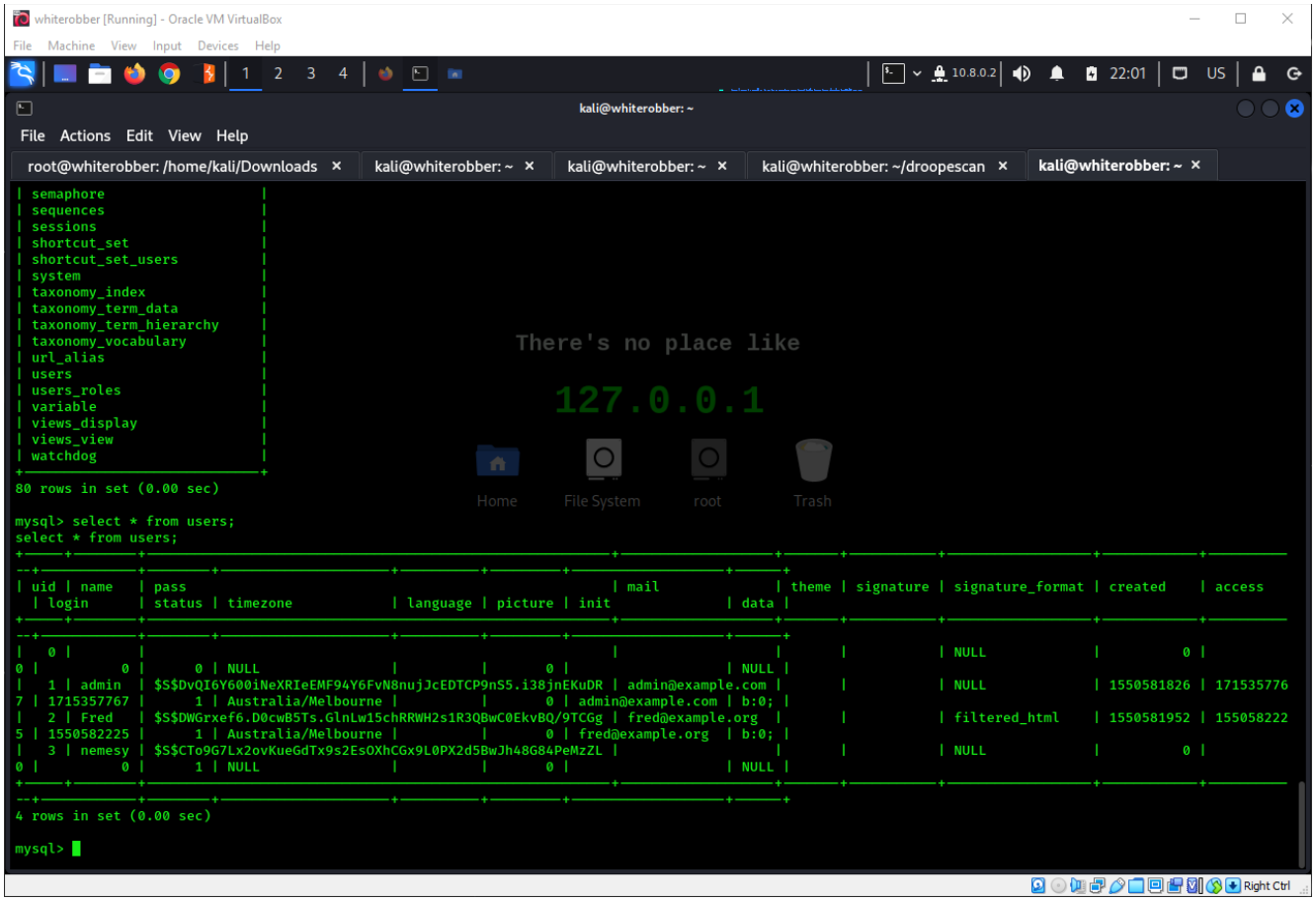


Рисунок 3.5.124 – Перегляд таблиці користувача users,
знайдено декілька хешів

Спробуємо виконати злам знайдених хешів в таблиці users за допомогою утиліти John the Ripper та словника rockyou.

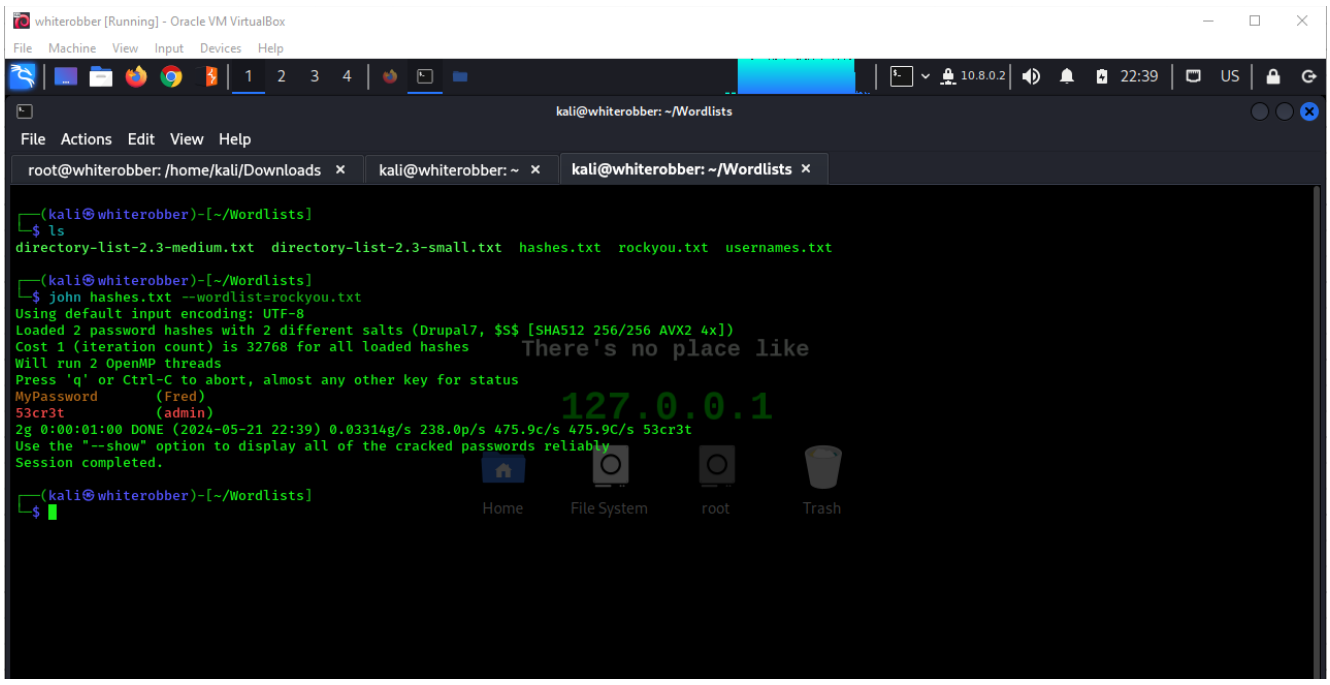


Рисунок 3.5.125 – Успішно зламані хеші

Отримавши облікові дані користувача admin, спробуємо виконати вхід в admin панель.

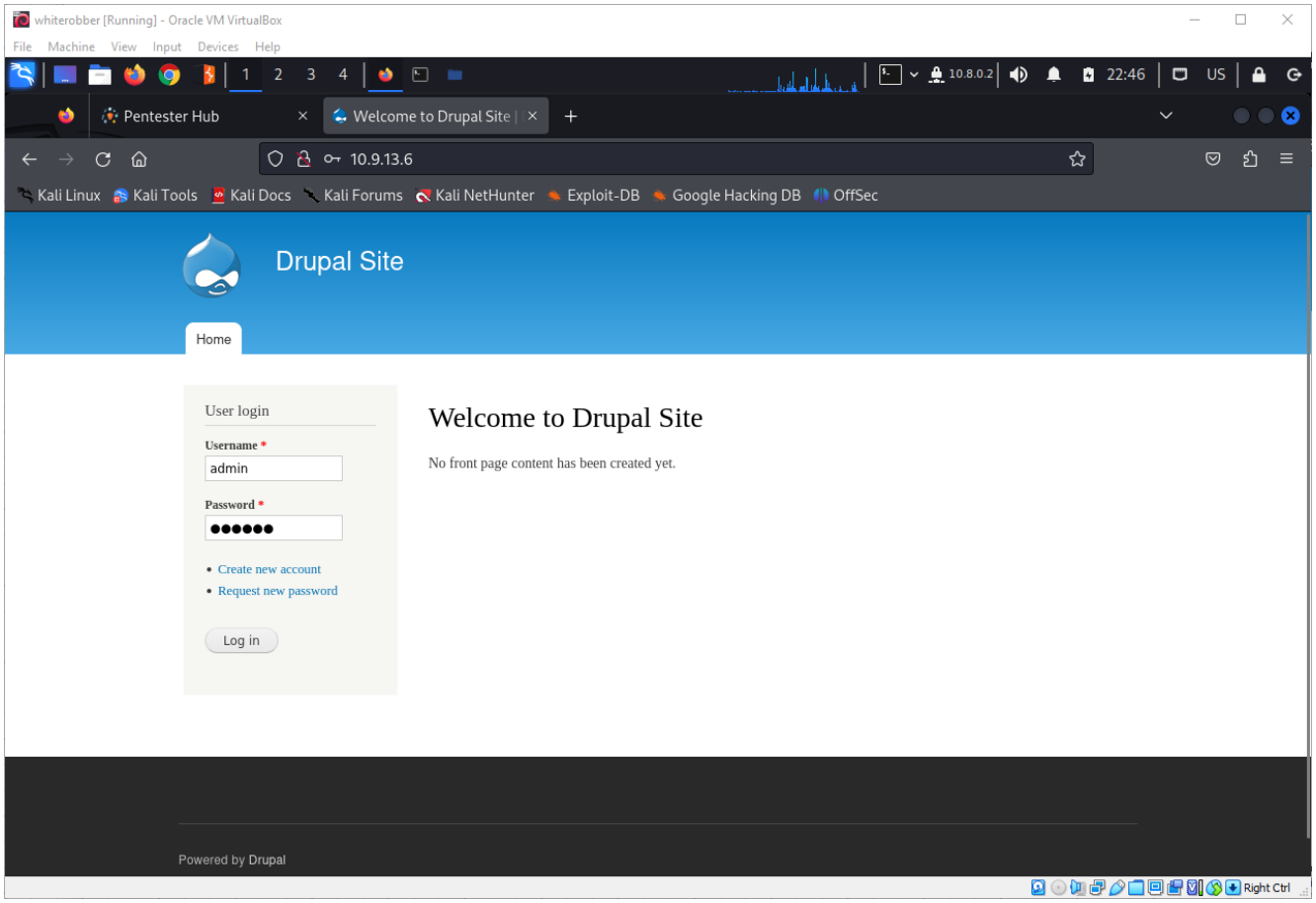


Рисунок 3.5.126 – Виконання входу в admin панель

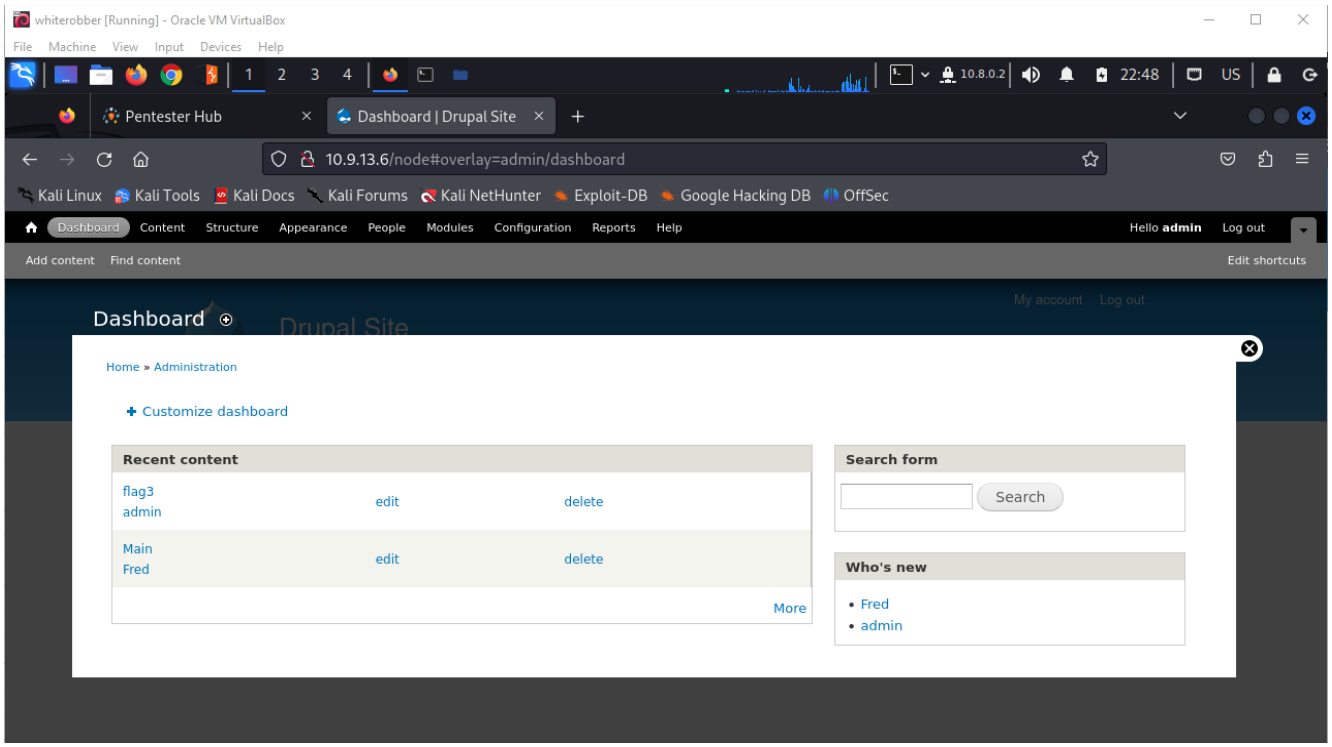


Рисунок 3.5.127 – Вхід успішно виконано, отримано доступ до admin панелі

Звертаючи увагу на Dashboard, було помічено недавній контент з назвою flag3, спробуємо переглянути його.

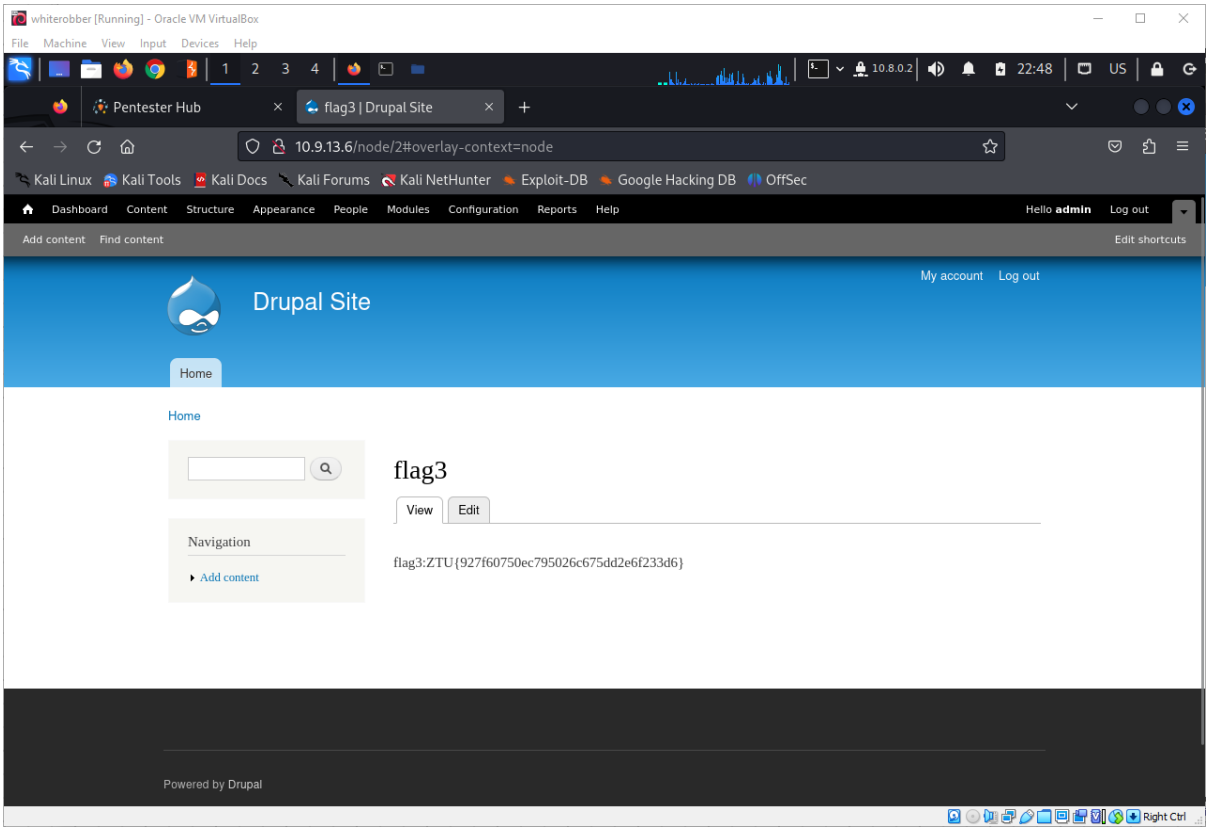


Рисунок 3.5.128 – Контент flag3, третій прапор успішно знайдено

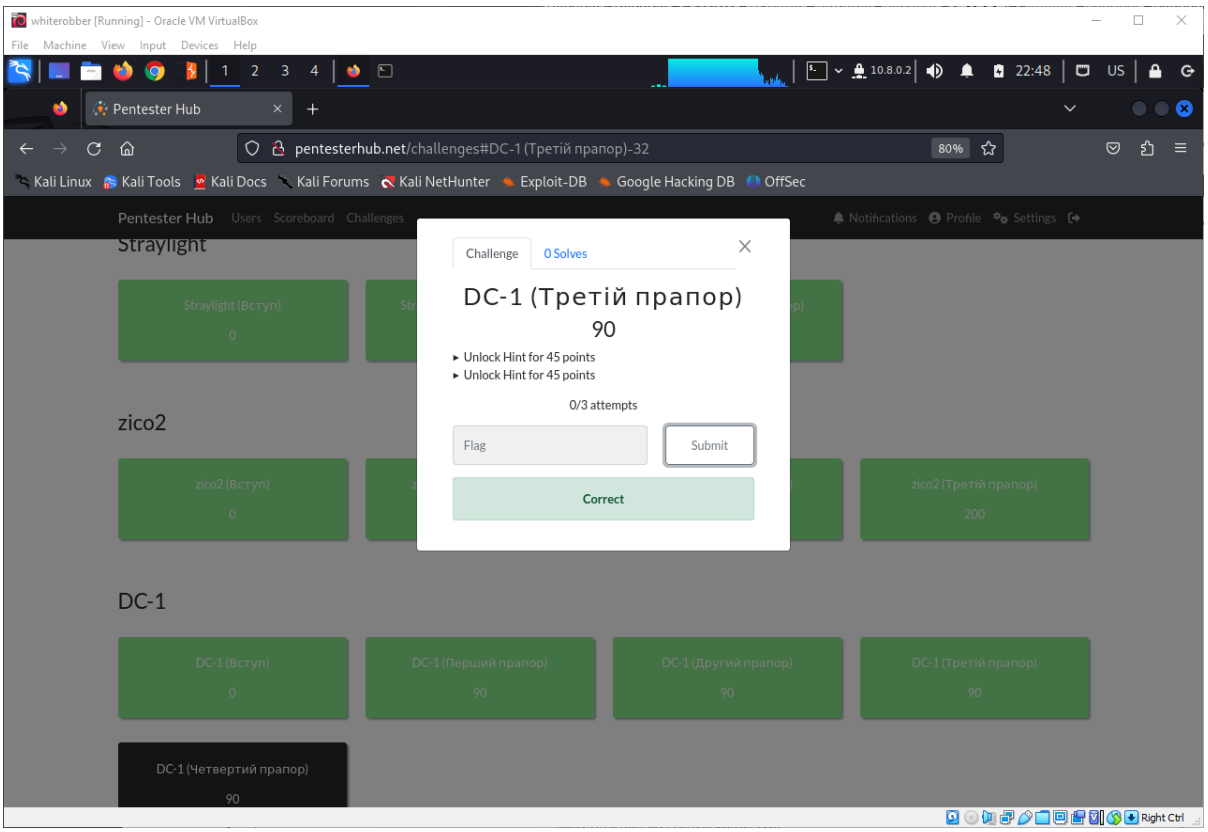


Рисунок 3.5.129 – Прапор успішно надіслано, завдання зараховано

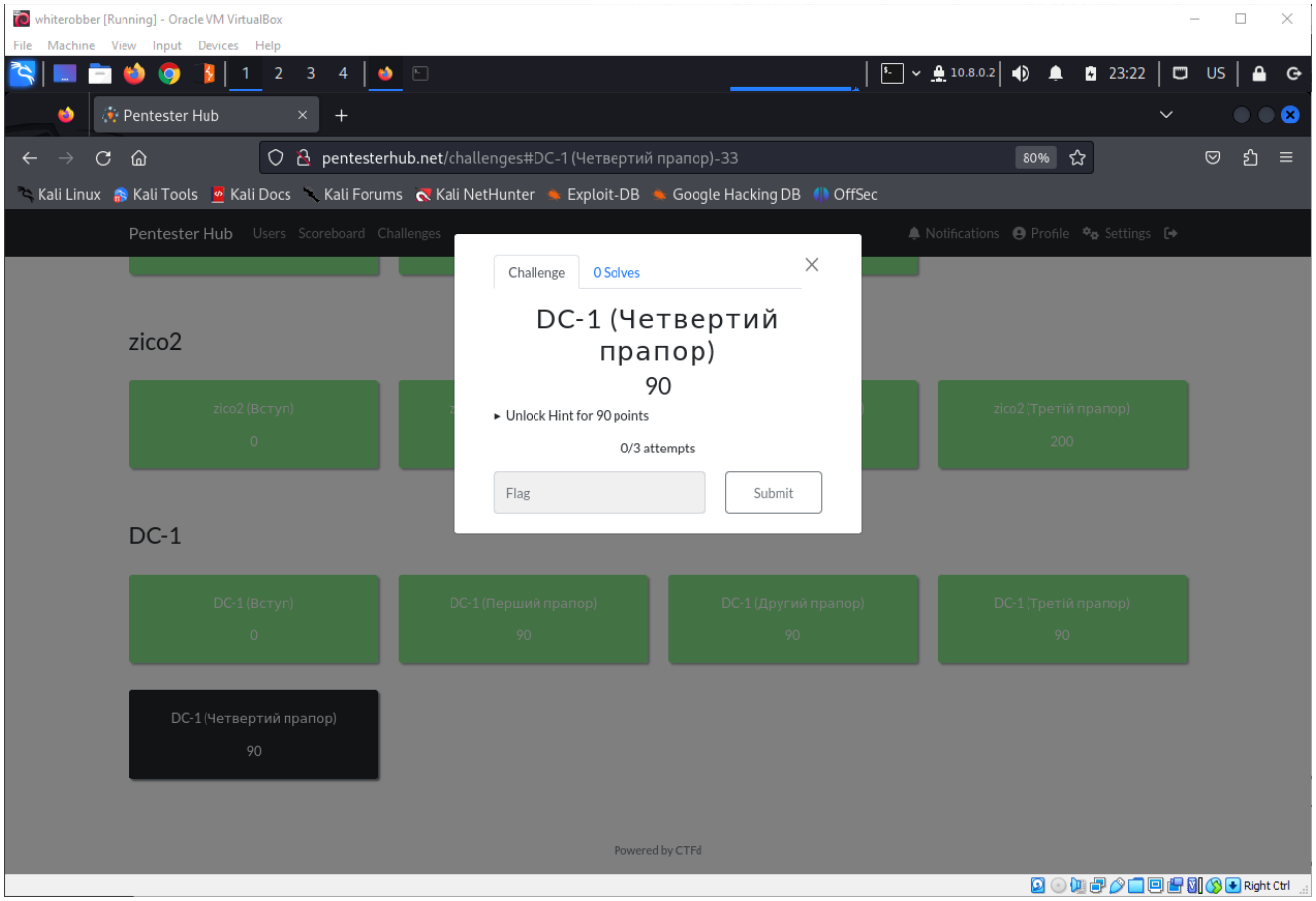


Рисунок 3.5.130 – П'яте завдання DC-1 (Четвертий прапор)

Переглянемо директорію home, можливо там є щось цікаве.

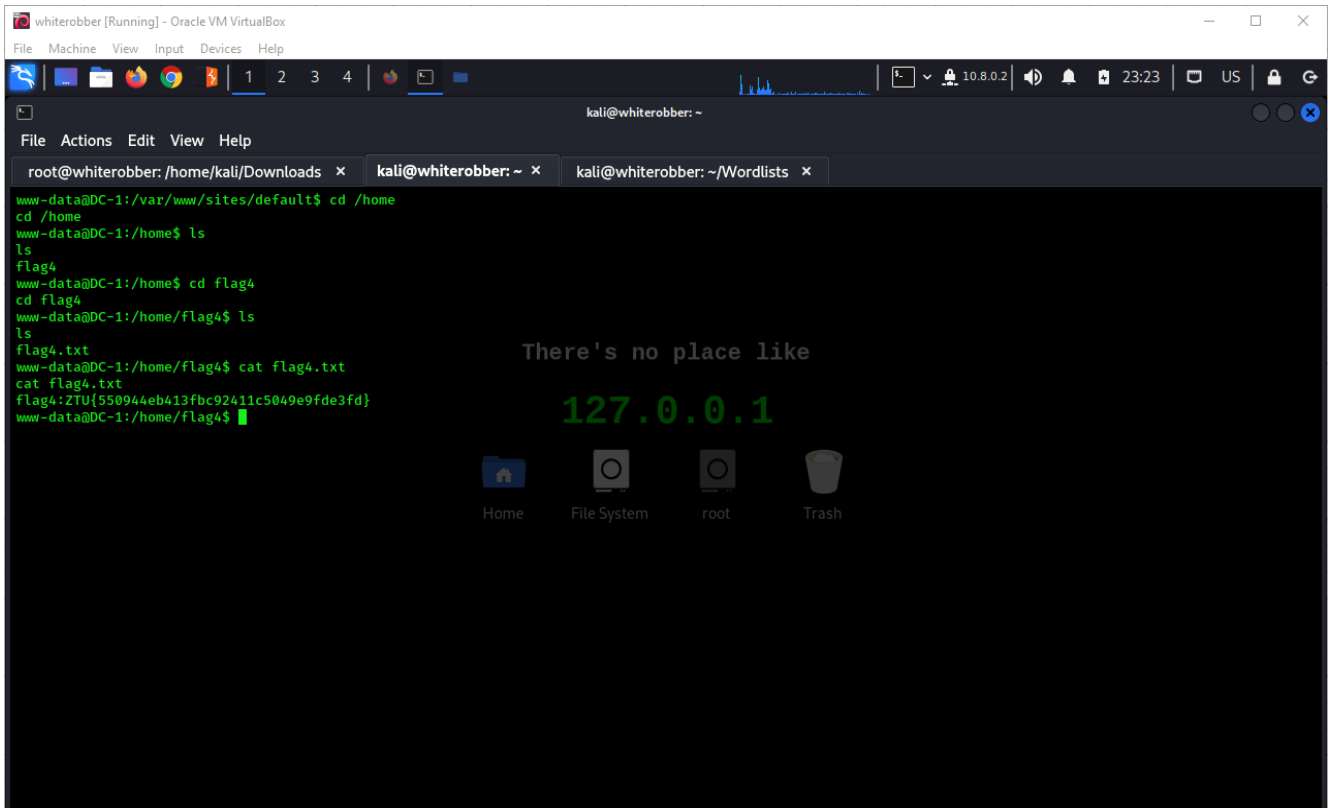


Рисунок 3.5.131 – Четвертий прапор успішно знайдено

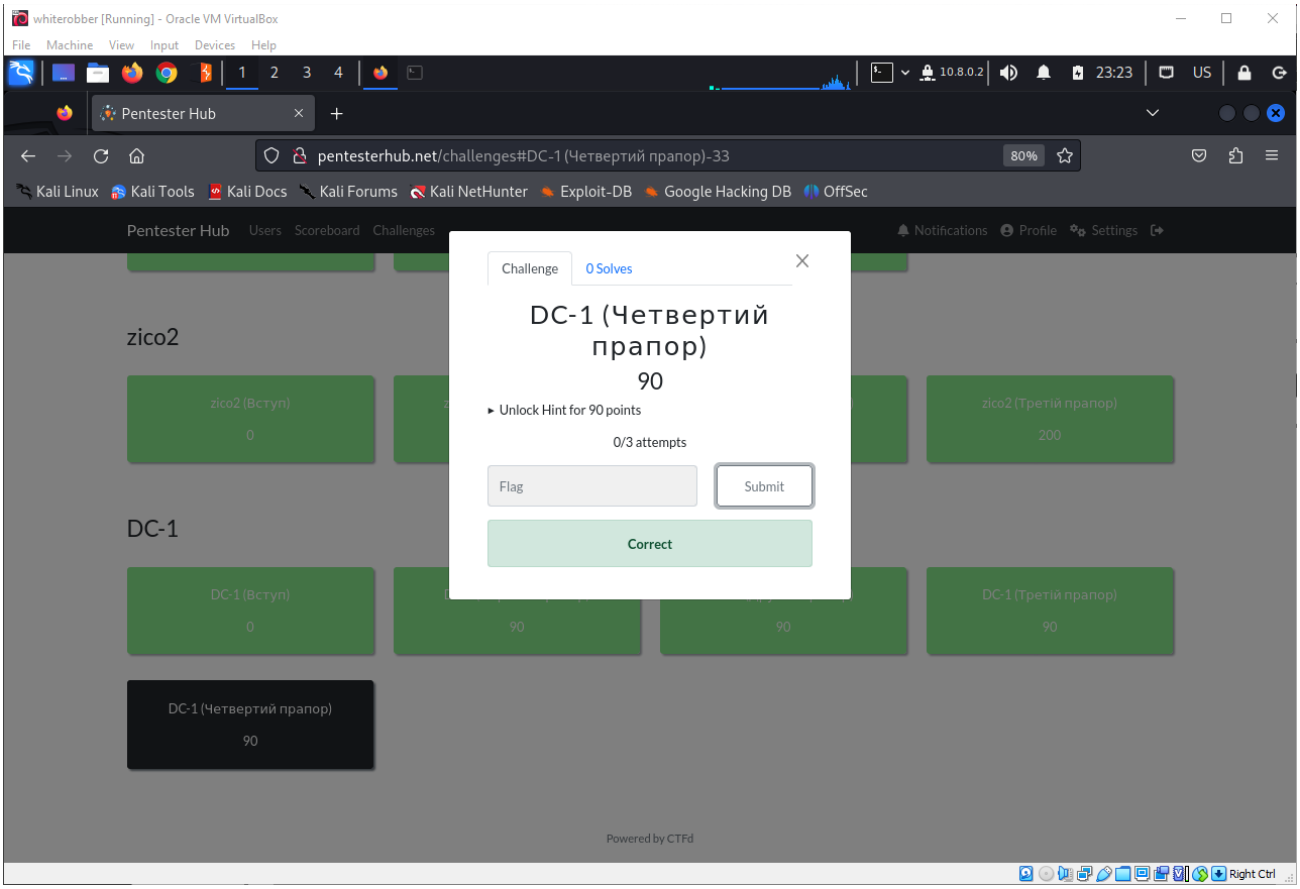


Рисунок 3.5.132 – Прапор успішно надіслано, завдання зараховано

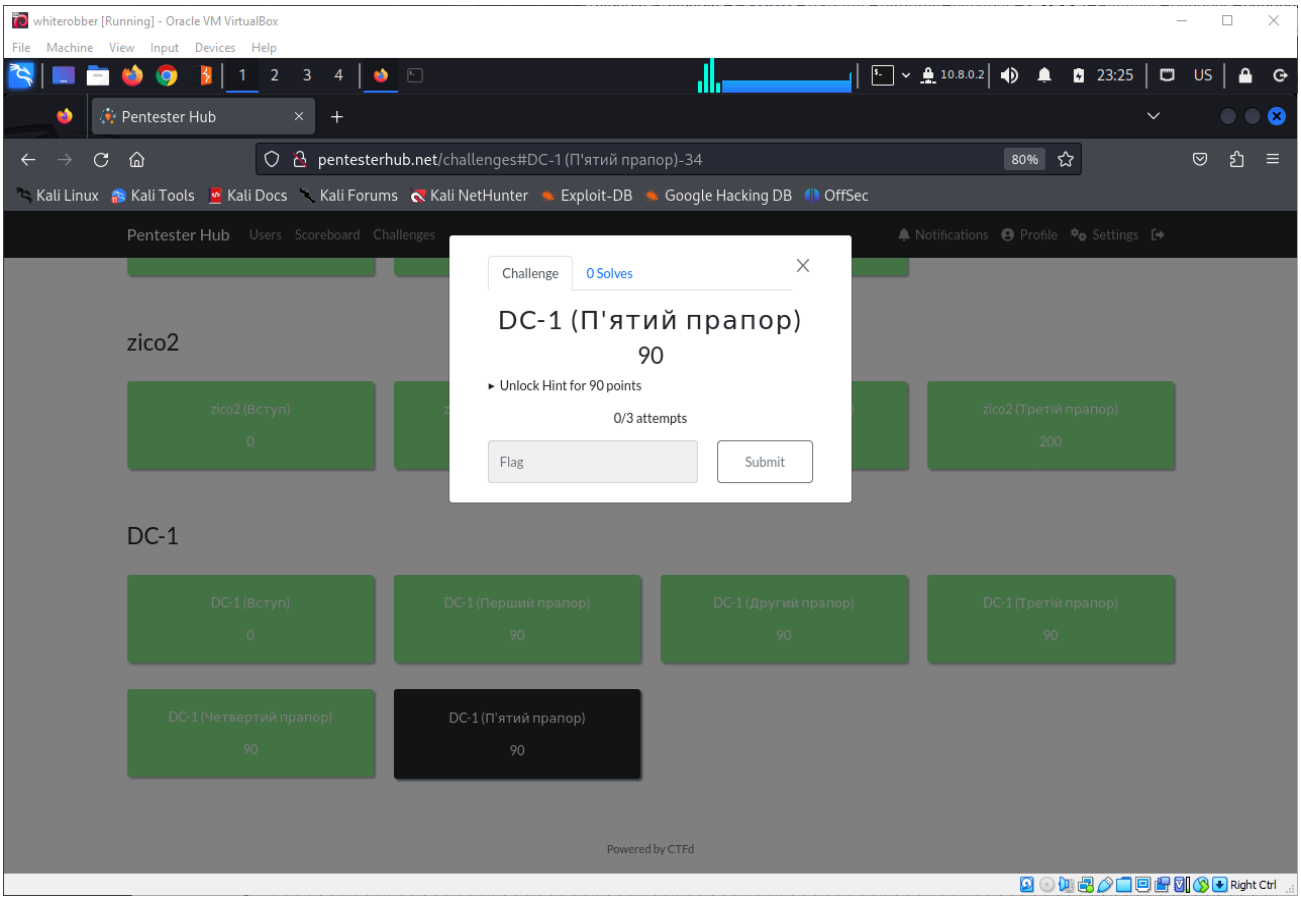


Рисунок 3.5.133 – Шосте/Фінальне завдання DC-1 (П'ятий прапор)

Виконаємо пошук файлів та команд, які можна запускати від імені root.

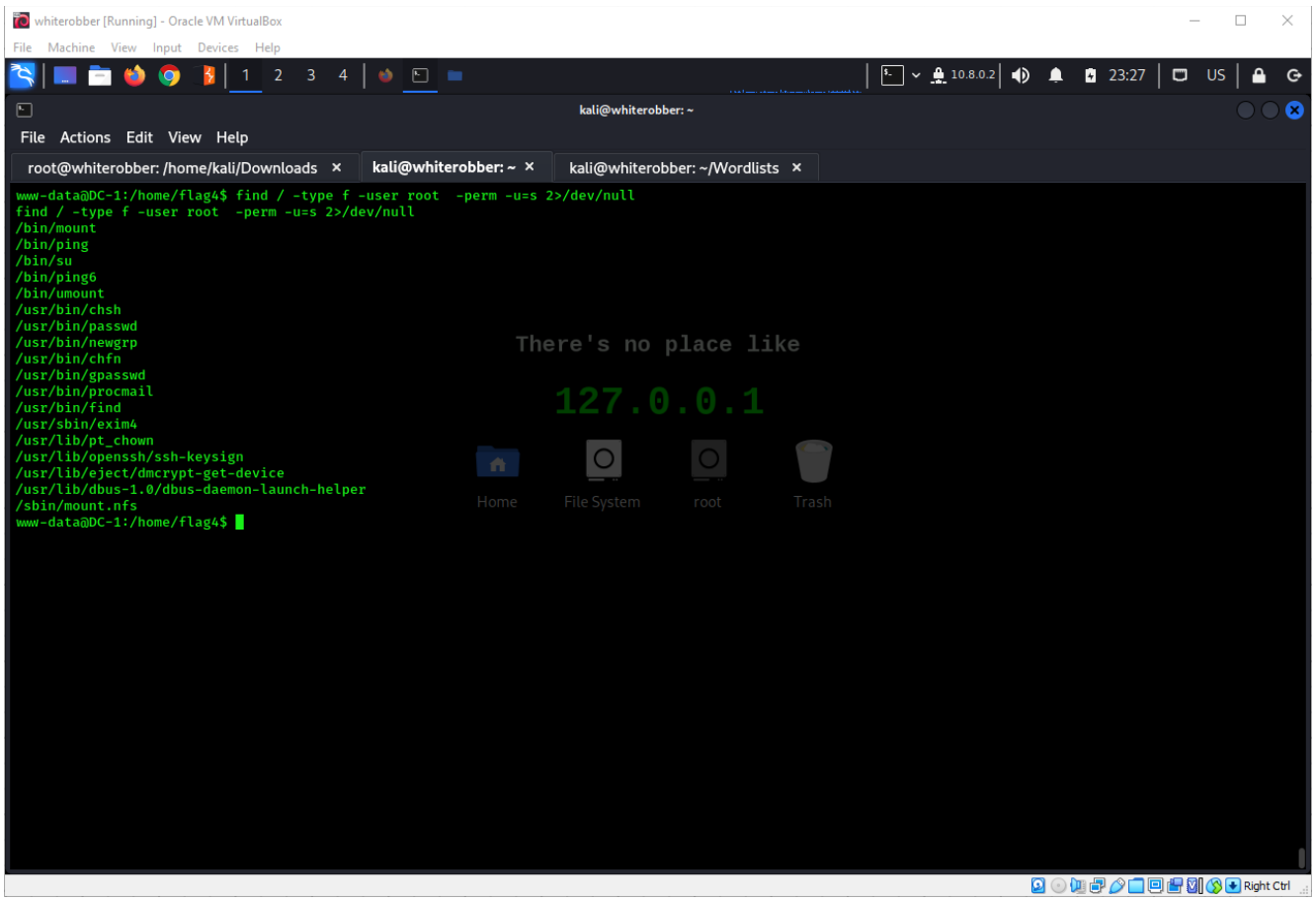


Рисунок 3.5.134 – Виконання команди find / -type f -user root -perm -u=s 2>/dev/null

Вивід команди показав, що можна використати find для підвищення привілеїв.

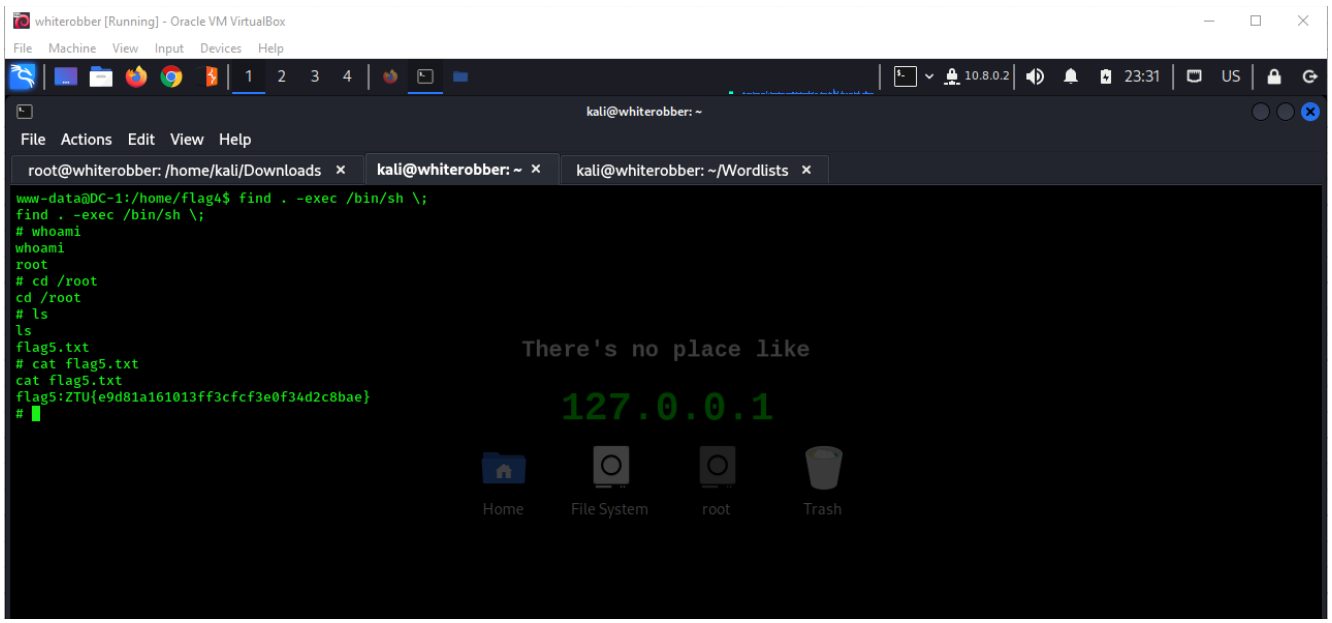


Рисунок 3.5.135 – Підвищення привілеїв успішне,
п'ятий прапор успішно знайдено

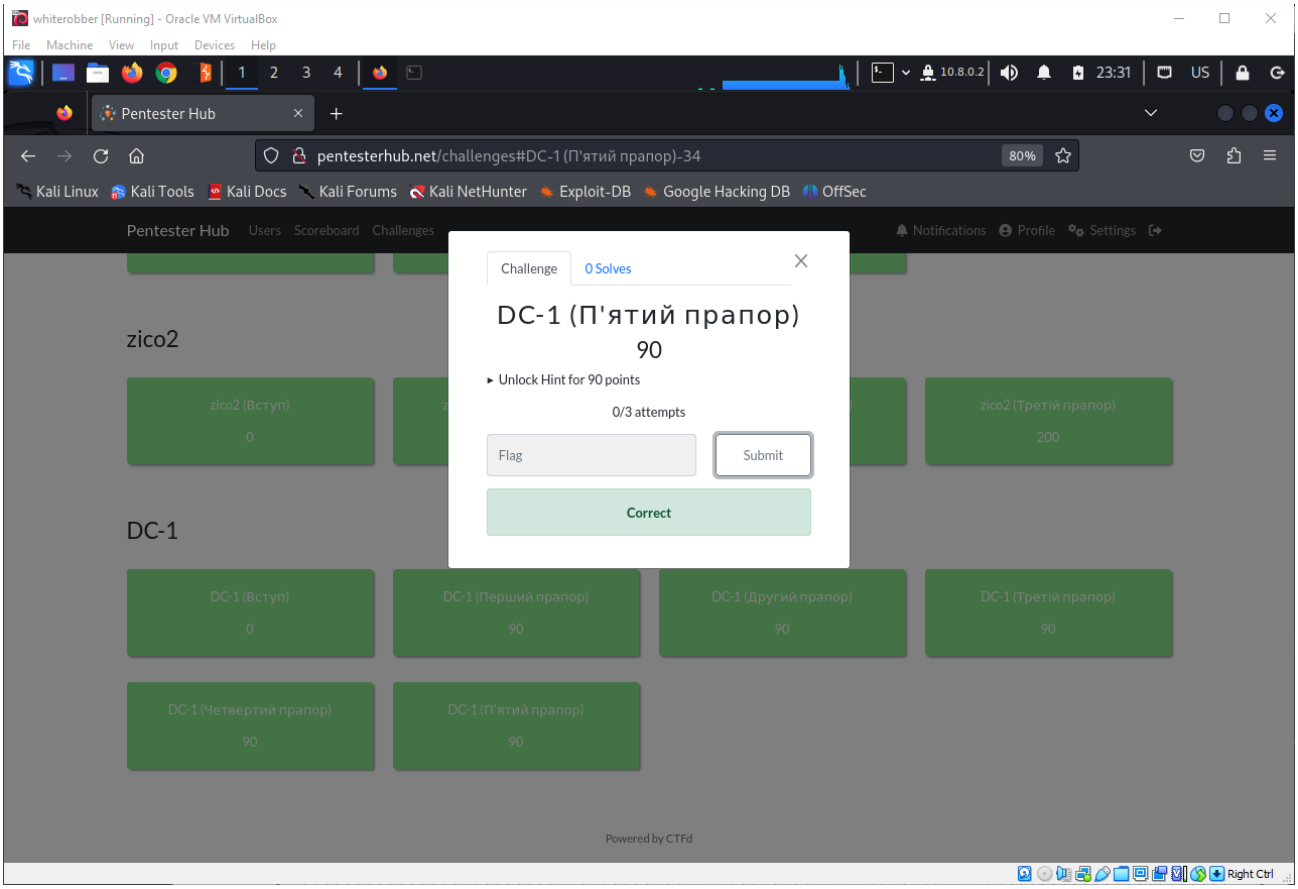


Рисунок 3.5.136 – Прапор успішно надіслано, завдання зараховано

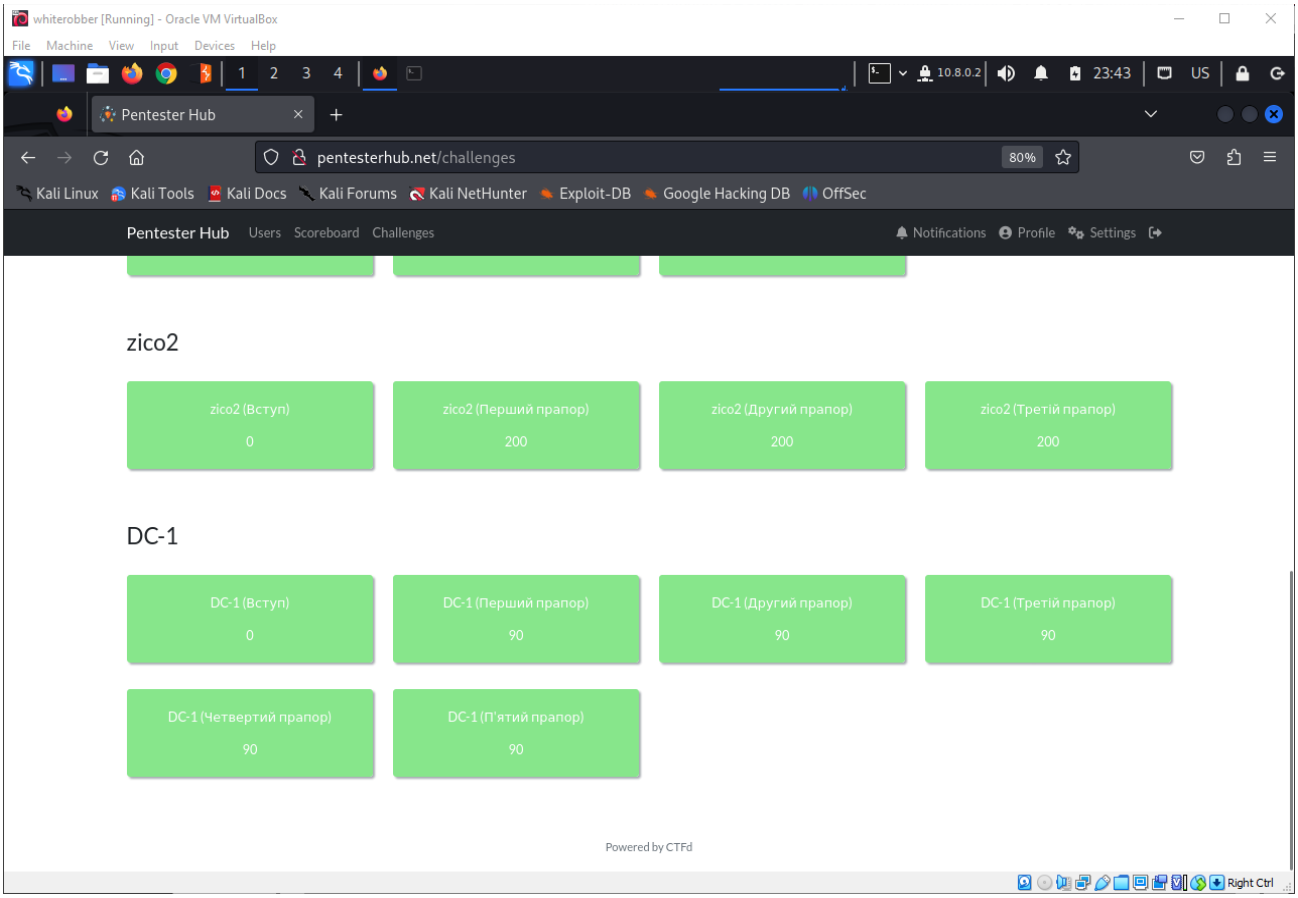


Рисунок 3.5.137 – Серія завдань DC-1 успішно протестована (вирішена)

Функціонал та серії завдань на платформі протестовані, все працює коректно.

Переглянемо профіль користувача vadym-nesterov.

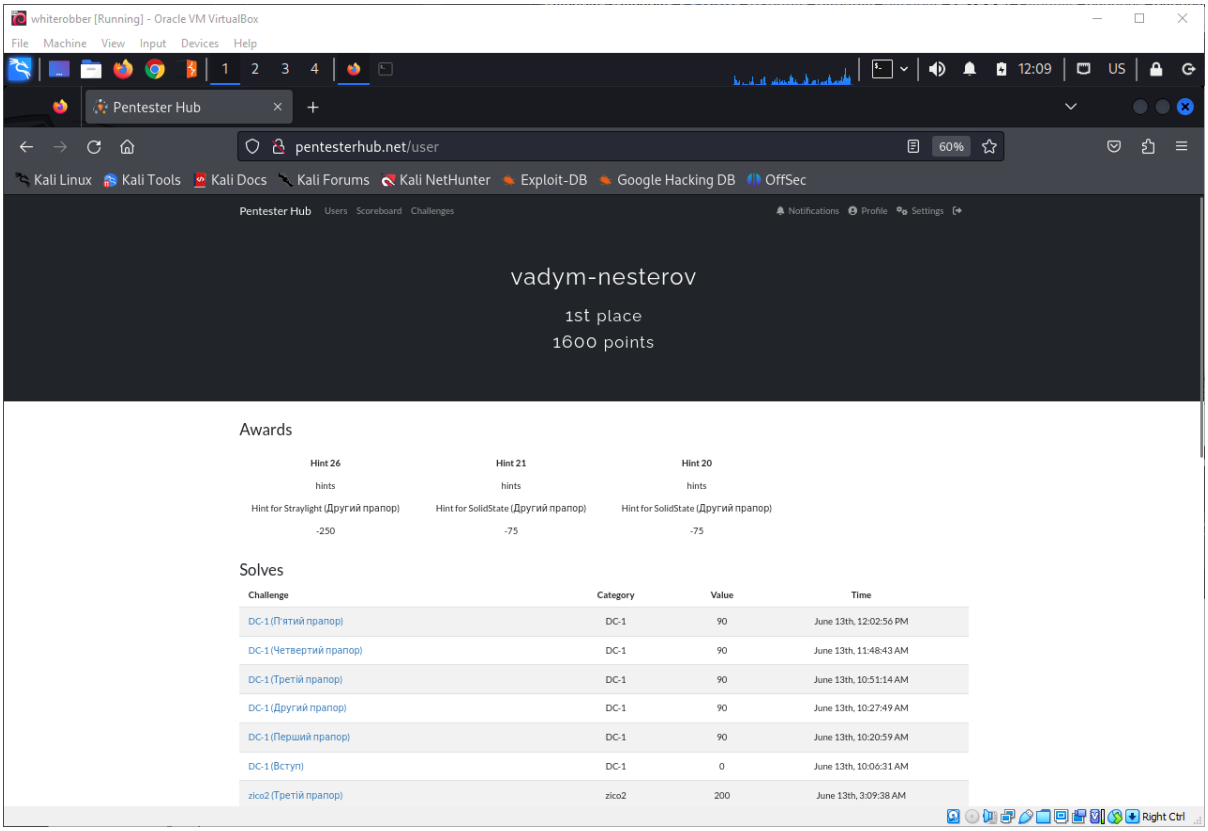


Рисунок 3.5.138 – Перегляд профілю користувача vadym-nesterov

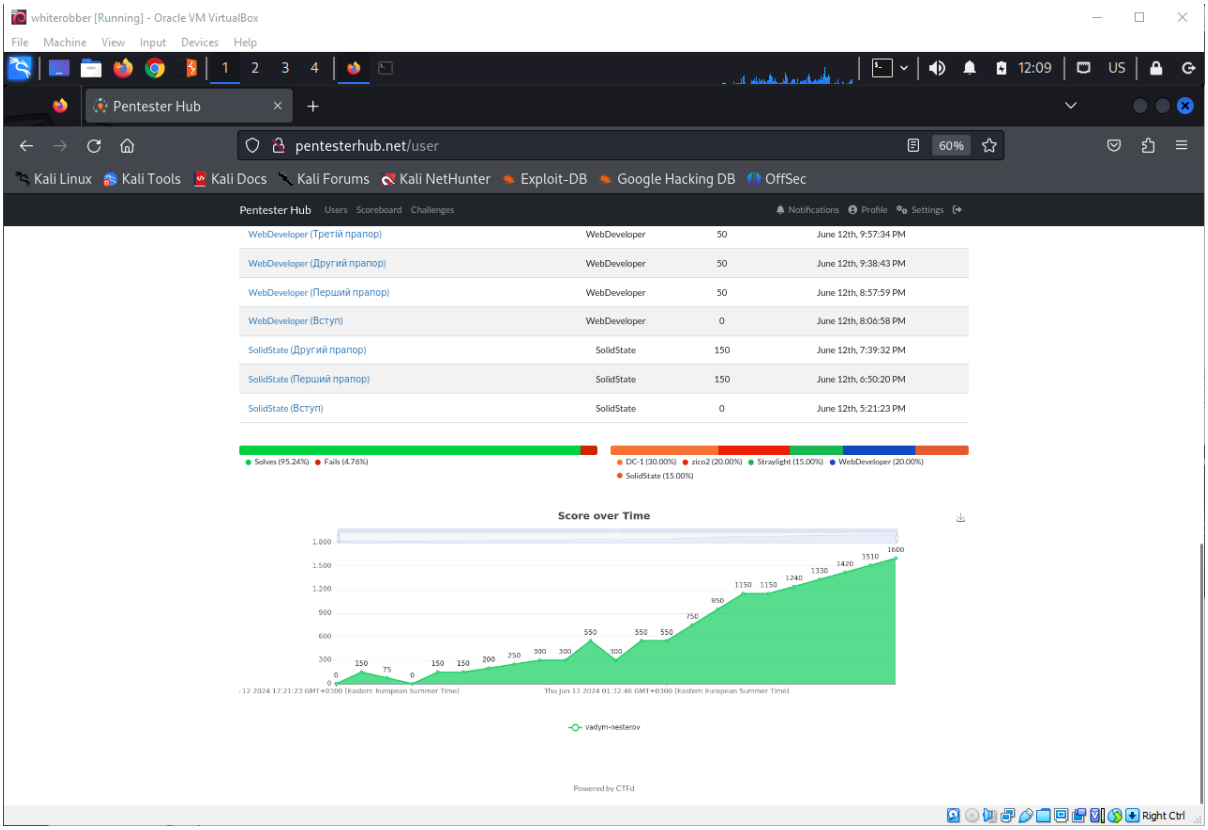


Рисунок 3.5.139 – Перегляд профілю користувача vadym-nesterov

Створимо ще два користувача та виконаємо декілька серій завдань, щоб перевірити працездатність таблиці розподілення балів.

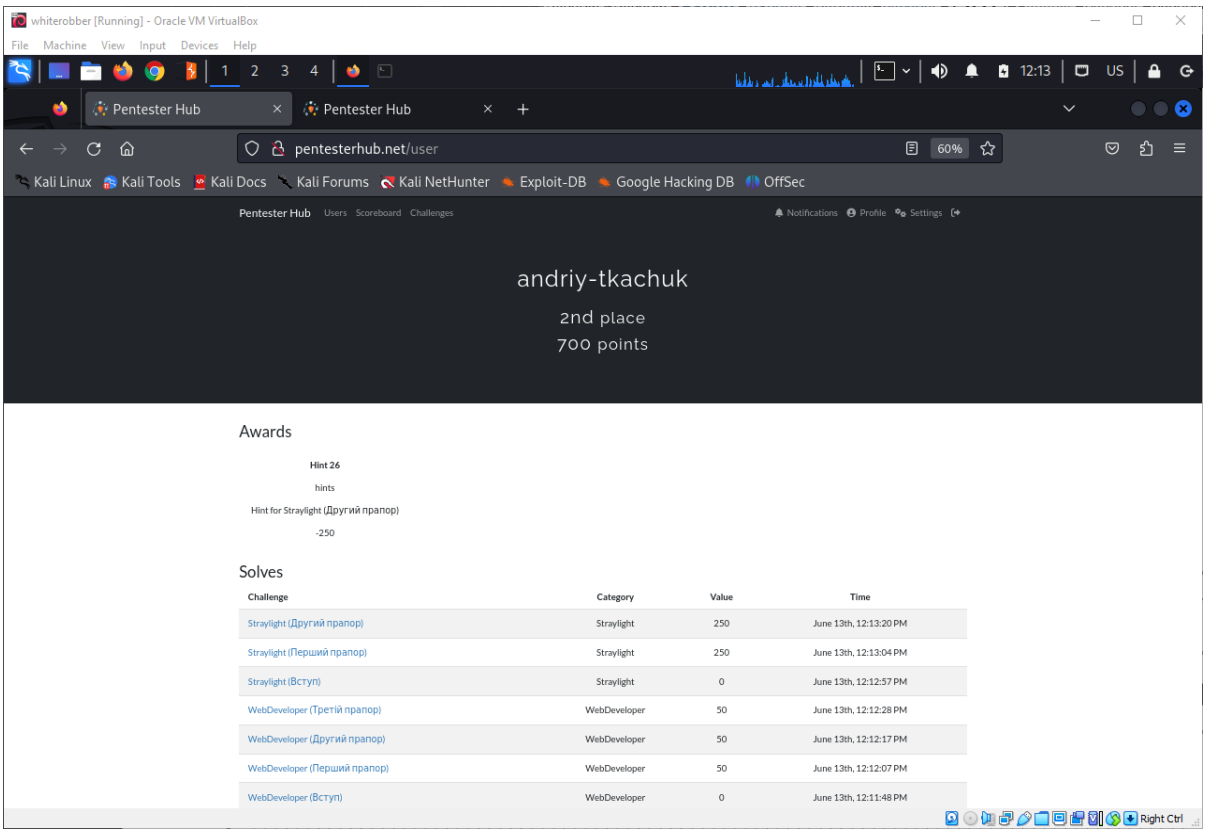


Рисунок 3.5.140 – Створено користувача andriy-tkachuk, перегляд профілю

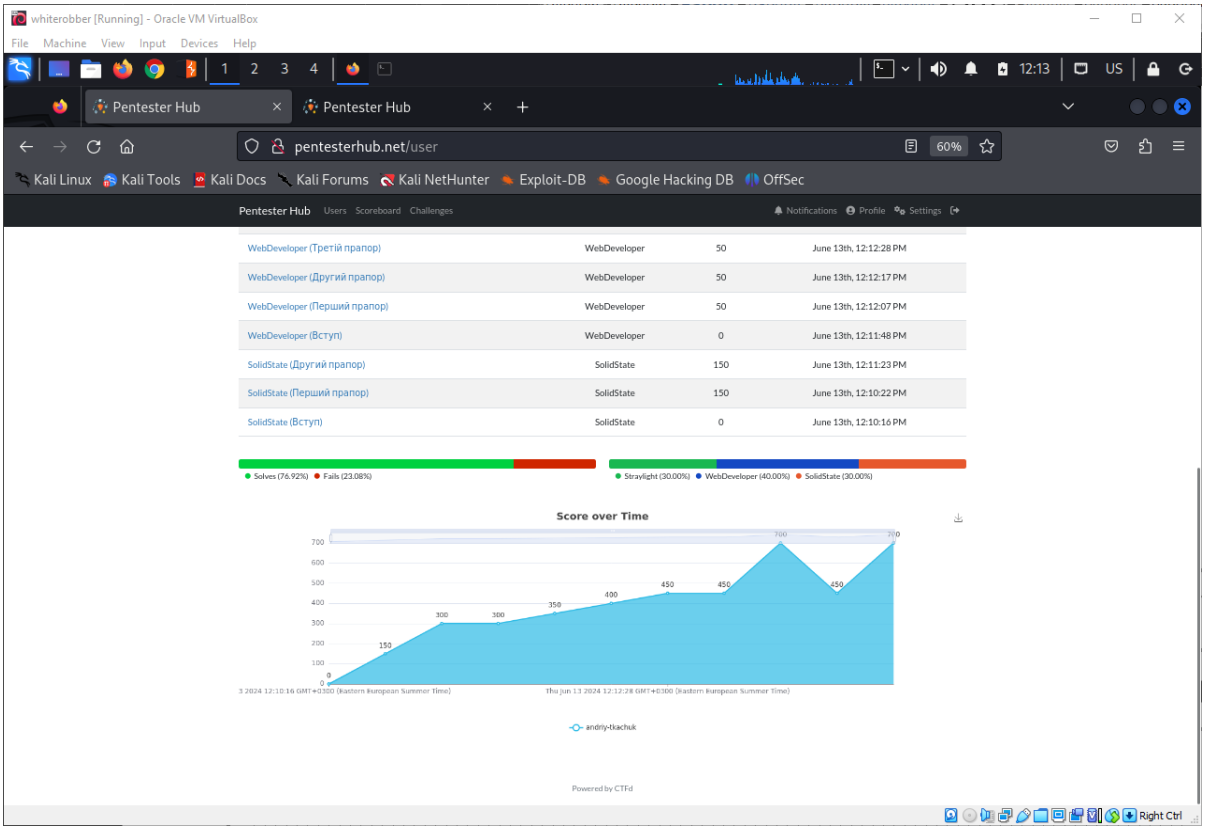


Рисунок 3.5.141 – Створено користувача andriy-tkachuk, перегляд профілю

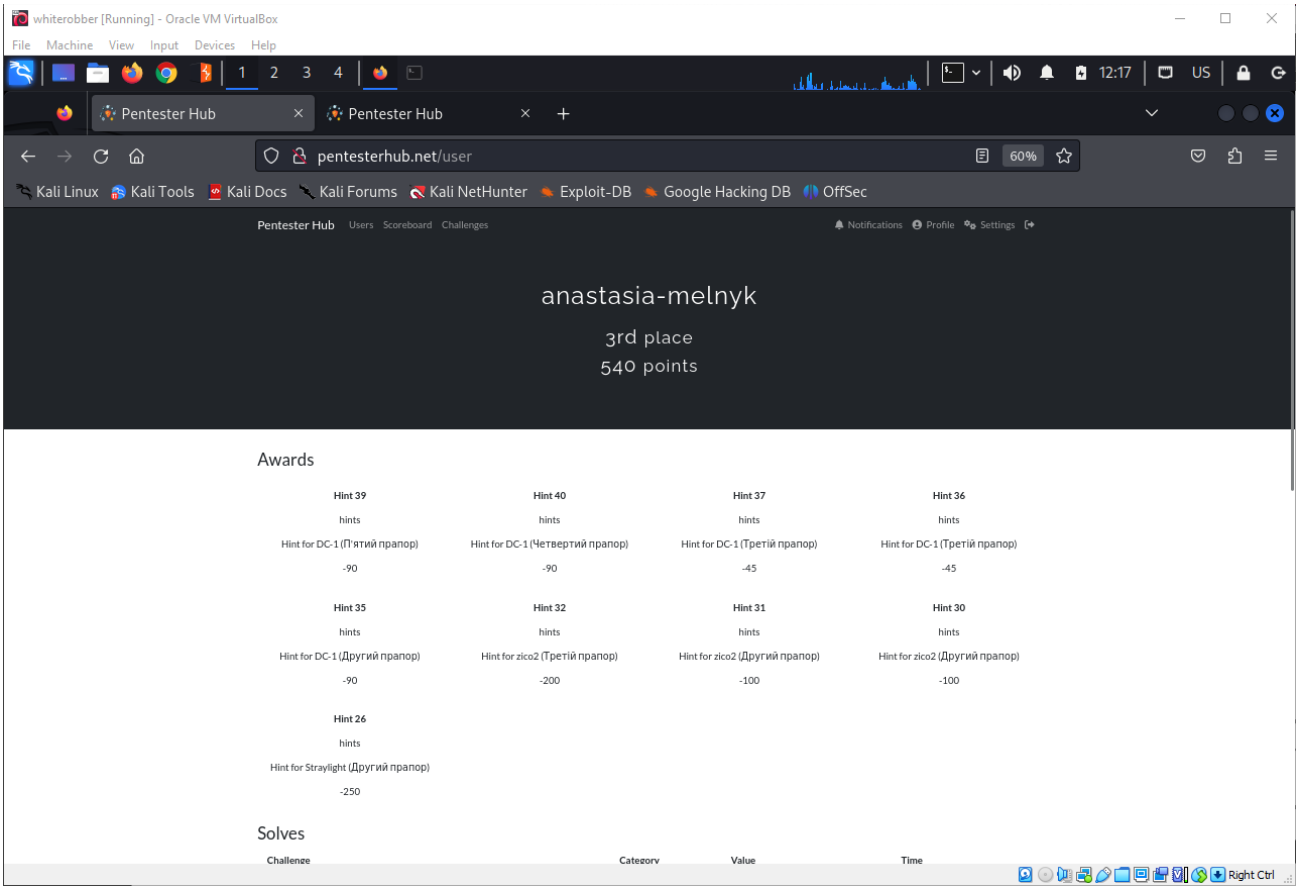


Рисунок 3.5.142 – Створено користувача anastasia-melnyk, перегляд профілю

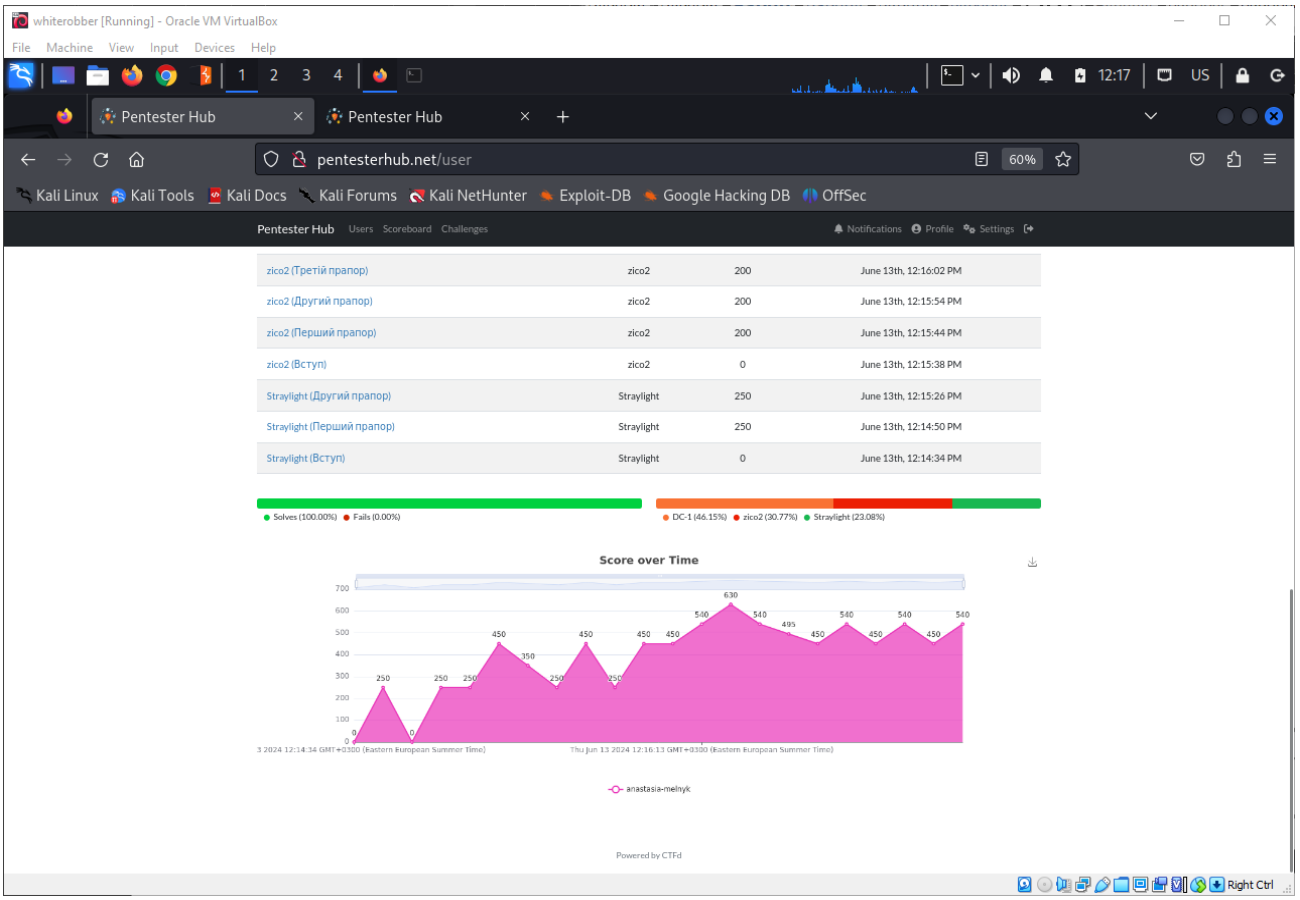


Рисунок 3.5.143 – Створено користувача anastasia-melnyk, перегляд профілю

Користувачі успішно створені, серії завдань виконані, переглянемо таблицю розподілення балів.

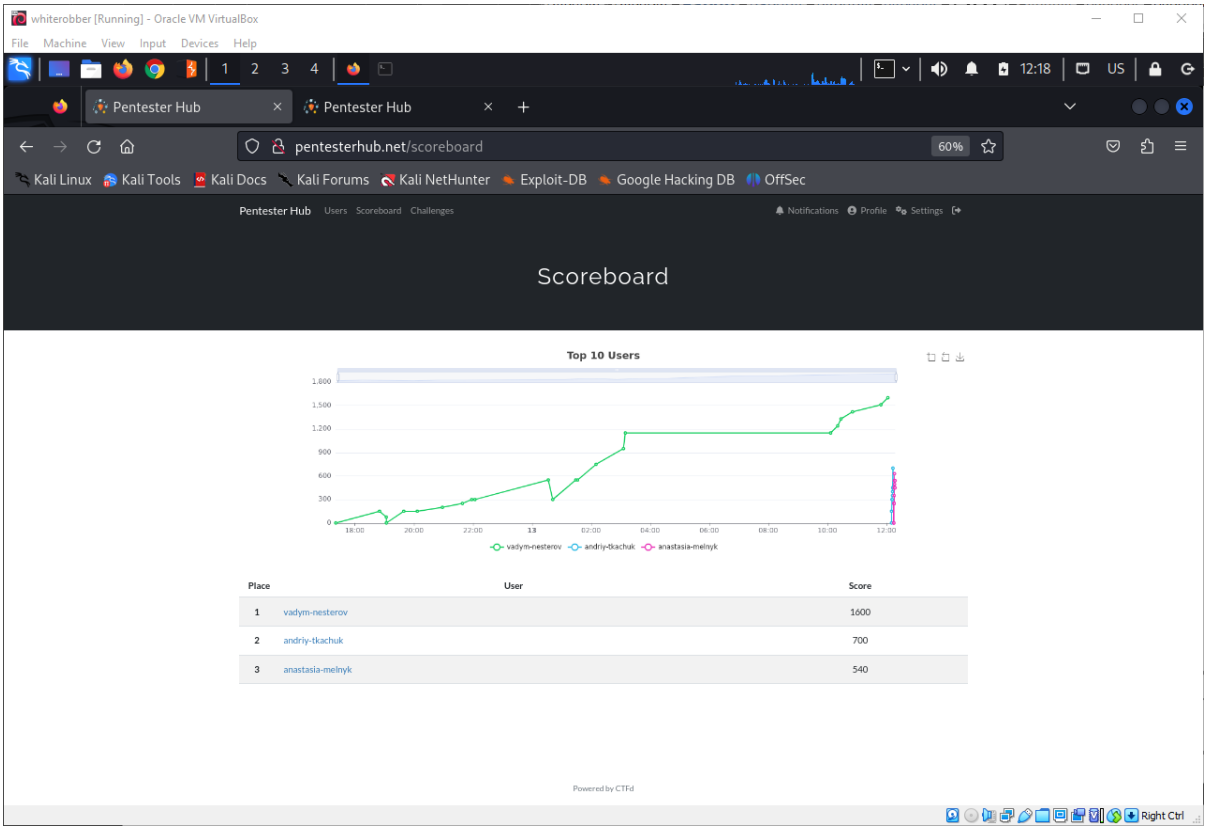


Рисунок 3.5.144 – Перегляд таблиці розподілення балів, все функціонує коректно

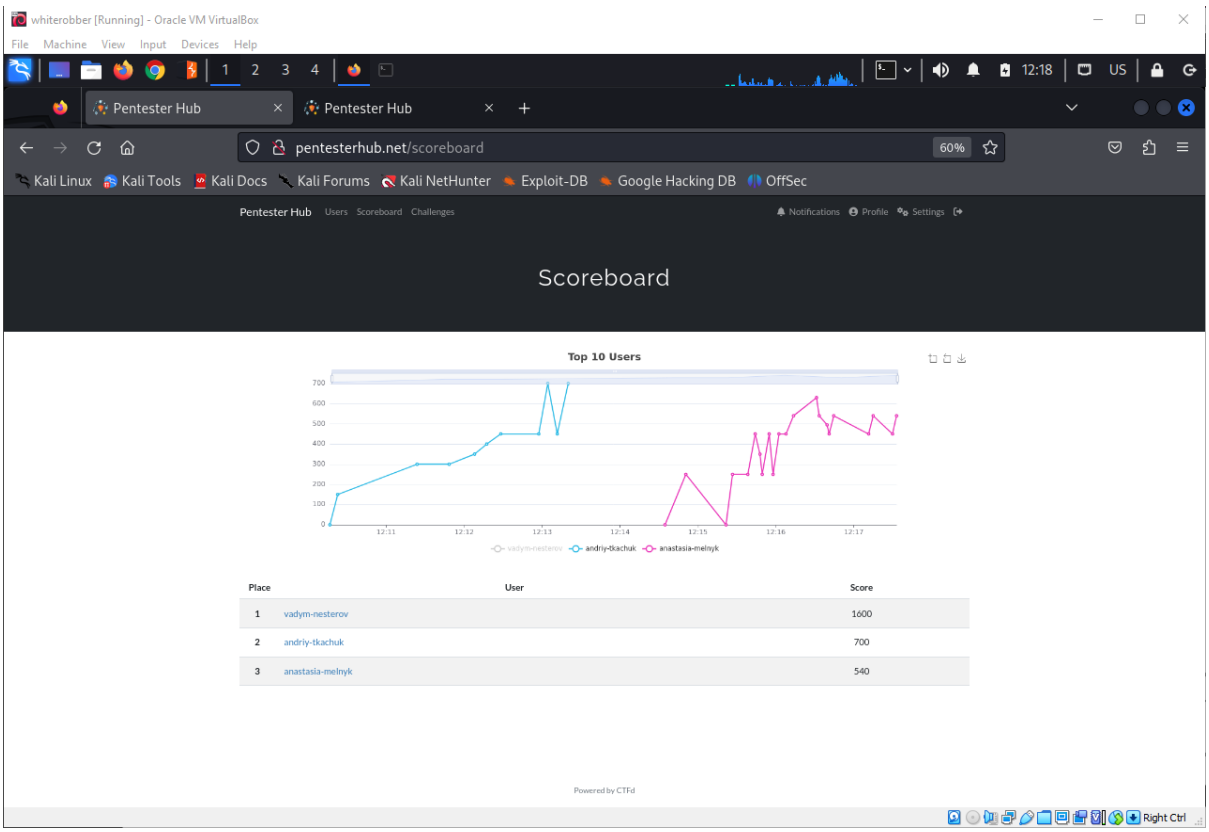


Рисунок 3.5.145 – Перегляд таблиці розподілення балів, все функціонує коректно

ВИСНОВКИ

В результаті виконання випускної кваліфікаційної роботи бакалавра було успішно досягнуто мети, розгортання інтерактивної платформи для проведення навчальних кібербезпекових досліджень за напрямом Penetration Testing у форматі CTF.

Детально розглянуто теоретичні аспекти розгортання платформ, Penetration Testing та формату CTF. Досліджено сервери та їх типи, включаючи фізичні, віртуальні та хмарні рішення. Розглянуто та проаналізовано гіпервізори, зокрема гіпервізори типу 1, гіпервізори типу 2 та гібридні гіпервізори. Досліджено поняття Penetration Testing, включаючи типи, обсяг, етапи та інструменти. Проаналізовано концепцію та типи CTF. Розглянуто наявні рішення для розгортання платформ у форматі CTF та ресурс готових VM-жерт VulnHub.

Після проведення детального огляду та аналізу теоретичних аспектів було спроєктовано мережеву інфраструктуру для розгортання платформи та розроблено схему адресації мережі. Було обрано відповідне мережеве обладнання, операційні системи, програмне забезпечення, VM-жертви та виконано розробку структури завдань під кожну VM-жертву.

На основі результатів проєктування було створено мережеву інфраструктуру, виконано налаштування мережевого обладнання, розгорнуто та налаштовано платформу на базі готового рішення CTFd, встановлено та налаштовано OpenVPN як VPN-сервера, наповнено платформу завданнями, зокрема створено серії завдань для кожної VM-жертви, після чого було проведено тестування роботи мережі та функціоналу платформи.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

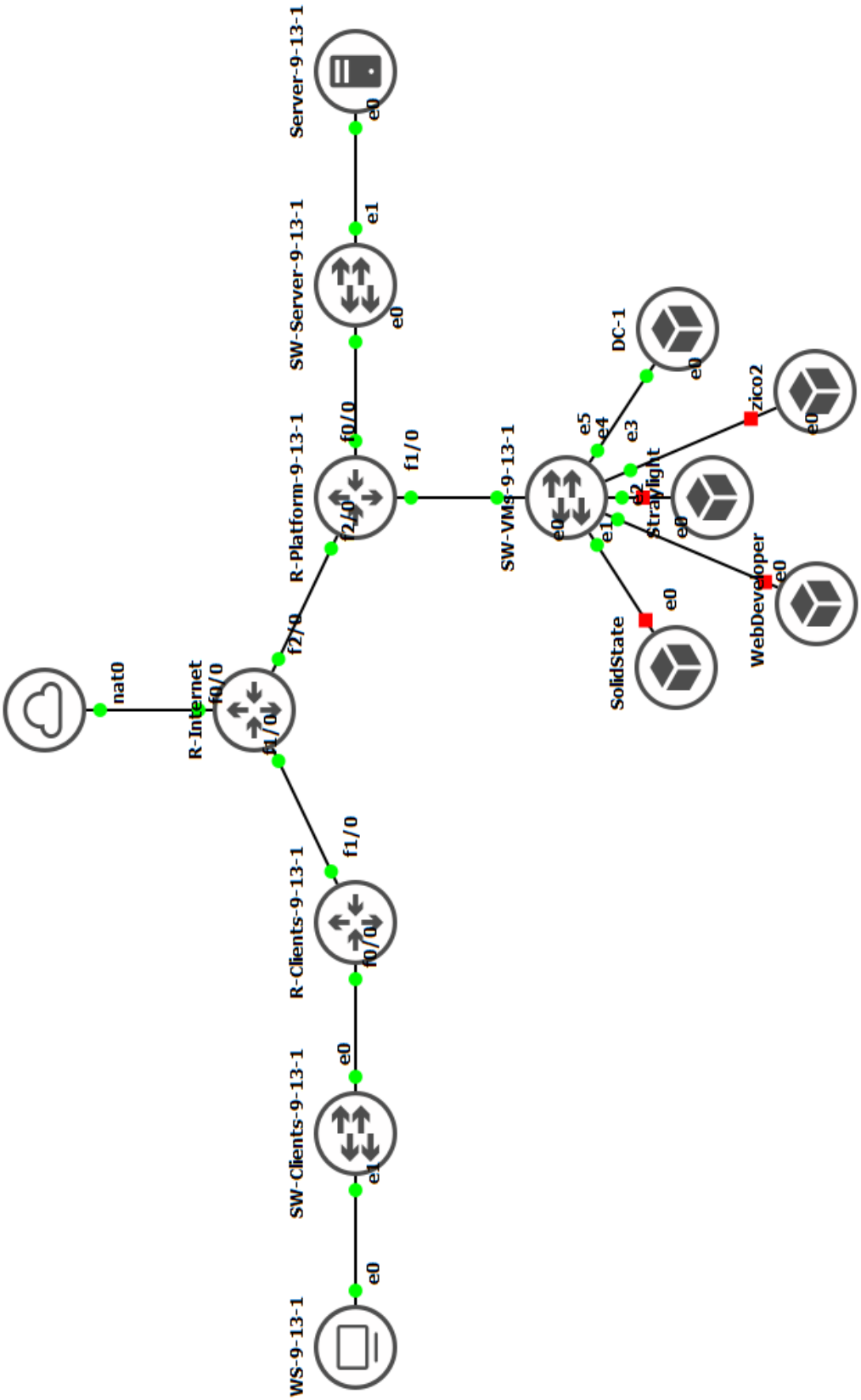
1. Baloch R. Ethical Hacking and Penetration Testing Guide. Auerbach Publishers, Incorporated, 2017.
2. Buchanan C. Kali Linux CTF Blueprints. Packt Publishing, Limited, 2014.
3. How To Guide: Set Up & Configure OpenVPN Client/server VPN. URL: https://openvpn.net/community-resources/how-to/.
4. Install the Compose plugin. URL: https://docs.docker.com/compose/install/linux/#install-using-the-repository.
5. Karamagi R. Penetration Testing. Independently Published, 2019.
6. Kim P. The Hacker Playbook 3: Practical Guide To Penetration Testing. Independently published, 2018. 289 p.
7. Sancheti S. CTF and its Types For Beginners !. URL: https://medium.com/@007ssancheti/ctf-and-its-types-for-beginners-aeb9904e9df.
8. Self-hosting a CTF Platform (CTFd). URL: https://medium.com/csictf/self-hosting-a-ctf-platform-ctfd-90f3f1611587.
9. Team N. Handbook for CTFers. Springer Singapore Pte. Limited, 2022.
10. Unveiling the World of Servers: What They Are and How They Work. URL: https://codeinstitute.net/global/blog/what-is-a-server/.
11. Weidman G. Penetration Testing: A Hands-On Introduction to Hacking. No Starch Press, 2014. 528 p.
12. What is a Cloud Server? | VMware Glossary. URL: https://www.vmware.com/topics/glossary/content/cloud-server.html.
13. What is a Hypervisor? | VMware Glossary. URL: https://www.vmware.com/content/vmware/vmware-published-sites/us/topics/glossary/content/hypervisor.html.html.
14. What is a hypervisor? The basics of virtualization technology. URL: https://www.sim-networks.com/en/blog/hypervisors-vmware-kvm-xen-openvz#containers.

15. What is a Hypervisor?. URL: https://www.pluralsight.com/blog/it-ops/what-is-hypervisor.
16. What is a Server?. URL: https://www.servers.com/news/blog/what-is-a-server.
17. What is a Virtual Server?. URL: https://www.cherryservers.com/blog/what-is-a-virtual-server-vps.
18. What is Kali Linux?. URL: https://www.kali.org/docs/introduction/what-is-kali-linux/.
19. What is Penetration Testing and How Does It Work? | Synopsys. URL: https://www.synopsys.com/glossary/what-is-penetration-testing.html#B.
20. What is Penetration Testing?. URL: https://www.ibm.com/topics/penetration-testing.
21. Wylie P. L., Crawley K. Pentester BluePrint: Starting a Career As an Ethical Hacker. Wiley & Sons, Incorporated, John, 2020. 192 p.
22. Керований комутатор Cisco WS-C2960+48TC-S. URL: https://e-server.com.ua/uk/komutator-kerovanii-cisco-ws-c296048tc-s.

ДОДАТКИ

Додаток А

Детальна схема побудованої мережі



Додаток Б

Лістинг конфігураційного файла R-Internet

```
R-Internet#show run
Building configuration...

Current configuration : 2485 bytes
!
upgrade fpd auto
version 12.4
service timestamps debug datetime msec
service timestamps log datetime msec
no service password-encryption
!
hostname R-Internet
!
boot-start-marker
boot-end-marker
!
logging message-counter syslog
!
no aaa new-model
ip source-route
no ip icmp rate-limit unreachable
ip cef
!
ip host pentesterhub.net 200.9.13.2
no ipv6 cef
!
multilink bundle-name authenticated
!
archive
 log config
  hidekeys
!
interface FastEthernet0/0
 ip address dhcp
 ip nat outside
 ip virtual-reassembly
 duplex half
!
interface FastEthernet1/0
 ip address 193.9.13.1 255.255.255.0
 ip verify unicast reverse-path
 ip nat inside
 ip virtual-reassembly
 rate-limit input access-group 101 64000 96000 96000
conform-action transmit exceed-action drop
 rate-limit output access-group 101 64000 96000
96000 conform-action transmit exceed-action drop
 duplex half
!
interface FastEthernet2/0
 ip address 194.9.13.1 255.255.255.0
 ip verify unicast reverse-path
 ip nat inside
 ip virtual-reassembly
 rate-limit input access-group 101 64000 96000 96000
conform-action transmit exceed-action drop
 rate-limit output access-group 101 64000 96000
96000 conform-action transmit exceed-action drop
 duplex half
!
interface FastEthernet3/0
 no ip address
 ip verify unicast reverse-path
 rate-limit input access-group 101 64000 96000 96000
conform-action transmit exceed-action drop
 rate-limit output access-group 101 64000 96000
96000 conform-action transmit exceed-action drop
 shutdown
 duplex half
!
!
interface FastEthernet4/0
 no ip address
 shutdown
 duplex half
!
interface FastEthernet5/0
 no ip address
 shutdown
 duplex half
!
interface FastEthernet6/0
 no ip address
 shutdown
 duplex half
!
router ospf 913
 log-adjacency-changes
 network 192.168.122.0 0.0.0.255 area 0
 network 193.9.13.0 0.0.0.255 area 0
 network 194.9.13.0 0.0.0.255 area 0
!
ip forward-protocol nd
no ip http server
no ip http secure-server
!
ip dns server
ip nat inside source list 1 interface FastEthernet0/0
overload
!
access-list 1 permit any
access-list 101 permit udp any any
access-list 101 permit icmp any any echo
access-list 101 permit icmp any any echo-reply
no cdp log mismatch duplex
!
control-plane
!
gatekeeper
 shutdown
!
!
line con 0
 exec-timeout 0 0
 privilege level 15
 logging synchronous
 stopbits 1
line aux 0
 exec-timeout 0 0
 privilege level 15
 logging synchronous
 stopbits 1
line vty 0 4
 login
!
end
```

Додаток В

Лістинг конфігураційного файла R-Clients-9-13-1

```
R-Clients-9-13-1#show run
Building configuration...

Current configuration : 2216 bytes
!
upgrade fpd auto
version 12.4
service timestamps debug datetime msec
service timestamps log datetime msec
no service password-encryption
!
hostname R-Clients-9-13-1
!
boot-start-marker
boot-end-marker
!
logging message-counter syslog
!
no aaa new-model
ip source-route
no ip icmp rate-limit unreachable
ip cef
!
ip dhcp excluded-address 195.9.13.1
!
ip dhcp pool Clients
   network 195.9.13.0 255.255.255.0
   default-router 195.9.13.1
   dns-server 192.168.122.23
!
ip name-server 192.168.122.23
no ipv6 cef
!
multilink bundle-name authenticated
!
archive
 log config
  hidekeys
!
interface FastEthernet0/0
 ip address 195.9.13.1 255.255.255.0
 ip verify unicast reverse-path
 rate-limit input access-group 101 64000 96000
96000 conform-action transmit exceed-action drop
 rate-limit output access-group 101 64000 96000
96000 conform-action transmit exceed-action drop
 duplex half
!
interface FastEthernet1/0
 ip address 193.9.13.2 255.255.255.0
 ip verify unicast reverse-path
 rate-limit input access-group 101 64000 96000
96000 conform-action transmit exceed-action drop
 rate-limit output access-group 101 64000 96000
96000 conform-action transmit exceed-action drop
 duplex half
!
interface FastEthernet2/0
 no ip address
shutdown
!
duplex half
!
interface FastEthernet3/0
 no ip address
 shutdown
 duplex half
!
interface FastEthernet4/0
 no ip address
 shutdown
 duplex half
!
interface FastEthernet5/0
 no ip address
 shutdown
 duplex half
!
interface FastEthernet6/0
 no ip address
 shutdown
 duplex half
!
router ospf 913
 log-adjacency-changes
 network 193.9.13.0 0.0.0.255 area 0
 network 195.9.13.0 0.0.0.255 area 0
!
ip forward-protocol nd
ip route 0.0.0.0 0.0.0.0 193.9.13.1
no ip http server
no ip http secure-server
!
ip dns server
!
access-list 101 permit udp any any
access-list 101 permit icmp any any echo
access-list 101 permit icmp any any echo-reply
no cdp log mismatch duplex
!
control-plane
!
gatekeeper
 shutdown
!
!
line con 0
 exec-timeout 0 0
 privilege level 15
 logging synchronous
 stopbits 1
line aux 0
 exec-timeout 0 0
 privilege level 15
 logging synchronous
 stopbits 1
line vty 0 4
 login
!
end
```

Лістинг конфігураційного файла R-Platform-9-13-1

```
R-Platform-9-13-1#show run
Building configuration...

Current configuration : 2710 bytes
!
upgrade fpd auto
version 12.4
service timestamps debug datetime msec
service timestamps log datetime msec
no service password-encryption
!
hostname R-Platform-9-13-1
!
boot-start-marker
boot-end-marker
!
logging message-counter syslog
!
no aaa new-model
ip source-route
no ip icmp rate-limit unreachable
ip cef
!
ip dhcp excluded-address 10.9.13.1
!
ip dhcp pool VMs
   network 10.9.13.0 255.255.255.0
   default-router 10.9.13.1
!
ip dhcp pool SolidState
   host 10.9.13.2 255.255.255.0
   hardware-address 0800.2729.fb2b
!
ip dhcp pool Straylight
   host 10.9.13.4 255.255.255.0
   hardware-address 0800.275a.74f9
!
ip dhcp pool zico2
   host 10.9.13.5 255.255.255.0
   hardware-address 0800.271d.c74f
!
ip dhcp pool DC-1
   host 10.9.13.6 255.255.255.0
   hardware-address 0800.27c7.1411
!
ip dhcp pool WebDeveloper
   host 10.9.13.3 255.255.255.0
   client-identifier
ffe2.343f.3e00.0200.00ab.11fb.b435.40f9.e8db.fa
!
ip name-server 192.168.122.23
no ipv6 cef
!
multilink bundle-name authenticated
!
archive
 log config
  hidekeys
!
interface FastEthernet0/0
 ip address 200.9.13.1 255.255.255.0
 ip verify unicast reverse-path
 rate-limit input access-group 101 64000 96000 96000
conform-action transmit exceed-action drop
 rate-limit output access-group 101 64000 96000
96000 conform-action transmit exceed-action drop
 duplex half
!
interface FastEthernet1/0
 ip address 10.9.13.1 255.255.255.0
 duplex half
!
interface FastEthernet2/0
 ip address 194.9.13.2 255.255.255.0
 ip verify unicast reverse-path
 rate-limit input access-group 101 64000 96000 96000
conform-action transmit exceed-action drop
 rate-limit output access-group 101 64000 96000
96000 conform-action transmit exceed-action drop
 duplex half
!
interface FastEthernet3/0
 no ip address
 shutdown
 duplex half
!
interface FastEthernet4/0
 no ip address
 shutdown
 duplex half
!
router ospf 913
 log-adjacency-changes
 network 194.9.13.0 0.0.0.255 area 0
 network 200.9.13.0 0.0.0.255 area 0
!
ip forward-protocol nd
ip route 0.0.0.0 0.0.0.0 194.9.13.1
ip route 10.8.0.0 255.255.255.0 200.9.13.2
no ip http server
no ip http secure-server
!
!
!
access-list 101 permit udp any any
access-list 101 permit icmp any any echo
access-list 101 permit icmp any any echo-reply
no cdp log mismatch duplex
!
control-plane
!
gatekeeper
 shutdown
!
line con 0
 exec-timeout 0 0
 privilege level 15
 logging synchronous
 stopbits 1
line aux 0
 exec-timeout 0 0
 privilege level 15
 logging synchronous
 stopbits 1
line vty 0 4
 login
!
end
```