Міністерство освіти і науки України
Національний технічний університет
«Дніпровська політехніка»

Інститут електроенергетики
Факультет інформаційних технологій
Кафедра безпеки інформації та телекомунікацій

Пояснювальна записка

Кваліфікаційної роботи ступеню бакалавра

Студента Мірошника Артема Юрійовича

Академічної групи 125-19-1

Спеціальності 125 кібербезпека

Спеціалізації

За освітньо-професійною програмою Кібербезпека

На тему Методи тестування вразливостей в системах автентифікації

Керівники	Прізвище, ініціали	Оцінка за шкалою		Підпис
		рейтинговою	інституційною	
кваліфікаційної роботи	д.т.н., проф. Корнієнко В.І.			
розділів:				
спеціальний	ст. викл. Тимофєєв Д. С			
економічний	доц., к.е.н. Пілова Д. П.	90	відмінно	

Рецензент				

Нормконтролер	проф. Гусєв О.Ю.			

Дніпро
2023

ЗАТВЕРДЖЕНО:
завідувач кафедри
безпеки інформації та комунікацій
_____д.т.н., проф. Корнієнко В.І
«_____»_____2023 року

ЗАВДАННЯ
на кваліфікаційну роботу
ступеню бакалавра

студенту Мірошнику А.Ю. **академічної групи** 125-19-1

спеціальності 125 Кібербезпека

спеціалізації_____

за освітньо-професійною програмою Кібербезпека

_____

на тему Методи тестування вразливостей в системах автентифікації

_____,

затверджену наказом ректора НТУ «Дніпровська політехніка» від 16.05.2023 № 350-с

Розділ	Зміст	Термін виконання
Розділ 1	Проаналізувати методи тестування систем автентифікації, розглянути типові властивості систем автентифікації	15.05.2023
Розділ 2	Практично протестувати системи автентифікації на базі типового об'єкту. Розробити покрокову інструкцію тестування.	30.05.2023
Розділ 3	Розглянути питання чи є проектування ефективної системи автентифікації для протидії типовим вразливостям доцільним.	05.06.2023

Завдання видано _____ _____
(підпис керівника) (прізвище, ініціали)

Дата видачі 01.05.2023

Дата подання до екзаменаційної комісії 13.05.2023

Прийнято до виконання _____ Мірошник А.Ю.
(підпис студента) (прізвище ініціали)

Реферат

Пояснювальна записка: 95 с., 58 рис., 1 табл., 6 додатків, 17 джерел.

Мета роботи: Проаналізувати типові вразливості систем автентифікації веб-додатків, розробити інструкції тестування цих систем на прикладі типового об'єкту для підвищення рівня захищеності систем автентифікації.

Об'єкт дослідження: Навчальний портал Portswigger та його навчальне середовище.

Предмет дослідження: Типові вразливості систем автентифікації веб-додатків.

В першому розділі було проаналізовано типові вразливості систем автентифікації веб-додатків, оглянуто типовий об'єкт на базі якого будуть розглядатися системи автентифікації. Досліджено методи захисту можливі методи захисту систем автентифікації. Була поставлена задача на спеціальний розділ кваліфікаційної роботи.

В спеціальному розділі практично показано використання типових вразливостей систем автентифікації веб-додатків на прикладі навчального порталу Portswigger, та розроблені загальні покрокові інструкції тестування різних систем автентифікації та надані способи підвищення рівня захисту систем автентифікації.

В економічному розділі розглянуто питання доцільності проектування ефективної системи автентифікації. Були розраховані капітальні витрати на проектування та річні витрати на підтримку систем автентифікації.

Практичне значення роботи полягає у зміцненні рівня захисту систем автентифікації методом їх тестування.

Результати здійснених у кваліфікаційній роботі досліджень можуть бути використані навчальних цілей, та зміцнення рівня захисту систем автентифікації.

КЛЮЧОВІ СЛОВА: ВРАЗЛИВОСТІ. СИСТЕМИ АВТЕНТИФІКАЦІЇ. АВТЕНТИФІКАЦІЯ. ТЕСТ НА ПРОНИКНЕННЯ. ВРАЗЛИВОСТІ.

Abstract

Explanatory note: 95 pages, 58 figures, 1 table, 6 appendices, 17 sources

Purpose: Analyze typical vulnerabilities in the authentication systems of web-apps. Develop instructions how these authentication systems can be tested on the Portswigger educational enironment example for increasing the authentication systems level of security.

Object of the research: Educational portal Portswigger and its educational enironment.

Subject: Typical vulnerabilities of authentication systems.

In the first part typical vulnerabilities of the authentication systems of web-apps were analyzed, reviewed the typical object on which example vulnerabilities were investigated. The methods of protecting authentication systems were researched. The task was set in this part for the special part of this qualification work.

In the special part the authentication systems were inspecred in practice on the Portswigger educational environment example. Were developed general step-by-step instructions for authentication systems testing. The recommendationas were provided for increasing the security level of the authentication systems.

In the third part the question relevance of creating efficient and protected authentication system was reviewed. The capital and annual expenditure were calculated and based on these calculations, the conlusion was suggested.

The practical significance of the work is to develop recommendations for increasing level of securiy of web-apps authentications systems.

KEY WORDS: VULNERABILITIES. AUTHENTICATION SYSTEMS. AUTHENTICATION. PENETRATION TEST. VULNERABILITIES.

ЗМІСТ

Вступ

Мета роботи: Проаналізувати типові вразливості систем автентифікації веб-додатків, розробити інструкції тестування цих систем на прикладі типового об’єкту для підвищення рівня захищеності систем автентифікації.

Об’єкт дослідження: Навчальний портал Portswigger та його навчальне середовище.

Предмет дослідження: Типові вразливості систем автентифікації веб-додатків.

В першому розділі було проаналізовано типові вразливості систем автентифікації веб-додатків, оглянуто типовий об’єкт на базі якого будуть розглядатися системи автентифікації. Досліджено методи захисту систем автентифікації. Була поставлена задача на спеціальний розділ кваліфікаційної роботи.

В спеціальному розділі практично показано використання типових вразливостей систем автентифікації веб-додатків на прикладі навчального порталу Portswigger, та розроблені загальні покрокові інструкції тестування різних систем автентифікації та надані рекомендації підвищення рівня захисту систем автентифікації.

В економічному розділі розглянуто питання доцільності проектування ефективної системи автентифікації. Були розраховані капітальні витрати на проектування та річні витрати на підтримку систем автентифікації.

Розділ 1 Стан питання. Постановка задачі.

1.1 Стан питання

Так як системи автентифікації є важливою частиною функціювання сайтів, завжди буде актуальне питання захищеності даних клієнтів та співробітників, тому однією с основних тем цієї роботи. Після пандемії COVID-19 гостро постало питання як продовжувати вести бізнес, якщо правила ізоляцію не дозволяли масове скупчення людей. Багато хто почав переходити до до створення своїх онлайн магазинів або веб-додатку. На рис 1.1 можна побачити як зросли об'єми онлайн торгівлі в Україні з 2016 до 2023 року.

1.1 Динаміка ринку електронної комерції

З 2019 року до 2023 року об'єм електронної комерції збільшився в 2 рази, завдяки розвитку технологій, та потреби ізоляції людей одне від одного за часів пандемії COVID-19. На рис 1.2 можна побачити динаміку кількості людей, які використовують онлайн магазини:

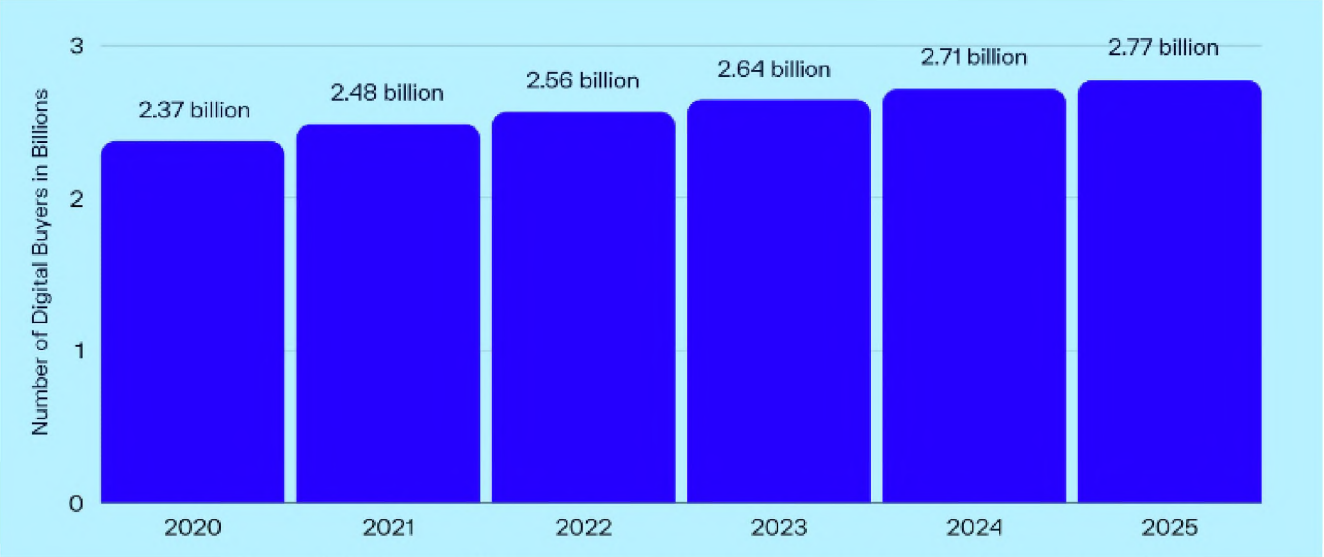


Рис. 1.2 Динаміка використання людьми онлайн магазинів.

З рис. 1.1 та рис. 1.2 можна зробити висновок, що онлайн комерція стає все більше популярною та майже третина людства замовляє будь-які товари саме використовуючи онлайн магазини. Багато магазинів та будь-яких інших веб-додатків вимагають мати користувача особистий акаунт з ними для того, щоби використовувати повний функціонал сайту (замовлення товарів, викладання постів у блозі та ін.). Веб-додатки для авторизації користувачів використовують системи автентифікації. В акаунтах часто зберігається конфеденційна інформація (наприклад як дані кредитної карти, або фізична адреса користувача). Тому актуальність створення ефективної системи автентифікації це питання, яке набирає позитивної динаміки.

На тему захисту та проектування ефективних автентифікаційних систем виходили наступні роботи «*An Efficient Multifactor Authentication System*»[1] від авторів Shreya Verma та Kittika Charturvedi написана у 2023 році, «*Efficient Authentication Mechanism for Defending Against Reflection-Based Attacks on Domain Name System*»[2], автор Rebeen Hama Amin 2020 року, та «*Security Authentication Mechanism of Spatio-Temporal Big Data Based on Blockchain*»[3], авторів Bao Zhou, Junsan Zhao 2023 року. Проте вони розповідають про системи автентифікації у базах даних, атаки на системи автентифікації на системи домених імен, або

проектування багатофакторної системи автентифікації використовуючи нестандартні алгоритм TWINE та використання картинки зашифрованої з використанням GENETIC алгоритму. Проте методи для проектування різних систем автентифікації для різних об’єктів, на яких вона буде використовуватися, може повністю відрізнятися одне від одного, хоча деякі аспекти можуть здаватися схожими. Актуальність даної кваліфікаційної роботи полягає у аналізі типових вразливостей систем автентифікації у веб-додатках на базі типового об’єкту, їх практичного тестування та на базі отриманих результатів розробити методи тестування систем автентифікації та рекомендації для підвищення рівня захищеності систем автентифікації у веб-додатках.

1.2 Аналіз механізмів роботи систем автентифікації.

Систем автентифікації мають власні механізми роботи в залежності від виду .На рис. 1.3 зображено схему механізму роботи системи однофакторної автентифікації

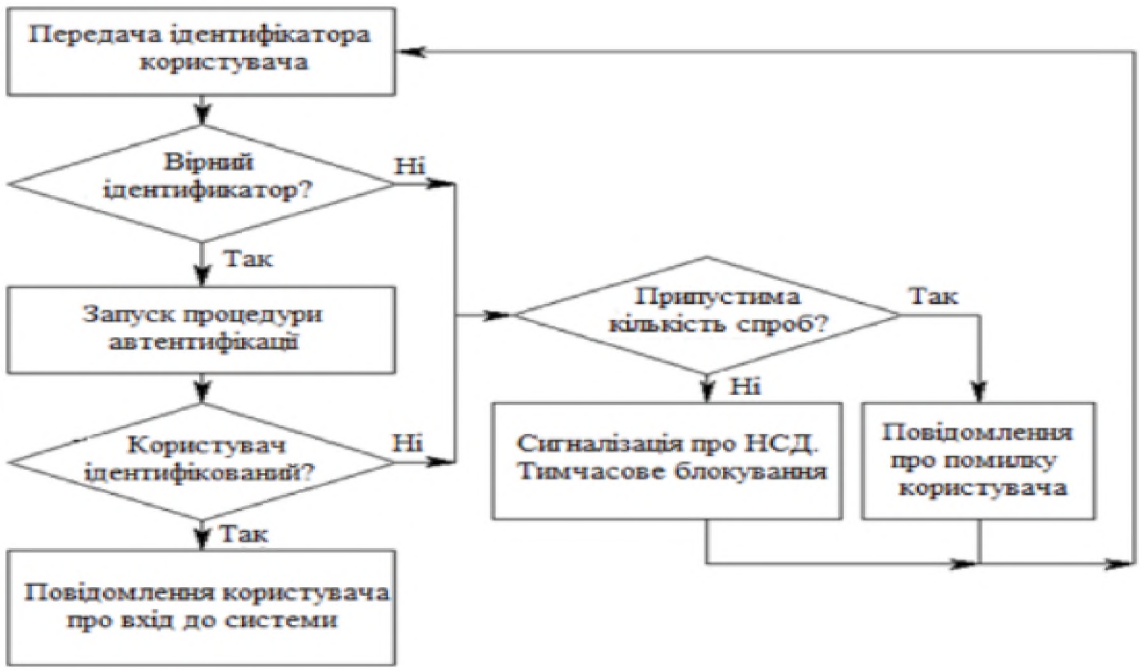


Рисунок 1.3 Схема однофакторної автентифікації

Автентифікація[4] це процедура верифікації належності ідентифікатора суб'єкту. Слід пам’ятати, шо автентифікація, це один із процесів для повної авторизації. Спочатку йду Ідентифікація (процес перевірки наявності облікового запису) за нею процедура автентифікації та в кінці авторизація (перевірка які рівні доступу та повноважень має користувач). Усі процеси є одним цілим для процедури повної автентифікації.

Відповідно до рис 1.3 в однофакторній автентифікації, користувач спочатку передає свій ідентифікатор (логін та пароль зазвичай передаються одночасно), система автентифікації перевіряє за логіном чи присутній такий логін в існуючій базі даних, у випадку, якщо було надано недійсний ідентифікатор процедура повної авторизації припиняється і система видає користувачу помилку, зазвичай с типовою помилкою «Invalid username». Якщо ідентифікатор (логін) було вказано вірно, розпочнеться процедура автентифікації (перевірка чи відноситься вказаний пароль до вказаного логіну), у разі пароль не співпадає, користувач може побачити іншу типову помилку: «Incorrect password», та процедура авторизації знову припиняється, та у разі якщо всі дані співпали, система авторизує користувача за наданими даними, та переадресовує до особистого кабінету.

Однофакторні системи автентифікації стають все менш популярними через те, що є ефективніші системи автентифікації, які використовують декілька факторів для автентифікації користувача. Такі системи ще називають багатофакторними.

На рис 1.4 представлено механізм роботи багатофакторної системи автентифікації.

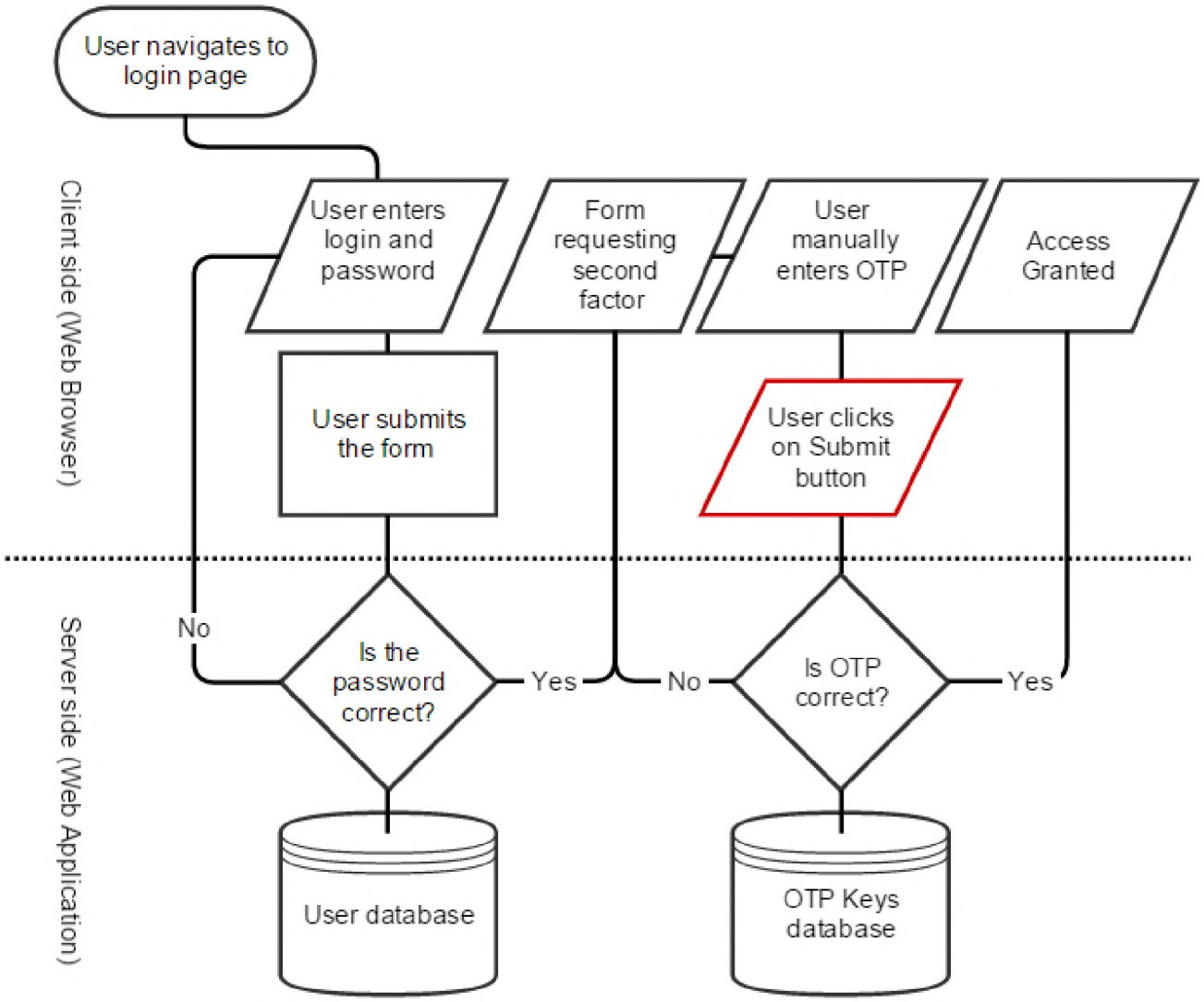


Рис. 1.4 Схема роботи багатофакторної автентифікації

Механізм роботи автентифікації в багатофакторній системі автентифікації схожі до механізму роботи системи однофакторної автентифікації за одним виключенням, що після комбінації логіну та паролю у користувача запитується додаткова інформація для перевірки ще одного фактору інформації. Це може бути як верифікаційний код відправлений до автентифікаційного додатку такого, як «Google authenticator», або носій з завантаженим до нього автентифікаційним ключем.

Види факторів автентифікації можна розподілити на три групи за факторами які система автентифікації перевіряє:

- Дещо, що вам відомо, або те, що ви знаєте, наприклад як пароль або відповідь на якесь секретне питання іноді називають цей фактор як “ фактор знання “
- Дещо, що ви маєте, це може бути фізичний об’єкт, мобільний телефон, захисний токен чи верифікаційний PIN код, який приходить до якогось автентифікаційного додатка. Це фактор іноді називають «Фактор володіння»
- Дещо, чим ви є, біометрика, також сюди може відноситися шаблони поведінки. Іноді називають цей фактор як «Фактор належення»

При проектуванні систем автентифікації, та методів багатофакторної автентифікації треба пам’ятати, що максимальну ефективність від багатофакторної системи автентифікації можна отримати тільки у тому разі, якщо вона перевіряє різні фактори. Наприклад, якщо під час стандартної автентифікації системи перевіряє комбінацію логіна та паролю на дійсність, то відправляти верифікаційний код до електронної пошти немає сенсу, так як ми перевіримо один і той самий фактор знань. Якщо буде використаний метод багатофакторної автентифікації через отримання коду в автентифікаційному додатку, або через носій з завантаженим ключем, будуть перевірені два різних фактор, де комбінація паролю та логіну це буде фактор знань, а верифікаційний код з додатку буде фактором володіння і в такому разі система автентифікації буде ефективною. Автентифікація за допомогою носія с ключем, також цей спосіб автентифікації називають U2F (Universal 2nd factor) наведено на рис 1.5

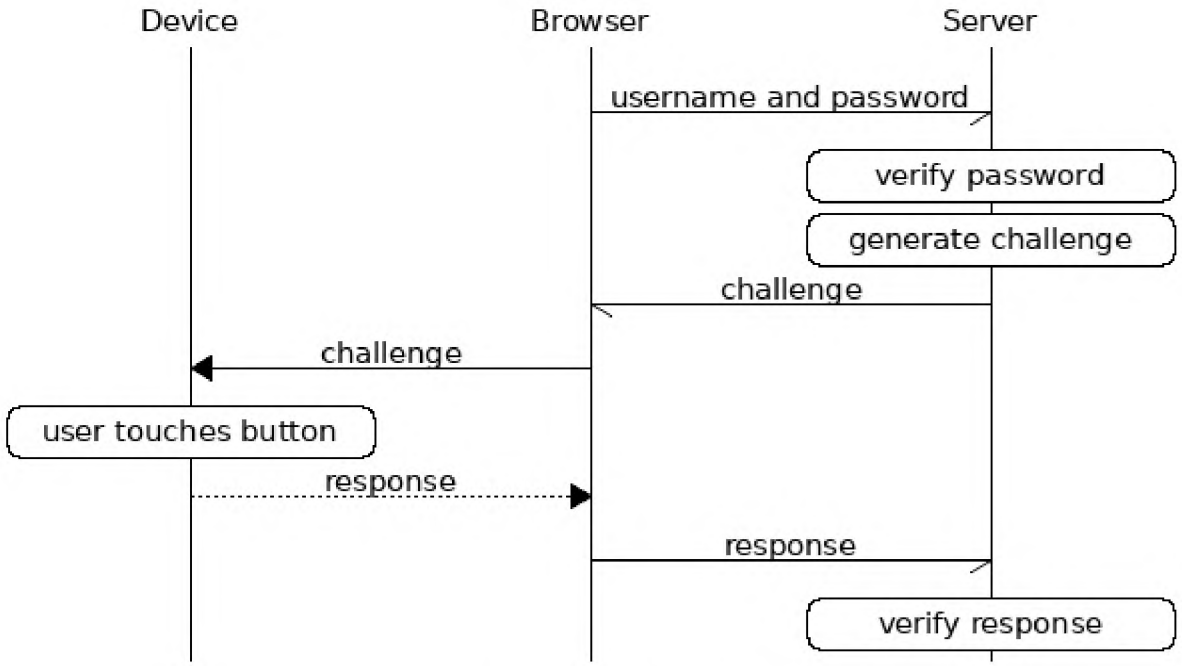


Рис 1.5 Схема роботи U2F

Отже, згідно до рис 1.5 механізм роботи U2F наступний: при реєстрації ключа для носія, користувач отримає відкритий ідентифікаційний ключ від веб-додатку на якому ця опція вмикається, у той же час генерується дескриптор цього ж ключа для веб-додатку для подальшої ідентифікації ключа. Під час автентифікації, коли запит на вхід до акаунту відправляється до серверу, сервер відправляє до носія дескриптор ключа через браузер, в свою чергу, носій передає до сервера через браузер відкритий ключ, якщо відкритий ключ підходить для дескриптора, вхід буде вдалий. Тобто, верифікаційний код на носії можна вважати ще одним ідентифікатором.

В спеціальній частині даної кваліфікаційної роботи було розглянено обидва види систем автентифікації (однофакторна та багатофакторна).

1.3 Аналіз типового об’єкту та середовища тестування

Так як намагатися провести атаку на якийсь реальний веб-додаток це незаконно, для цієї кваліфікаційної роботи буде використано навчальне середовище на базі веб-додатку Portswigger[1].

Portswigger платформа, яка надає теоретичні знання з різних відгалужень тесту на проникнення (або Pentest), там представлені не тільки загрози або вразливості саме систем автентифікації, а також SQL ін'єкцій, Вразливості DOM-дерева, Cross-site scripting і т.д.

Ці теми можна обирати для вивчення у довільному порядку. Перед усіма темами, Portswigger дає загальне представлення с приводу того, що таке тест на проникнення, які навички потрібні для здійснення атак, та які наслідки можуть бути, якщо атака виявилася вдалою. Якщо людина ніколи раніше не займалася тестом на проникнення, та загалом не дуже знає про структуру веб-додатків там веб-технологій, ця платформа також надає потрібний мінімум знань для того щоби людина яка вчиться могла проходити лабораторні роботи з майже повним розумінням усього, що вона робить.

Перед початком кожної теми, завжди надана теоретична частина, де можна зрозуміти як та чи інша атака може бути реалізована, які дані для цього потрібні, або як ті дані можна отримати протягом атак на навчальному полігоні (у навчальному середовищі представленому Portswigger). Також пояснюється чим страшні ці вразливості, та який збиток вони можуть принести компанії, або власнику веб-додатку, у разі вдалою атаки.

Як вже було сказано раніше, лабораторні роботи можна обирати в довільному порядку, тому кожен можна обрати для себе саме те відгалуження тесту на проникнення, яке подобається найбільше, або те в я кому більше досвіду. Ті лабораторні роботи в цілому мають інтерес як і для тих хто ще жодного разу не пробував себе у ролі тестера на проникнення, так і для майстрів своєї справи.

Зі структурі надання лабораторних робіт можна відмітити, що вони не закинуті до навчального курсу за випадковим чином. По-перше, роботи розподілені за складністю, та за порядком тем, які були вивчені. Деякі

лабораторні роботи для виконання потребують навички, які можна практично здобути лише у минулій роботі, тому іноді буває, що теми лабораторних робіт скачуть з брут-форсу до енумерації логінів, а згодом знову повертаються до брут-форсу тому, що необхідні знання були отримані під час виконання лабораторної роботи по енумерації логінів. Це надає можливості не застрягнути на лабораторній роботі через незнання, що робити далі, і не змушує шукати ті роботи, які можна бути виконати за допомогою навичок вже отриманих. Отже, навчальні матеріали та курси на Portswigger мають хорошу структуру та планування для навчання.

Сама структура знань на сайті розподілена на 3 великі групи це “Server-side topircs” - вразливості на стороні серверу, «Client-side topics» - вразливості зі сторони клієнта та «Advanced topics» - теми просунутого рівня. При цьому самі лабораторні роботи також розподілені на три групи за складністю та потрібними навичками для виконання, ці рівні мають наступні назви: «Apprentice», “ Practioner” та «Expert»

Перші два рівні складності лабораторних робіт не несуть ніякого інтересу для експертів, проте останній рівень вимагає вже креативного мислення та знання своєї справи.

Portswigget також надає доступ до програми Burp Suite, яка буде неодноразово використана під час цієї кваліфікаційної роботи. При виконані усіх лабораторних робіт на веб-додатку Portswigger, та проходження фінального тесту, можна бути отримати сертифікат «Burp Suite Practioner», який буде свідчить, що людина пройшла усю програму навчання, та має навички та досвід у роботі с програмою Burp Suite.

Саме наші типові об’єкти будуть навчальні середовища лабораторних робіт, які кожен раз створюються під різним доменим ім’ям, та неможливо буде використовувати одні і ті самі відповіді для лабораторних робіт. Усі лабораторні роботи у будь-якому випадку доведеться робити від початку до кінця. У цих середовищах ми будемо знати яка вразливість у системі існує, та наша задача буде

скористатися цієї вразливістю використовуючи інструменти, які в нас є (мова програмування Python для написання скриптів, та програма Burp Suite, для відстеження трафіку через HTTP протокол передачі трафіку, та запуску атак на наш типовий об'єкт.

1.4 Аналіз типових вразливостей та методи їх використання в системах автентифікації

Вразливість представляє собою слабке місце активу чи засобу управління, яке може бути використано однією та більше загрозою.

Вразливості для систем автентифікації мають деякі специфічні ознаки. Системи, що мають наступні ознаки, можуть наражатись на загрозу заповнення облікових даних:

- дозволяють багаторазово вводити неправильний логін без тимчасового блокування
- – користувачі використовують один й той самий пароль у декількох системах.

Системи, що мають наступні ознаки, можуть наражатись на загрозу захоплення облікового запису:

- приймають слабкі паролі
- дозволяють багаторазово вводити неправильний логін без тимчасового блокування;

В табл.1 наведено типові вразливості систем автентифікації, та їх критичність

Табл.1 Види типових вразливостей систем автентифікації

№	Вразливість	Критичність вразливості
1	Недостатня автентифікації	4
2	Небезпечне відновлення паролів	3
3	Вразливість до розщеплення HTTP-запиту	2
4	Неправильне збереження автентифікаторів	3
5	Відсутність тайм-ауту сеансу	3
6	Недостатня протидія автоматизації	3
7	Система дозволяє багаторазово вводити неправильний логін без тимчасового блокування	3
8	Помилка розмежування прав доступу до БД	3
9	Система приймає слабкі паролі	2
10	Користувачі використовують один і той самий пароль у декількох системах	3

Окрім цього до вразливостей систем автентифікації можна додати неправильне проектування, коли на програмну та логічному рівні допускається помилка, і деякі з обмежень, або систем захисту систем автентифікації можуть обходитися.

Э багато методів як можна використати вразливості систем автентифікації як однофакторних (стандартна автентифікація за допомогою логіна та паролю)

так і багатофакторних (наприклад логін, пароль, та PIN код який приходить до автентифікаційного додатку).

Для систем однфакторної автентифікації використовуються наступні методи використання вразливостей:

Атака методом брут-форсу — зловмисник використовує систему проб і помилок, намагаючись вгадати дійсні облікові дані користувача. Ці атаки зазвичай автоматизовані за допомогою списків імен користувачів і паролів. Автоматизація цього процесу, особливо з використанням спеціальних інструментів, потенційно дозволяє зловмиснику робити величезну кількість спроб входу з високою швидкістю.

Перебір — це не завжди випадкове вгадування імен користувачів і паролів. Використовуючи також базову логіку чи загальнодоступні знання, зловмисники можуть налаштувати атаки брут-форсу, щоб робити набагато точні припущення. Це значно підвищує ефективність таких атак. Веб-сайти, які покладаються на вхід на основі пароля як єдиний метод автентифікації користувачів, можуть бути дуже вразливими, якщо вони не реалізують достатнього захисту від брут-форсу.

Імена користувачів особливо легко вгадати, якщо вони відповідають розпізнаваному шаблону, наприклад, адресі електронної пошти. Наприклад, дуже часто можна побачити бізнес-логіни у форматі name.lastname@somecompany.com. Однак, навіть якщо очевидного шаблону немає, інколи навіть облікові записи з високим рівнем привілеїв створюються з використанням передбачуваних імен користувачів, таких як admin або адміністратор.

Паролі так само можуть бути знайдені методом брут-форсу, але складність залежить від надійності пароля. Багато веб-сайтів застосовують певну політику паролів, яка змушує користувачів створювати паролі з високою ентропією, які, принаймні теоретично, важче зламати, використовуючи лише підбір. Зазвичай це передбачає застосування паролів за допомогою:

- Мінімальна кількість символів

- Суміш малих і великих літер
- Принаймні один спеціальний символ

Однак, незважаючи на те, що комп'ютерам важко зламати високоентропійні паролі, ми можемо використати базові знання про людську поведінку, щоб використовувати вразливі місця, які користувачі мимоволі вводять у цю систему. Замість того, щоб створити надійний пароль із випадковою комбінацією символів, користувачі часто беруть пароль, який вони можуть запам'ятати, і намагаються ломом підібрати його до політики паролів. Наприклад, якщо mypassword заборонено, користувачі можуть спробувати щось на зразок Mypassword1! або замість цього Myp4$$w0rd.

У випадках, коли політика вимагає від користувачів регулярно змінювати свої паролі, також часто користувачі просто вносять незначні, передбачувані зміни до бажаного пароля. Наприклад, Mypassword1! стає Mypassword1? або Mypassword2!.

Перерахування імен користувачів — зловмисник може спостерігати за змінами в поведінці веб-сайту, щоб визначити, чи дійсне дане ім'я користувача.

Перерахування імен користувачів зазвичай відбувається або на сторінці входу, наприклад, коли ви вводите дійсне ім'я користувача, але неправильний пароль, або в реєстраційних формах, коли ви вводите ім'я користувача, яке вже зайняте. Це значно скорочує час і зусилля, необхідні для входу грубою силою, оскільки зловмисник може швидко створити короткий список дійсних імен користувачів.

Під час спроби брут-форсу сторінки входу слід звернути особливу увагу на будь-які відмінності в:

- Коди статусу: під час атаки методом грубої сили повернутий код статусу HTTP, ймовірно, буде однаковим для переважної більшості припущень, оскільки більшість із них буде неправильною. Якщо припущення повертає

інший код статусу, це є переконливим свідченням того, що ім'я користувача було правильним. Для веб-сайтів найкраще завжди повертати той самий код статусу незалежно від результату, але ця практика не завжди дотримується.

- Повідомлення про помилки: іноді повідомлення про помилку, що повертається, відрізняється залежно від того, чи неправильні і ім'я користувача, і пароль, чи неправильний лише пароль. Для веб-сайтів найкраще використовувати ідентичні загальні повідомлення в обох випадках, але іноді трапляються невеликі помилки введення. Лише один символ, який не на місці, робить два повідомлення різними, навіть у випадках, коли символ не видно на відтвореній сторінці.
- Час відповіді: якщо більшість запитів було оброблено з однаковим часом відповіді, будь-які відхилення від цього вказують на те, що за лаштунками відбувалося щось інше. Це ще одна ознака того, що вгадане ім'я користувача може бути правильним. Наприклад, веб-сайт може лише перевіряти правильність пароля, якщо ім'я користувача дійсне. Цей додатковий крок може призвести до незначного збільшення часу відповіді. Це може бути непомітно, але зловмисник може зробити цю затримку більш очевидною, ввівши надто довгий пароль, для обробки якого веб-сайту потрібно значно більше часу.

Цілком ймовірно, що атака грубою силою включатиме багато невдалих здогадок, перш ніж зловмисник успішно скомпрометує обліковий запис. Логічно, що захист від грубої сили спрямований на те, щоб автоматизувати процес якомога складніше та сповільнити швидкість, з якою зловмисник може спробувати ввійти. Два найпоширеніші способи запобігання атакам грубої сили:

- Блокування облікового запису, до якого віддалений користувач намагається отримати доступ, якщо він робить занадто багато невдалих спроб входу

- Блокування IP-адреси віддаленого користувача, якщо він робить занадто багато спроб входу підряд

Обидва підходи пропонують різний ступінь захисту, але жоден не є невразливим, особливо якщо реалізовано з використанням помилкової логіки.

Наприклад, іноді ви можете виявити, що ваша IP-адреса заблокована, якщо ви забагато разів не входите в систему. У деяких реалізаціях лічильник кількості невдалих спроб скидається, якщо власник IP успішно ввійшов у систему. Це означає, що зловмисник просто повинен буде входити у свій обліковий запис кожні кілька спроб, щоб запобігти досягненню цього ліміту.

У цьому випадку простого включення власних облікових даних через регулярні проміжки часу в список слів достатньо, щоб зробити цей захист практично марним.

Щодо багатофакторних систем автентифікації, існують таки види використання їх вразливостей:

Часом реалізація двохфакторної автентифікації має недоліки настільки, що її можна повністю обійти.

Якщо користувачеві спочатку пропонується ввести пароль, а потім пропонується ввести код підтвердження на окремій сторінці, користувач фактично перебуває в стані «ввійшов у систему» до того, як він введе код підтвердження. У цьому випадку варто перевірити, чи можна відразу перейти до сторінок «лише для входу» після завершення першого кроку автентифікації. Іноді ви виявите, що веб-сайт насправді не перевіряє, чи виконали ви другий крок перед завантаженням сторінки.

Іноді помилкова логіка двохфакторної автентифікації означає, що після того, як користувач виконав початковий крок входу, веб-сайт не перевіряє належним чином, що той самий користувач виконує другий крок.

Наприклад, на першому кроці користувач входить зі своїми звичайними обліковими даними, як описано нижче.

```
POST /login-steps/first HTTP/1.1
Host: vulnerable-website.com
...
username=carlos&password=qwerty
```

Потім їм призначається файл cookie, який стосується їх облікового запису, перш ніж вони переходять до другого етапу процесу входу:

```
HTTP/1.1 200 OK
Set-Cookie: account=carlos

GET /login-steps/second HTTP/1.1
Cookie: account=carlos
```

Під час надсилання коду підтвердження запит використовує цей файл cookie, щоб визначити, до якого облікового запису користувач намагається отримати доступ:

```
POST /login-steps/second HTTP/1.1
Host: vulnerable-website.com
Cookie: account=carlos
...
verification-code=123456
```

У цьому випадку зловмисник може увійти, використовуючи власні облікові дані, але потім змінити значення файлу cookie облікового запису на будь-яке довільне ім’я користувача під час надсилання коду підтвердження.

```
POST /login-steps/second HTTP/1.1
Host: vulnerable-website.com
Cookie: account=victim-user
```

```
...
verification-code=123456
```

Це небезпечно, якщо потім зловмисник зможе підібрати код підтвердження, оскільки це дозволить йому входити в облікові записи довільних користувачів виключно на основі їх імені користувача. Їм навіть не знадобиться знати пароль користувача.

1.5 Дослідження методів захисту систем автентифікації

1. Переконатися, що були використані стандарти безпечної розробки.

Традиційне тестування безпеки відбувається безпосередньо перед або після випуску. Це може бути занадто пізно для перевірки безпеки. Це також може призвести до неочікуваних затримок критичних термінів, якщо проблеми з безпекою будуть виявлені пізніше.

Стандарти безпечної розробки диктують, що замість тестування безпеки під час або після випуску, планування безпеки вводиться на кожному етапі конвеєра. Це дозволяє проводити тестування на ранніх етапах циклу розробки програмного забезпечення. Перевага цього підходу полягає в тому, що кодова база безпечна та перевірена на безпеку перед тим, як її прийняти до збірки.

Стандарти безпечної розробки не замінюють традиційне тестування безпеки.

2. Регулярна перевірка безпеки:

Щоразу, коли створюється нова інфраструктура, необхідно проводити перевірки інформаційної безпеки, щоб виявити та зменшити ризики. Перевірки

також необхідно проводити, коли в існуючу систему вносяться значні зміни або коли надається доступ третім особам до внутрішніх систем.

3. Статичний аналіз коду

Статичний аналіз коду — це аналіз сховищ коду, який виконується без виконання самого коду (аналіз запущених програм називається «динамічним аналізом»). Статичний аналіз коду виявляє неоптимальний і неефективний код, сигналізуючи командам інженерів про необхідність усунути проблеми. Цей процес означає, що команда надсилає лише якісний надійний код.

Статичний аналіз коду – це не те саме, що огляд коду. Статичний аналіз коду виконується машиною, тоді як перевірка коду виконується людиною. По суті, статичний аналіз коду автоматизований, а перевірка коду виконується вручну.

4. Тестування на проникнення:

Тест на проникнення (тест пера) — це метод грубої сили, який проводять хакери кібербезпеки для виявлення вразливостей. Тести на проникнення проводяться власними співробітниками або підрядниками, які імітують дії зловмисника.

Існує три типи тестування пером: біла коробка надає тестувальнику всі деталі про систему організації, чорна коробка надає тестувальнику жодних знань про систему, а сіра коробка надає тестувальнику часткові знання.

5. Програма знаходження помилок коду (Bug Bounty):

Програма винагороди за помилки – це програма винагород, яку пропонує організація, яка стимулює повідомлення про помилки та вразливості. Це чудовий спосіб доручити виявлення помилок безпеки іншим розробникам. Великі технологічні компанії часто запускають програми винагород за помилки.

6. Зробити системи захисту від брут-форс атак:

Хакери, як правило, використовують методи грубої сили, щоб увійти у вашу систему. Одним із найпоширеніших методів боротьби з цим є обмеження кількості користувачів на основі IP. Для цього потрібно запобігти маніпулюванню хакерами їхньою IP-адресою. Якщо користувач кілька разів намагався увійти, попросіть його пройти тест CAPTCHA.

7. Правильно спроектувати методи багатофакторної системи автентифікації:

Багатофакторна автентифікація вимагає принаймні двох етапів перевірки особи, тому вона набагато безпечніша, ніж вхід на основі пароля. Паролі вразливі для хакерів, тому в сучасних умовах пароля недостатньо для підтвердження онлайн-ідентичності. Багатофакторна автентифікація пропонує додатковий рівень захисту.

8. Перевіряти журнали входу:

Журнал аудиту виявляє дії потенційного зловмисника, позначаючи будь-що підозріле. Відповідальністю за збір журналів мають займатися як служби безпеки, так і групи розробки програмного забезпечення. Журнали корисні для оцінки інцидентів безпеки під час посмертного аналізу.

Журнали також важливі для моніторингу. Якщо було помічено незвичайний трафік сервера з нерозпізнаної IP-адреси, можна швидко просканувати журнали, щоб виявити незвичну активність і швидше зменшити ризики та вразливі місця. Щоб підтримувати стабільність системи, переконайтеся, що належні журнали створюються та перевіряються адміністраторами, мережевими інженерами та розробниками.

1.6 Постановка задачі

Згідно з метою кваліфікаційної роботи, поставлені наступна задачі до спеціальної частини:

1. Розглянути інструментарій, який буде використовуватися під час тестування систем автентифікації на прикладі типового об’єкту.
2. За допомогою навчального середовища типового об’єкту, протестувати системи автентифікації на наявнвість вразливостей, та використати їх для отримання неавторизованого доступу до акаунту користувача
3. За результатами тестування систем автентифікації на прикладі типового об’єкту розробити покрокову інструкцію тестування систем автентифікації
4. За результатами тестування розробити рекомендації для підвищення рівня захисту систем автентифікації веб-додатків.

1.7 Висновок

В першому розділі була визначена актуальність обраної теми для кваліфікаційної роботи. Розглянуті механізми роботи різних систем автентифікації (однофакторної та багатофакторної системи), та видів (верифікаційний код до автентифікаційного додатку та носій с електронним ключем для методу багатофакторної автентифікації U2F) , приведені схеми роботи систем автентифікації та проаналізовані типові вразливості систем автентифікації веб-додатків. Оглянуто методи захисту систем автентифікації та були поставлені задачі до спеціального розділу кваліфікаційної роботи.

Розділ 2 СПЕЦІАЛЬНИЙ РОЗДІЛ

2.1 Характеристика інструментарію

Для тестів на вразливості систем автентифікації буде використовуватися допоміжне програмне забезпечення під назвою Burp Suite. На рис. 2.1 зображено як виглядає Burp Suite та його стандартний функціонал.

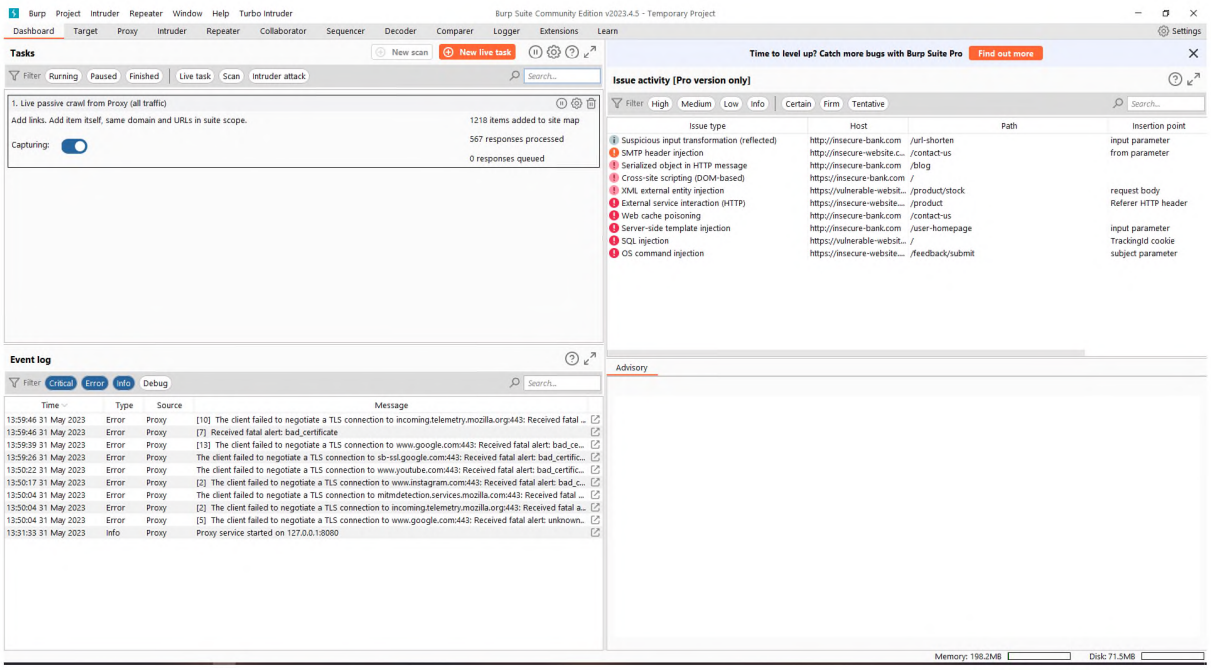


Рис. 2.1 Головне меню Burp Suite

Burp Suite має функціонал, який потрібен для здійснення багатьох атак на веб-додатки. По більшій було використовувати функцію Intruder, для ініціації атаки брут-форсу. Функціонал Burp Suite, окрім стандартних опцій, можна розширити за допомогою “Розширень”, які можна завантажити прямо с додатка. На рис 2.2 зображено де можна отримати нові додатки у Burp Suite.

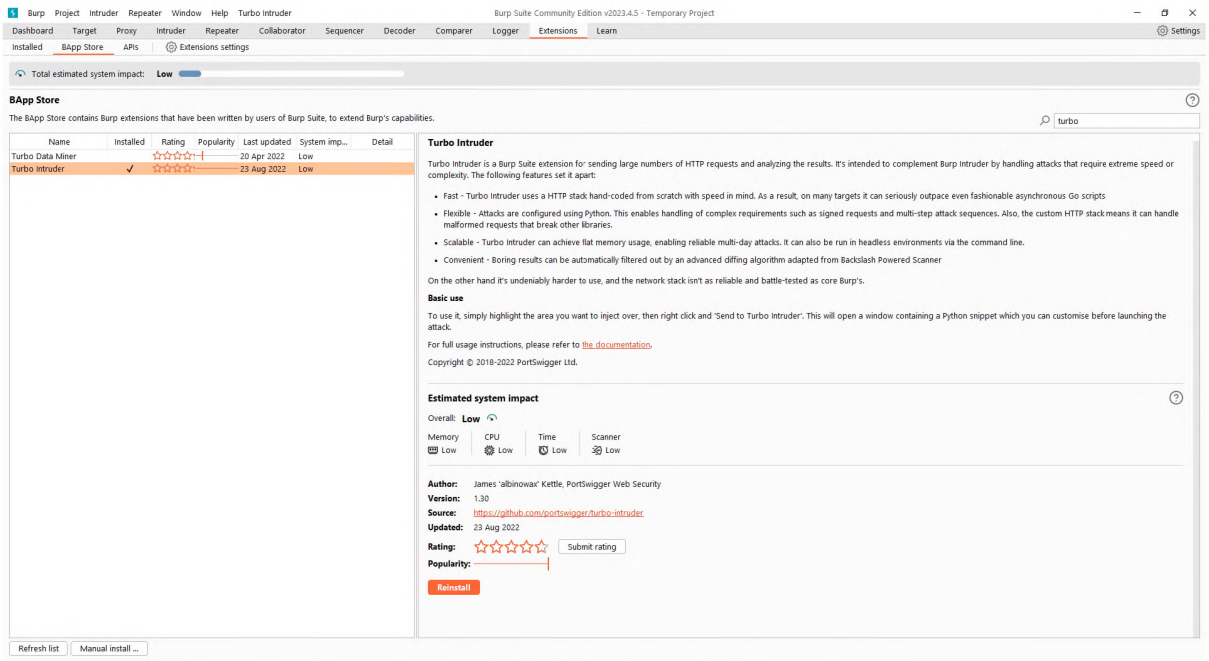
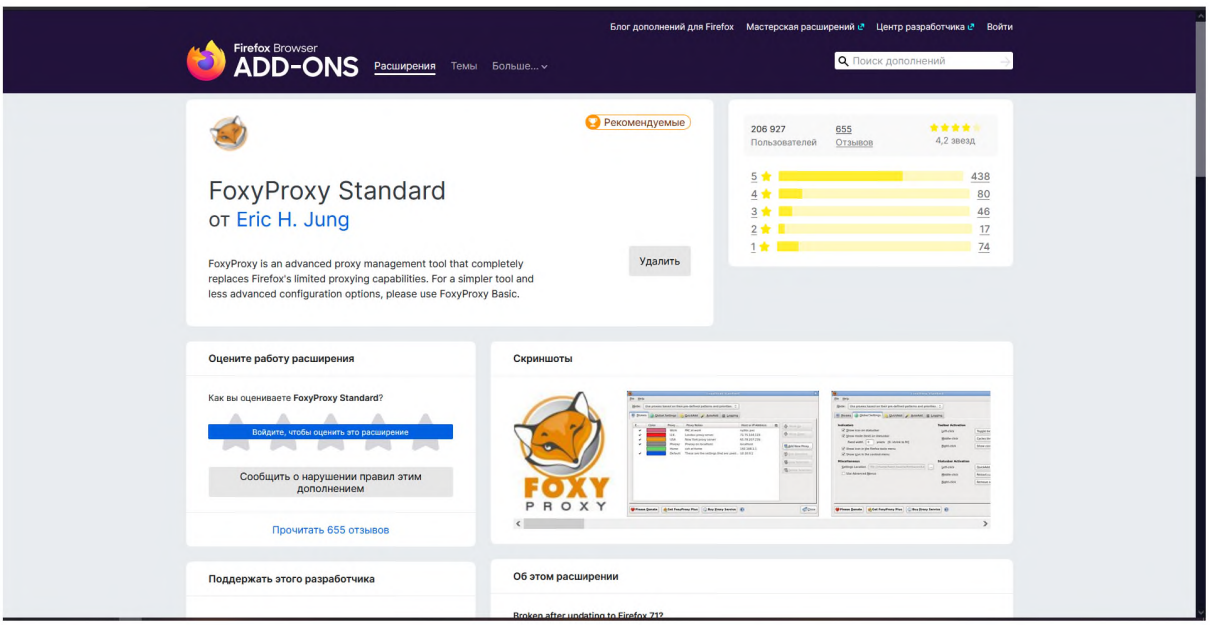


Рисунок 2.2 Меню додатків Burp Suite

За для того, щоби з'їднати Burp Suite із браузером, ще треба встановити та налаштувати спеціальний веб-VPN під назвою FoxyProxy, сторінку с VPN наведено у рис. 2.3, на рис. 2.4 наведено приклад налаштувань VPN для використання з браузером FireFox.



2.3 Сторінка с Веб-VPN FoxyProxy

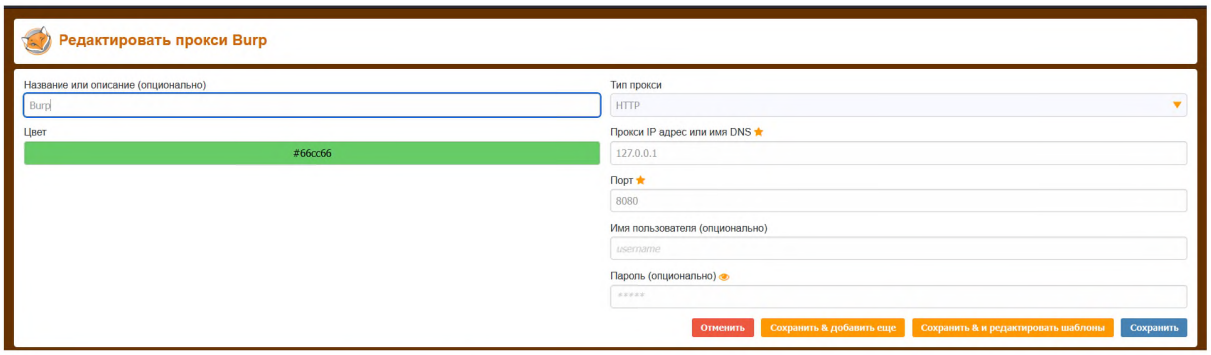


Рис 2.4 Налаштування FoxyProxy

Деталі для налаштування VPN були взяті напряму с Burp Suite. Деталі налаштування наведено на рис. 2.5

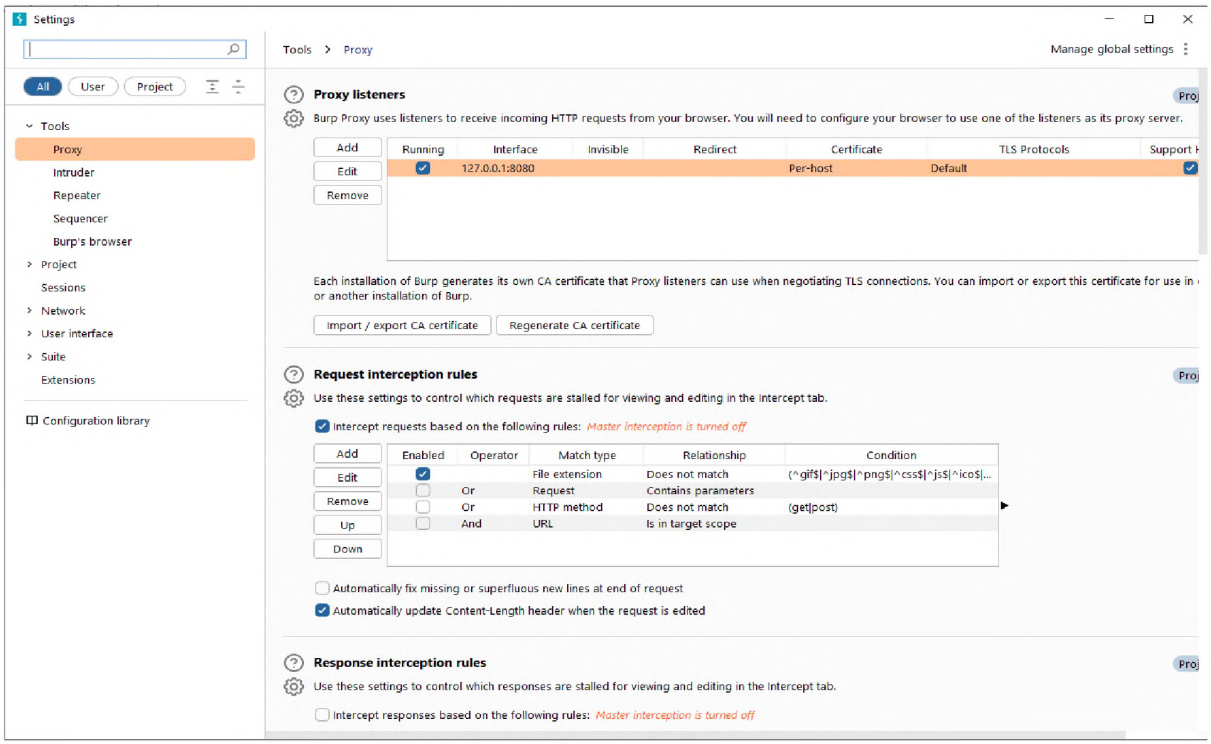


Рис 2.5 Деталі для налаштування Burp Suite

Окрім Burp Suite, також будуть використовуватися скрипти написані мовою програмування Python. Як редактор коду буде використано редактор коду Visual Code.

Всі коди скриптів винесені окремо до додатків.

2.2 Дослідження вразливостей систем однофакторної автентифікації до атак брут-форсу

Загалом видів вразливосте систем однофакторної автентифікації та способів отримання доступів до акаунтів існує багато, та брут-форс не єдина вразливість систем автентифікації. До типових вразливостей можна також відносити наприклад SQL ін'єкції, HTML ін'єкції, Cross-site scripting, та інші. В дані роботі сконцентровано увагу саме на вразливостях систем однофакторної автентифікації до брут-форс атак тому, що вони прості в використані, вони масові, кількість атак завжди велика, через простоту ініціювання, тому треба розробляти ефективні методи захисту від такого виду атак.

Проаналізовано декілька випадків використання вразливостей "Username Enumeration". Для тесту був використаний навчальне середовище веб-сайт, для тестування системи автентифікації. Нам одразу відомо, що веб-сайт вразливий до брут-форсу.

У нас навчальне сероедовище веб-сайт (рис. 2.6):

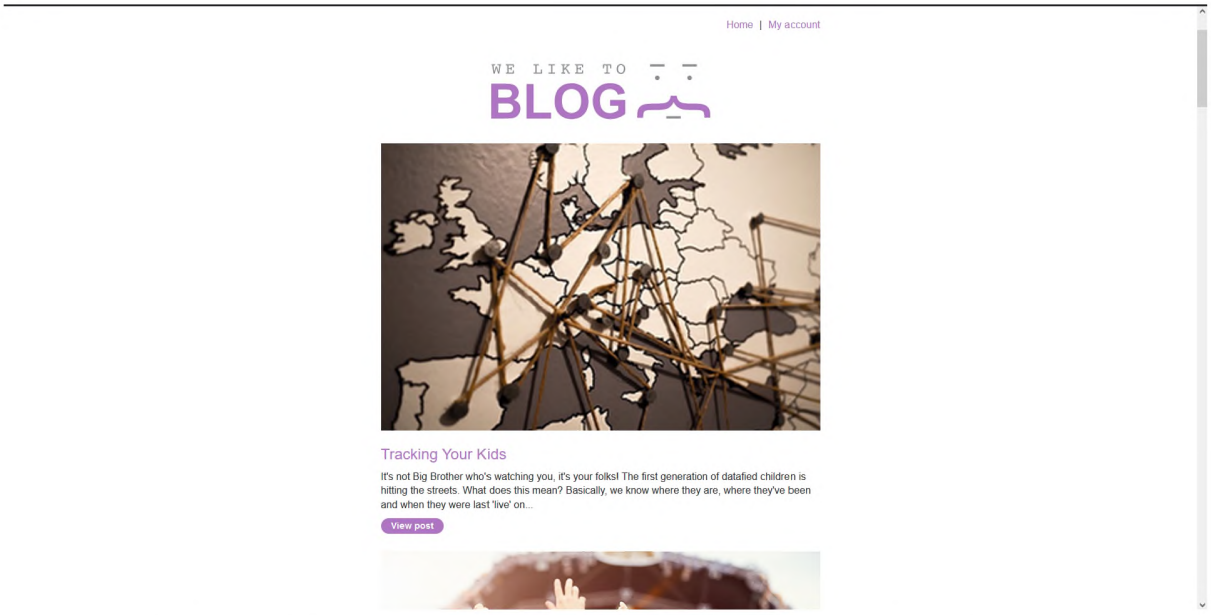


Рис. 2.6 Знімок екрану навчального середовища веб-сайту

Це звичайний веб-сайт, де ведеться блог від різних користувачів. Сайт не містить багатого функціоналу. Проте, цей сайт містить особистий кабінет для входу, та створення статей. На рис 2.7 зображено форму логіну до особистого кабінету.

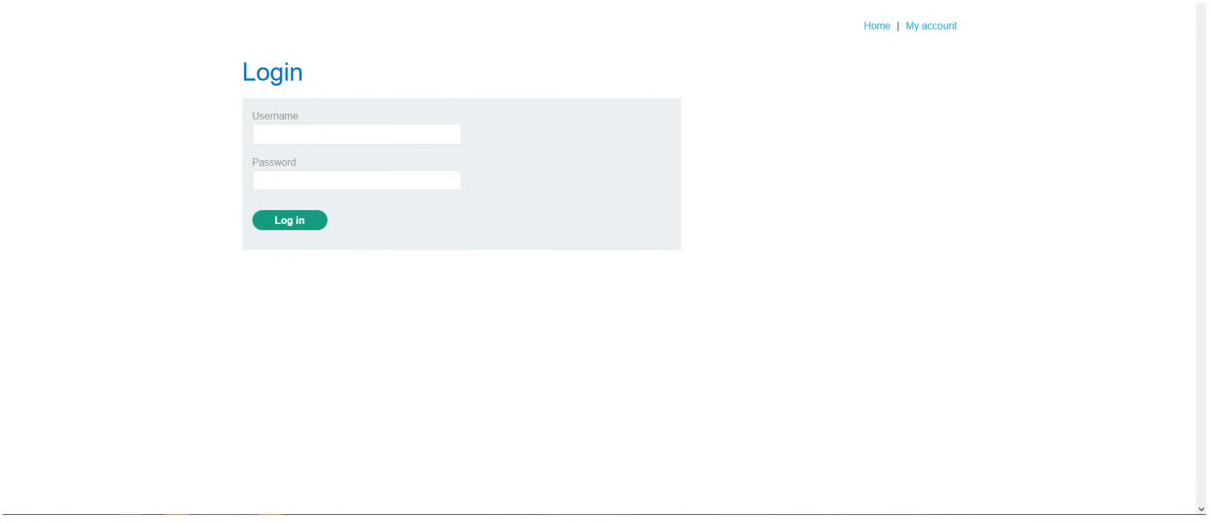


Рис. 2.7 Вікно логіну веб-сайту полігона

Через це вікно було почато майже усі атаки. Відомо, що цей веб-сайт вразливий до енумерації логінів, тому можна провести брут-форс атаку на цей сайт.

Брут-форс атака проводилась за допомогою Burp Suite. Для цього потрібно його запустити та налаштувати для відстеження трафіку с веб-сайту, на який готується атака. Вікно відстеження трафіку представлено у рис. 2.8.

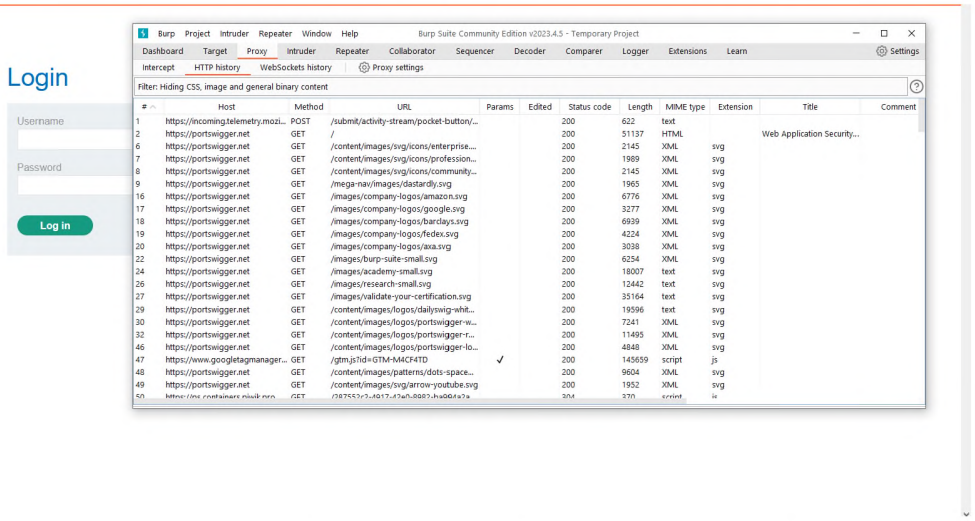


Рис.2.8 Вікно відстеження трафіку у Burp Suite

На рис.2.8 видно багато різних записів, довжин, значень, та методів запиту до серверів. В списку присутні багато зайвих доменів, трафік з яких відстежувати непотрібні, тому треба додати необхідний домен до цілі дослідження Burp Suite. На рис. 2.9 зображено параметр, який нам треба відстежувати.

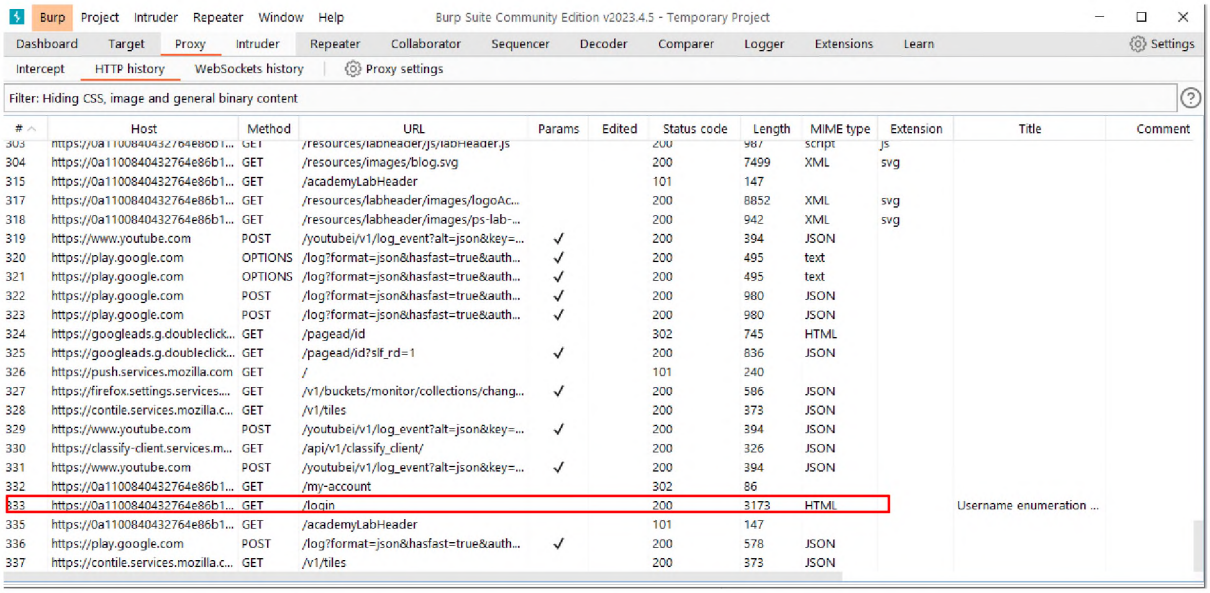


Рис.2.9 Параметр який буде відстежуватися

Щоби додати цей параметр до цілі дослідження, треба визвати контекстне меню та натиснути «Add to scope». На рис 2.10 зображено процес додавання доменого імені до цілі дослідження.

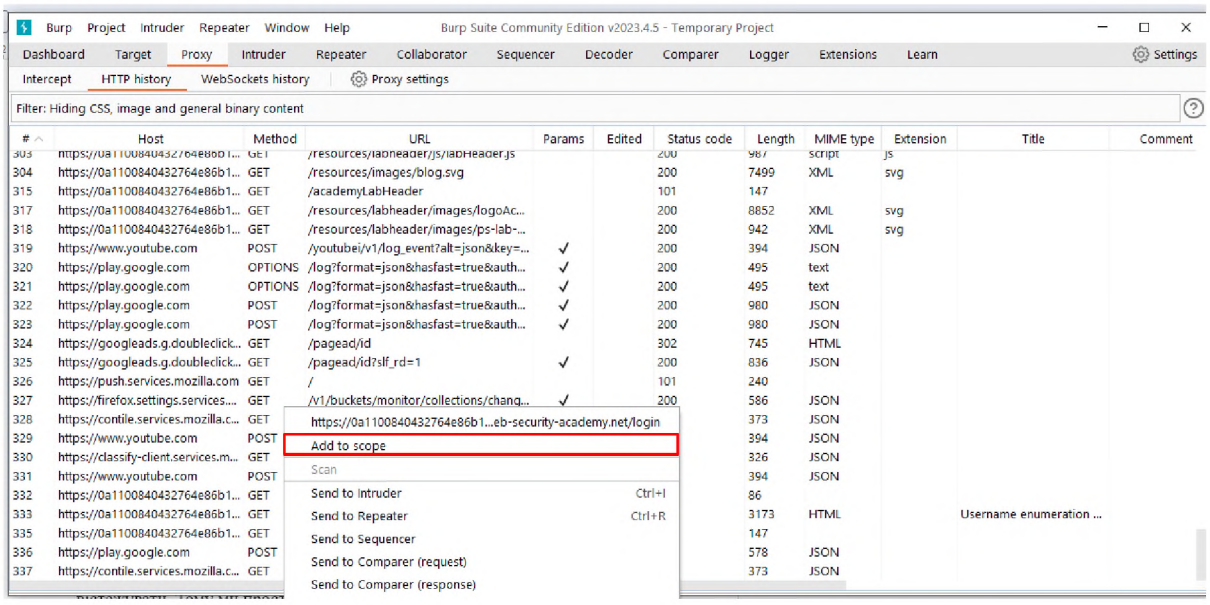
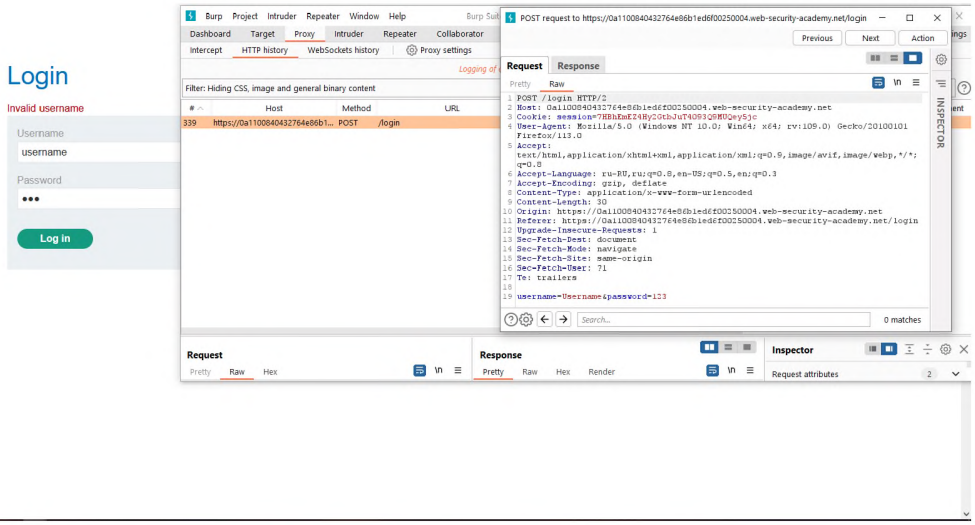


Рис. 2.10 Процес додавання домена до цілі спостереження.

Після цього можна очистити історію HTTP запитів для того, щоби позбутися непотрібних записів і бачити тільки нашу ціль відстеження.

Тепер коли видно лише потрібний домен, та зайві записи не заважають можна зробити запит на вхід до особистого кабінету використовуючи будь-які дані, тільки для того, щоби подивитися як система реагує на спроби входу до акаунту. Було використаємо наступні дані: Логін: Username, Пароль: 123. Результат такого запиту зображено на рис. 2.11



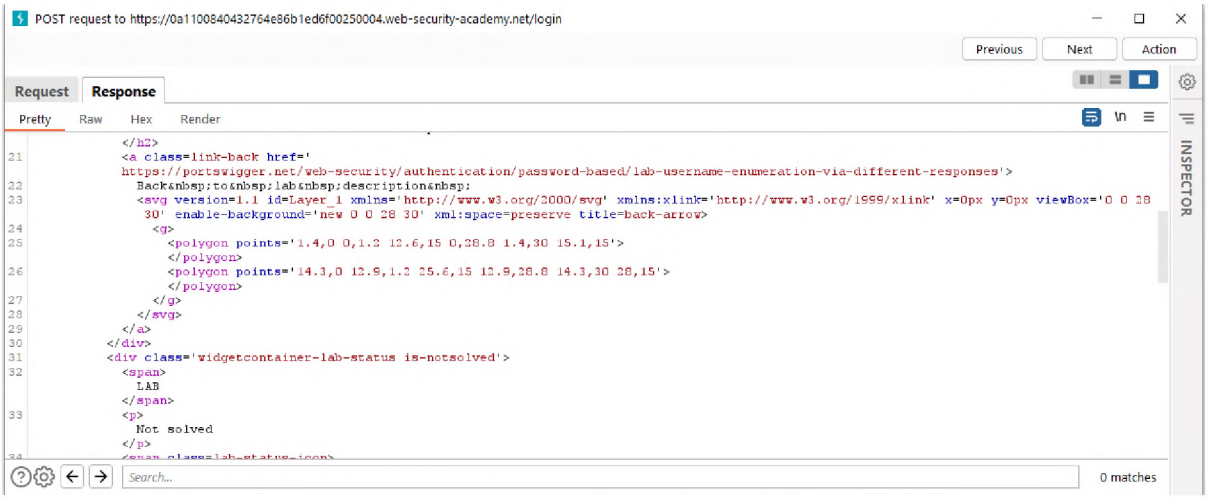
Результат 2.11 Результат запиту на логін використовуючи випадкові деталі.

На рис 2.11 ми можна побачити, що запит на вхід з даними приведеними вище опинився невдалим. Можна побачити деталі нашого запиту в Burp Suite у вкладці «Request». На разі, тут немає нічого корисного. Можна помітити ідентифікатор кукі сесії, версію браузеру, мову, кодування, проте ніякої користі у даному випадку це не несе. Проте потрібне поле за номером 19 надає наступну інформацію:

“username=Username&password=123 “

Саме через ці поля було виконано брут-форс атаку, вибираючи один с потрібних там пунктів, username або password, та під ставляючи завчасно заготовлені словники

Що дійсно цікавить у цьому випадку, це вікно відповіді серверу на наш запит, який зображено на рис. 2.12.



2.12 вікно відповіді на запит входу

Повний код запиту буде додано до Додатку А. У цьому запиті можна побачити, відповідь сервера “ HTTP/2 200 OK “, що означає, що з’єднання було встановлено, та запит був відпрацьований успішно. Також можемо побачити довжину поверненої нам відповіді «Content-Length: 3130». Далі іде звичайна верстка HTML. Важливий пункт, який вписаний у цьому тезі:

```
<p class=is-warning>Invalid username</p>
```

Сторінка повернула відповідь “ Invalid username “. Цей тег можна побачити реалізованим на сайті. Приклад наведений на рис. 2.13.

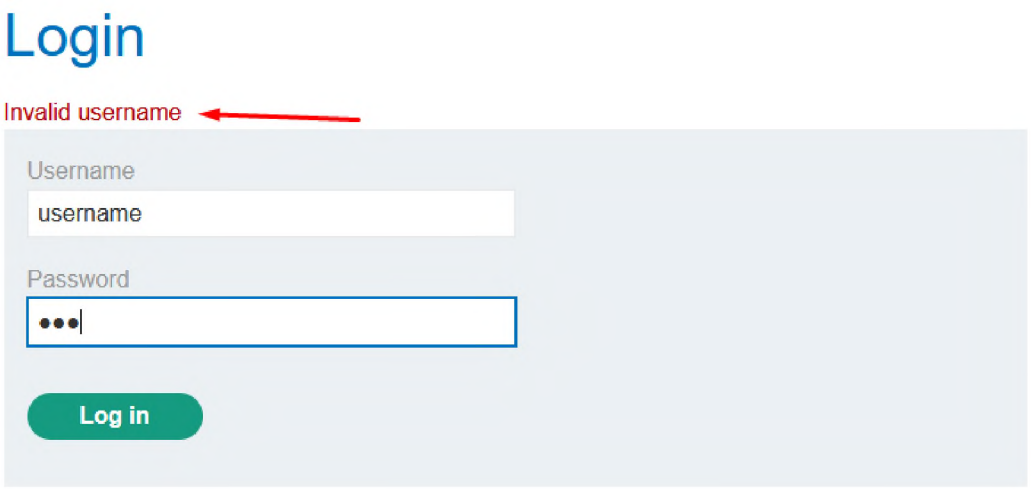


Рис. 2.13 Помилка «Invalid Username» при веденні не дійсного логіну

Що означає, що вели неіснуюче ім'я користувача в цій базі даних, і якщо буде введемо правильне, то система скаже щось на кшталт "Invalid password", що дасть знати, що існуюче ім'я користувача було знайдено, та залишилося підібрати до нього пароль. Слід зауважити, що такі відповіді та реакції систем автентифікації можуть дуже відрізнятися одне від одного на різних веб-додатках, тому слід пам'ятати шаблони того як система може реагувати, та залежно від цього обирати наступний крок, або метод яким система автентифікації буде тестуватися далі.

Для того, щоб підібрати логін та пароль було використано заготовлені для цього словники.

Перед тим як буде можливо використати словник для підстановки логінів, потрібно додати кладку с вікном логіну до Burp Suite Intruder (Розширення, яке дозволяє надсилати запити на вхід до акаунту до системи автентифікації, використовується для брут-форсу методом перевірки усіх значень зі словника с одним полем входу.) Також можна піти зовсім грубою силою та просто почати перебирати усі логіни с усіма паролями із списку. Так, це більше автоматизований процес, проте набагато повільніший. На підбір методом снайпингу може піти від декількох хвилин до годин, в разі ж с перебором усіх значень, цей процес може затягнутися на доби, через велику кількість комбінацій.

Словники також можуть бути здобуті різними шляхами. Починаючи від написання скриптів, які генерують логіни та паролі, до сайтів, які дають такі словники з поширеними та популярними логінами та паролями.

Отже, спочатку треба вибрати поле " Username ", та подивитися, чи можливо підібрати валідне ім'я використовуючи словник логінів. Процес додавання вкладки с формою логіну до Burp Suite intruder представлено у рис 2.14.

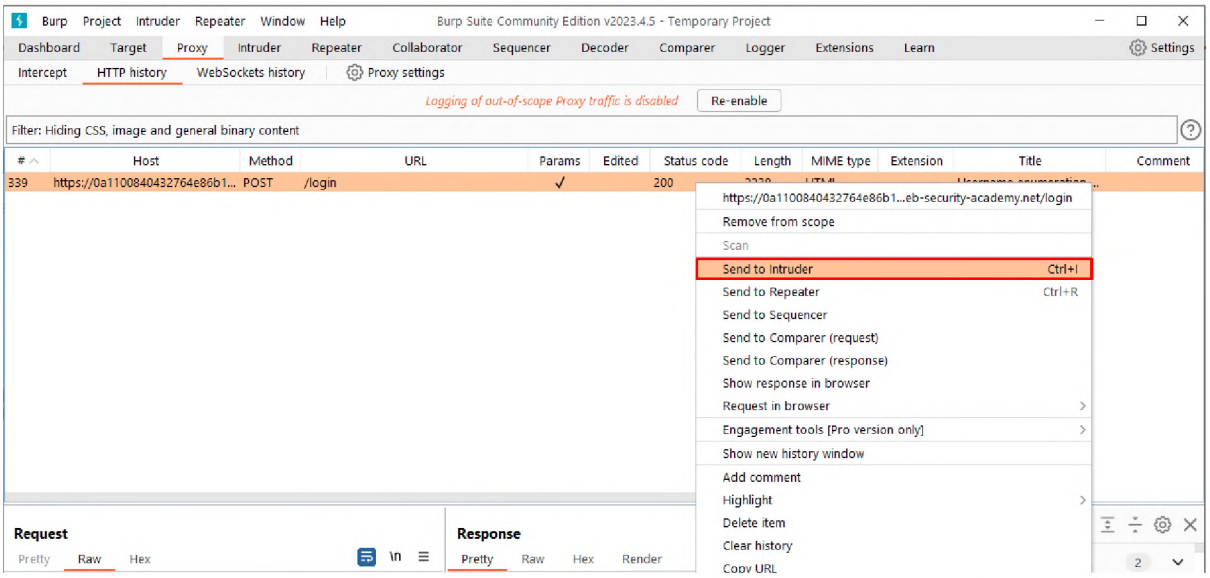


Рис.2.14 Процес додавання вкладки /login до Burp Suite Intruder/

Після того, як ця вкладка була додана до Burp Suite Imtruder, треба перейти до цього розширення в меню. Меню Burp Suite Intruder можна розглянути на рис. 2.15

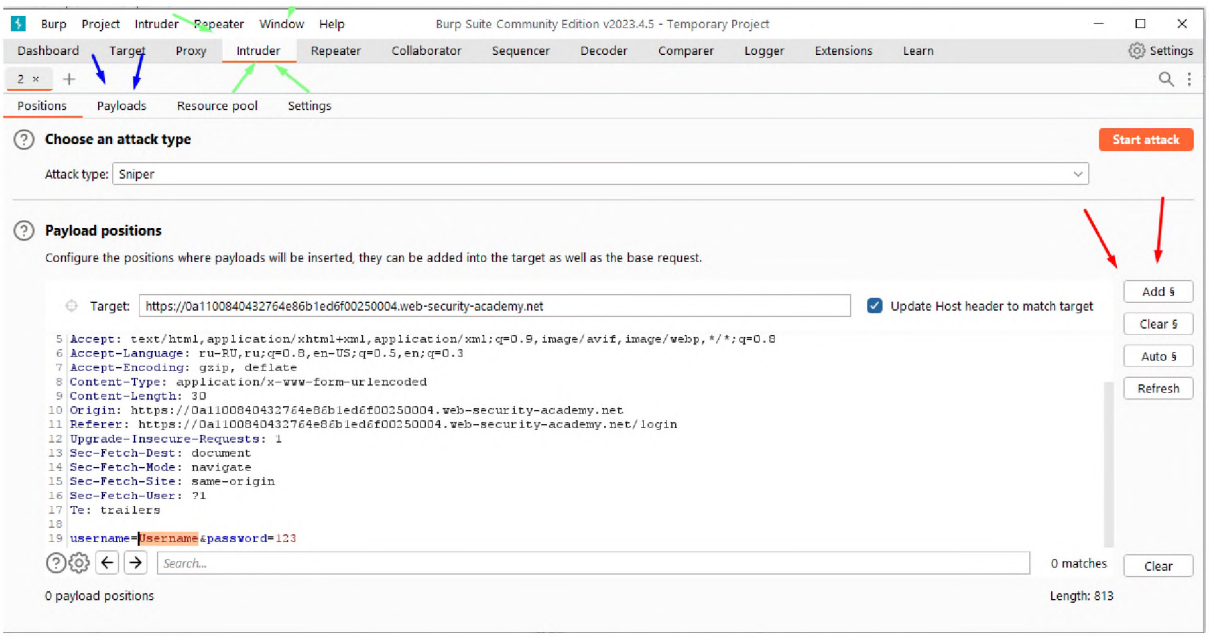


Рис. 2.15 Меню Burp Suite Intruder

Для того, щоби перейти до меню Burp Suite Intruder, треба у верхній навігаційній панелі обрати вкладу “Intruder”. Тут є вкладка Payloads, де будуть

розташовані наші словники, які будуть використовуватися для перебору. Ця вкладка помічена синіми вказівниками.

Для того, щоби почати підставляти слова зі словнику, треба обрати потрібний параметр та оточити його символами “ § “, так система зрозуміє, що в цих рамках треба підставляти слова з Payloads. Кнопка для додавання цього символу відмічена червоними вказівниками.

Меню вкладки Payloads зображено на рис. 2.16

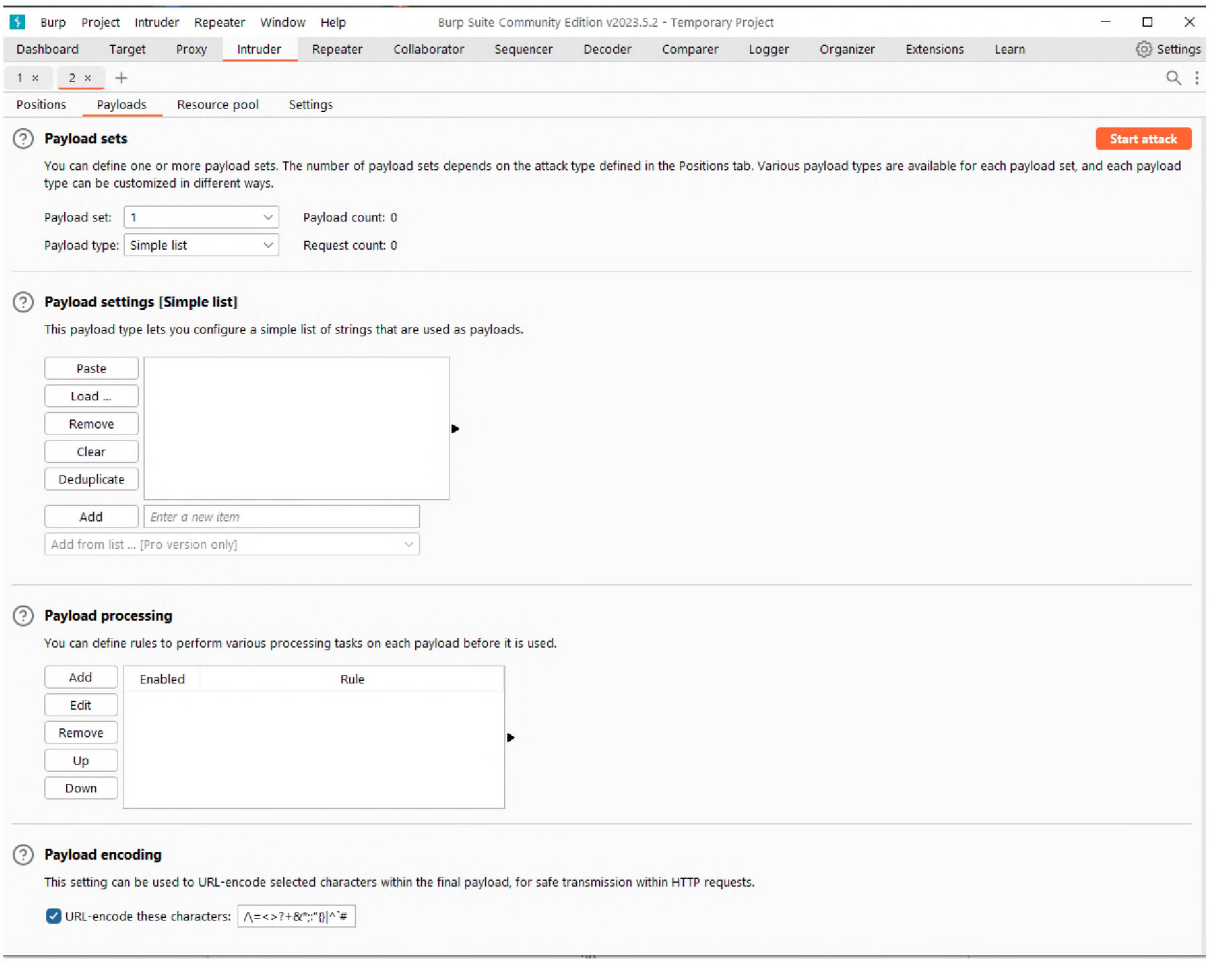


Рис. 2.16 Меню словників.

Так як обрано лише одна позиція для підстановки, то можна додати лише один словник одночасно. Для того, щоби додати словник, треба скопіювати його та вставити його у поле Payload settings (Simple list). Фінальний результат надано на рис. 2.17:

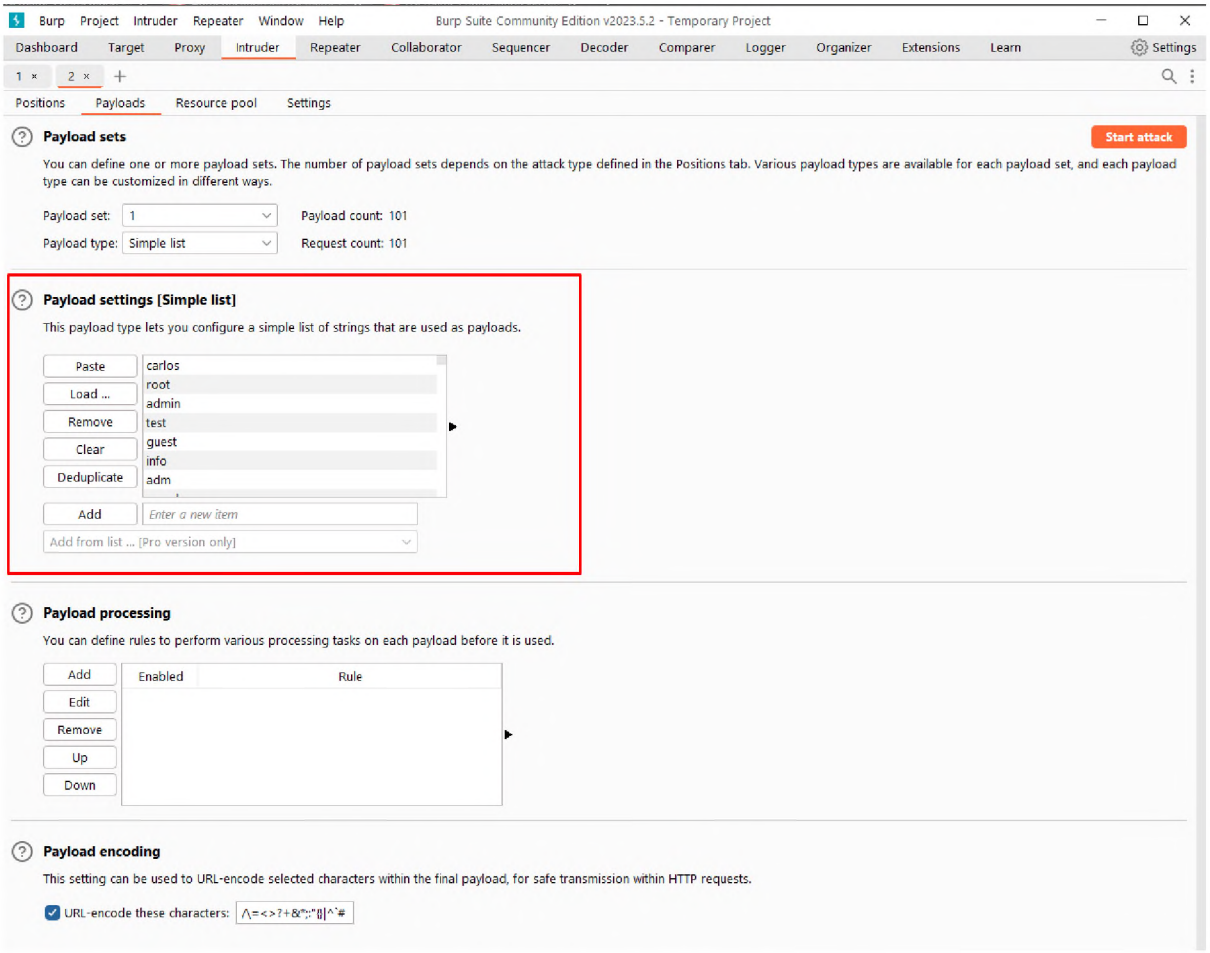
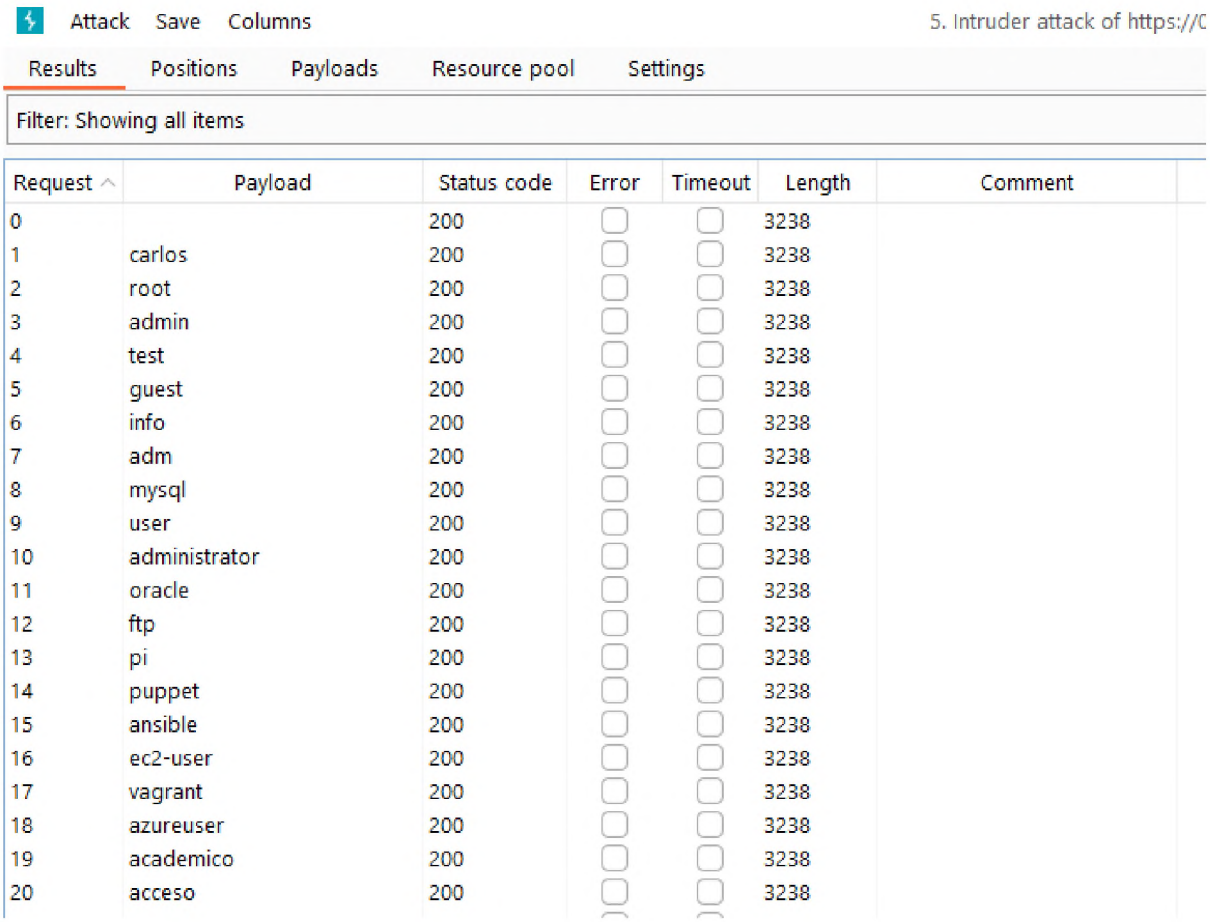


Рис 2.17 Меню словників

Для початку атаки треба натиснути на кнопку Start attack, та відстежити результати, які ми отримаємо внаслідок атаки. На рис. 2.18 можна побачити меню Burp Suite Intruder під час атаки на веб-додаток. Там наведені логіни з нашого словнику, поле статус коду відповіді сервера на запит входу до акаунту, та довжина відповіді. Ці ключові значення знадобляться для того, щоби знайти існуючий логін в базі даних. На рис 2.19 позначено який параметр підставляється під час атаки.



Request	Payload	Status code	Error	Timeout	Length	Comment
0		200			3238	
1	carlos	200			3238	
2	root	200			3238	
3	admin	200			3238	
4	test	200			3238	
5	guest	200			3238	
6	info	200			3238	
7	adm	200			3238	
8	mysql	200			3238	
9	user	200			3238	
10	administrator	200			3238	
11	oracle	200			3238	
12	ftp	200			3238	
13	pi	200			3238	
14	puppet	200			3238	
15	ansible	200			3238	
16	ec2-user	200			3238	
17	vagrant	200			3238	
18	azureuser	200			3238	
19	academico	200			3238	
20	acceso	200			3238	

Рис. 2.18 Процес атаки.

```
1  POST /login HTTP/2
2  Host: 0ab800b4047660568273 3d4d0000001b.web-security-academy.net
3  Cookie: session=fXrUzCK5dqojRo0TtXsu3pNLm7OS9z7k
4  User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0
5  Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
6  Accept-Language: ru-RU,ru;q=0.8,en-US;q=0.5,en;q=0.3
7  Accept-Encoding: gzip, deflate
8  Content-Type: application/x-www-form-urlencoded
9  Content-Length: 30
10 Origin: https://0ab800b4047660568273 3d4d0000001b.web-security-academy.net
11 Referer: https://0ab800b4047660568273 3d4d0000001b.web-security-academy.net/login
12 Upgrade-Insecure-Requests: 1
13 Sec-Fetch-Dest: document
14 Sec-Fetch-Mode: navigate
15 Sec-Fetch-Site: same-origin
16 Sec-Fetch-User: ?1
17 Te: trailers
18
19 username=§Username§&password=123
```

Рис 2.19 Параметр, який підставляється за допомогою словника

Тобто, іде перебір усіх можливих логінів для того, щоб перевірити чи однаково система реагує на запити з різними даними. Результат перевірки можна побачити на рис. 2.20

Request	Payload	Status code	Error	Timeout	Length
0		200	☐	☐	3238
1	carlos	200	☐	☐	3238
3	admin	200	☐	☐	3238
2	root	200	☐	☐	3238
5	guest	200	☐	☐	3238
4	test	200	☐	☐	3238
6	info	200	☐	☐	3238
7	adm	200	☐	☐	3238
8	mysql	200	☐	☐	3238
9	user	200	☐	☐	3238
10	administrator	200	☐	☐	3238
11	oracle	200	☐	☐	3238
12	ftp	200	☐	☐	3238
13	pi	200	☐	☐	3238
14	puppet	200	☐	☐	3238
15	ansible	200	☐	☐	3238
16	ec2-user	200	☐	☐	3238
17	vagrant	200	☐	☐	3238
18	azureuser	200	☐	☐	3238
19	academico	200	☐	☐	3238
20	acceso	200	☐	☐	3238
21	access	200	☐	☐	3238
22	accounting	200	☐	☐	3238
23	accounts	200	☐	☐	3238
24	acid	200	☐	☐	3238
25	activestat	200	☐	☐	3238
26	ad	200	☐	☐	3238
27	adam	200	☐	☐	3238
28	adkit	200	☐	☐	3238
29	admin	200	☐	☐	3238
30	administracion	200	☐	☐	3238
31	administrador	200	☐	☐	3238
32	administrator	200	☐	☐	3238
33	administrators	200	☐	☐	3238
34	admins	200	☐	☐	3238
35	ads	200	☐	☐	3238
36	adserver	200	☐	☐	3238
37	adsl	200	☐	☐	3238
38	ae	200	☐	☐	3238
39	af	200	☐	☐	3238
40	affiliate	200	☐	☐	3238
41	affiliates	200	☐	☐	3238
42	afiliados	200	☐	☐	3238
43	ag	200	☐	☐	3238

Рис. 2.20 Результат підбору логінів.

В цілому усі запити майже не відрізняються одне від одного. Усі запити сервер обробив та відав відповідь «200 OK», проте більшість з цих логінів не є дійсними, ми можемо це перевірити у вкладці «Responsee» на рис. 2.21

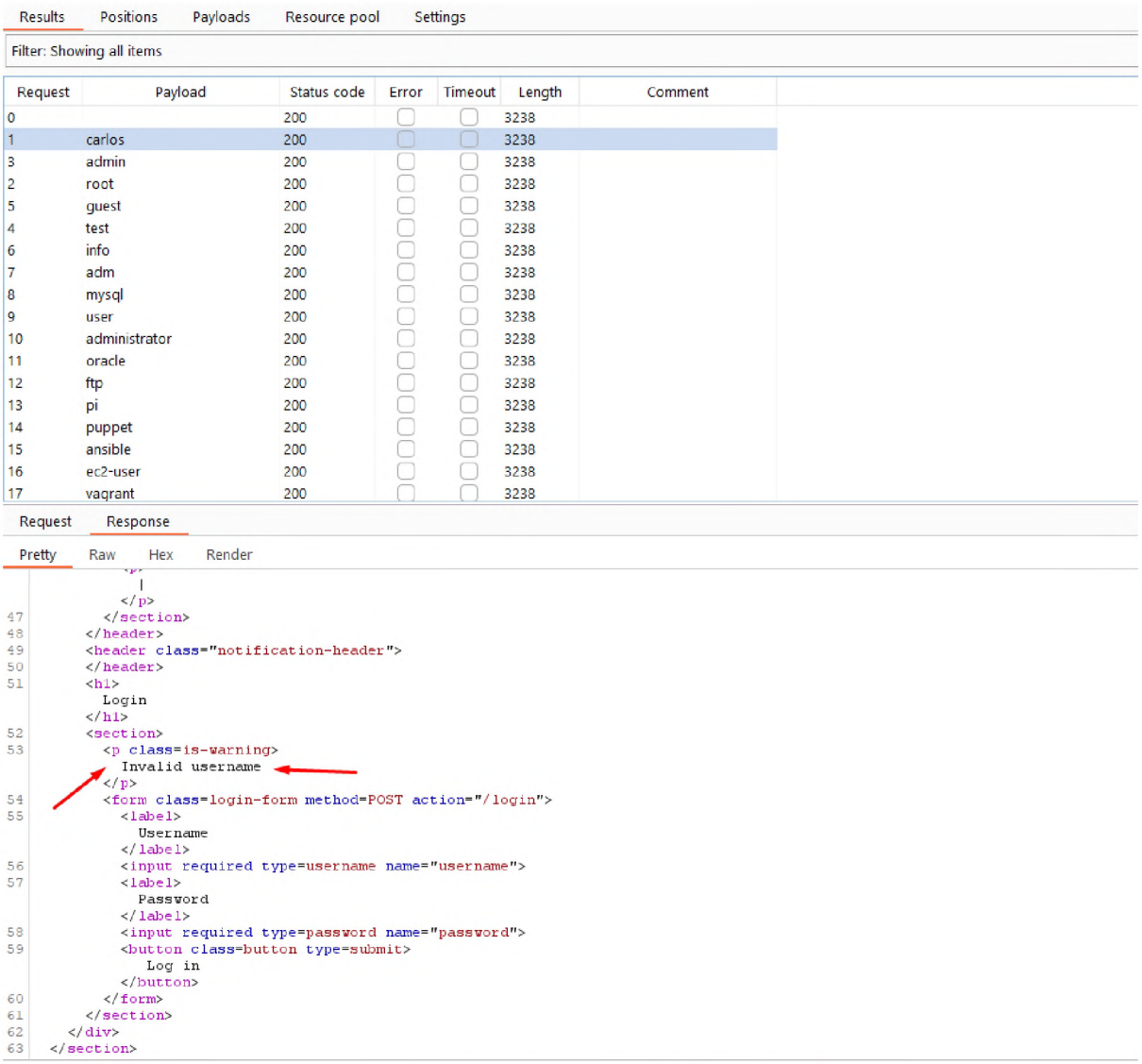


Рис.2.21 Вкладка «Response»

Якщо зробити сортування відповідей за довжинами, можна бути помітити, шо не всі відповіді мають однакову довжину і одна з них виокремлюється від інших, це і може бути дійсний логін, який може бути використаний для подальших спроб брут-форсу. Виконаємо сортування, та подивимося на відповідь сервера, щодо цього логіну зображеного на рис. 2.22:

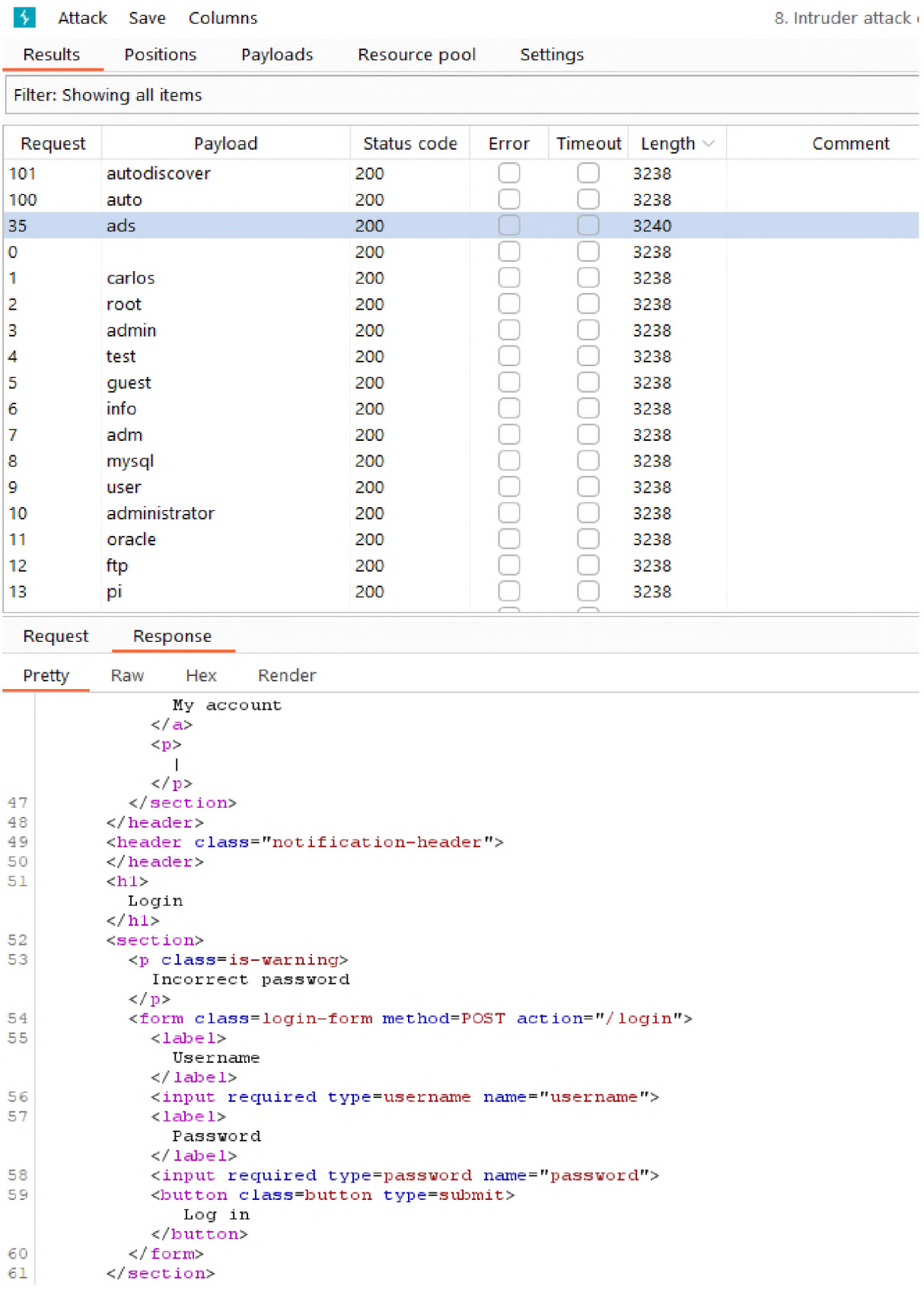


Рис 2.22 Вікно Burp Suite intruder,відповідь з іншою довжиною.

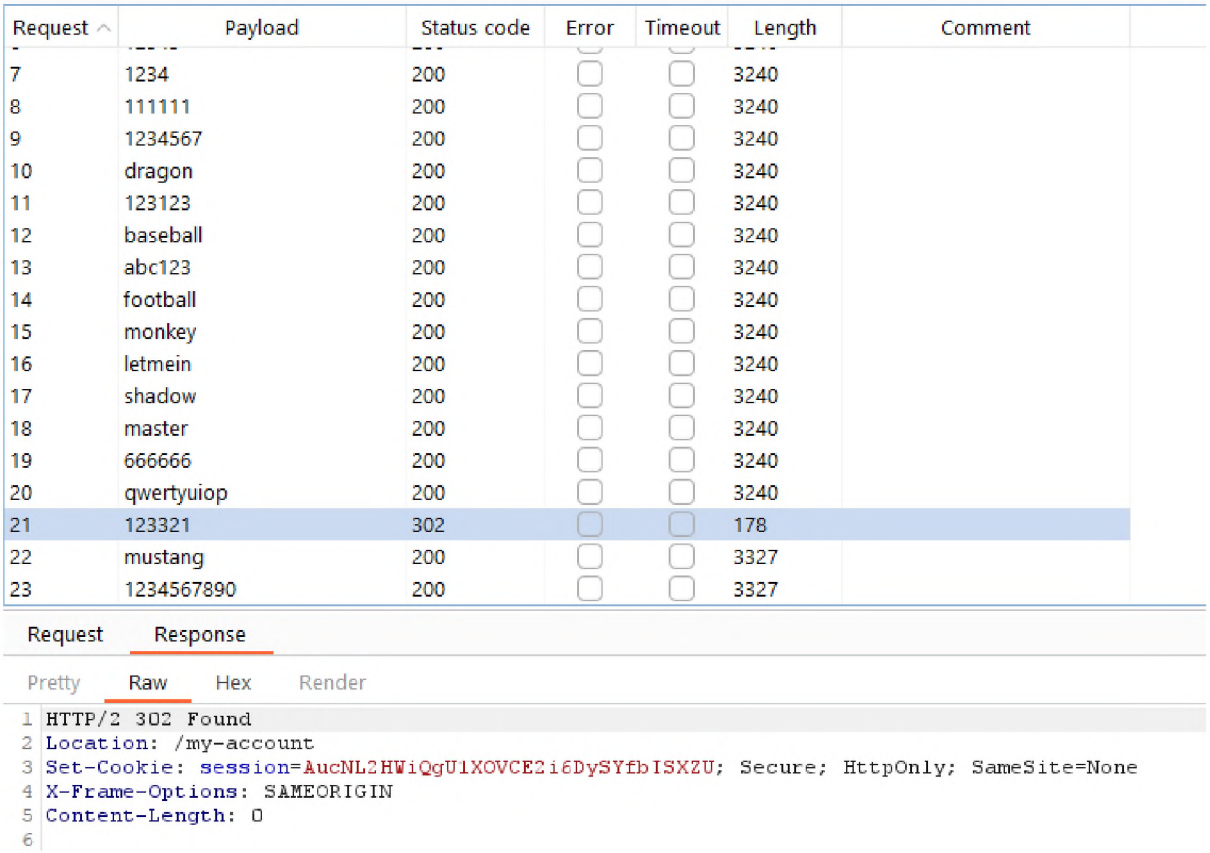
У логіна app1 інша довжина відповіді, якщо подивитися на цю відповідь, то можна побачити, що цього разу, замість “Incorrect Username” вже помилка “Incorrect password”, що цілком може означати, що було підібрано логін, а тепер методом підбора треба підставити потрібний пароль для цього логіну. Щоби це зробити слід змінити параметр логіну зі значення “Username” на логін, який було здобуто, а саме: “ads”. Щоби підставляти паролі до цього логіну, треба взяти пароль до позиції в яку буде підставлятися словник. Процес зміни позиції зображено на рис 2.23

```
POST /login HTTP/2
Host: 0a30007203027e3e80f97b6600e600bd.web-security-academy.net
Cookie: session=MbQuhB5kwuUGupw8PbIo5RaCcM5uhGfW
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/113.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
Accept-Language: ru-RU,ru;q=0.8,en-US;q=0.5,en;q=0.3
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 23
Origin: https://0a30007203027e3e80f97b6600e600bd.web-security-academy.net
Referer: https://0a30007203027e3e80f97b6600e600bd.web-security-academy.net/login
Upgrade-Insecure-Requests: 1
Sec-Fetch-Dest: document
Sec-Fetch-Mode: navigate
Sec-Fetch-Site: same-origin
Sec-Fetch-User: ?1
Te: trailers

username=ads&password=$123$
```

Рис. 2.23 Процес підставлення знайденого логіну ads до параметру username

Тепер коли логін внесено, треба підібрати пароль для нього. Тому цілю підстановки даних тепер буде параметр “password”. Алгоритм дії точно такий же як і для логіну. Відкриваємо вкладку Payloads та додаємо словник можливих паролів, та запускаємо атаку. За довжиною відповіді від серверу ми також можемо знайти правильний пароль як це зображено на рис 2.24



Request	Payload	Status code	Error	Timeout	Length	Comment
7	1234	200			3240	
8	111111	200			3240	
9	1234567	200			3240	
10	dragon	200			3240	
11	123123	200			3240	
12	baseball	200			3240	
13	abc123	200			3240	
14	football	200			3240	
15	monkey	200			3240	
16	letmein	200			3240	
17	shadow	200			3240	
18	master	200			3240	
19	666666	200			3240	
20	qwertyuiop	200			3240	
21	123321	302			178	
22	mustang	200			3327	
23	1234567890	200			3327	

Request Response

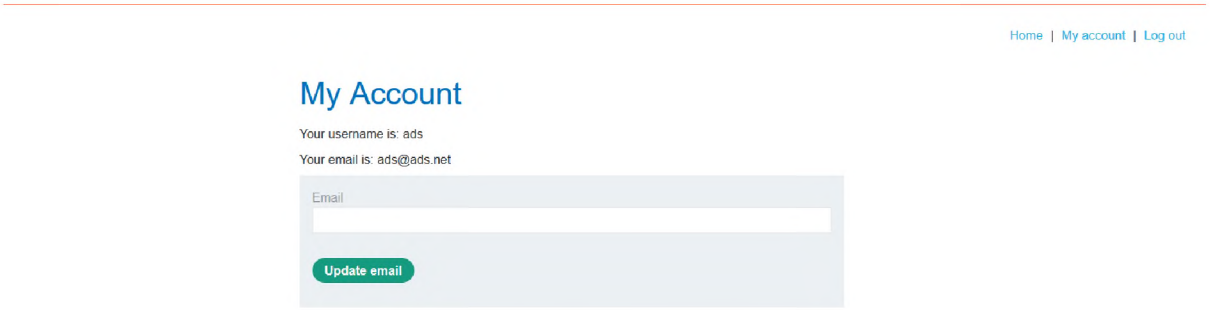
Pretty Raw Hex Render

```
1 HTTP/2 302 Found
2 Location: /my-account
3 Set-Cookie: session=AucNL2HWiQgU1XOVCE2i6DySYfbISXZU; Secure; HttpOnly; SameSite=None
4 X-Frame-Options: SAMEORIGIN
5 Content-Length: 0
6
```

Рис 2.24 Відповідь від сервера з іншою довжиною

Можна побачити довжину повідомлення, яка відрізняється від інших, та у самій відповіді, ми можемо побачити, що сторінка відкрилася у директорії /my-account, що є акаунтом користувача “ads”.

Давайте спробуємо увійти до акаунту використовуючи отримані дані, де логін є “ads” та пароль є “123321”, результат вхожу можна побачити на рис 2.25



2.25 Вдала спроба входу в акаунт який було отримано способом брут-форсу.

Отже, було отримано доступ до акаунту користувача та тепер змінити пошту, та отримати повний контроль над акаунтом є можливим. За допомогою легкого брут-форсу можна отримати достуи до акаунту будь-якого клієнта, якщо система автентифікації не була належним чином спроектована.

Деякі веб-додатки мають захист від брут-форс атак через блокування IP адрес, або через блокування акаунту на деякий час, щоби запобігти множині спроби брут-форсу. Це не ідеальний вид захисту, так як акаунт все одно може пасти жертвою брут-форсу через деякий час. Проте спроби брут-форсу з блокуванням акаунту після невдалих спроб може зайняти дуже багато часу. Уявіть, що після кожної третьої невдалої спроби акаунт буде блокуватися на 24 години, а брут-форс атака може потребувати перевірку десятків тисяч комбінацій, або навіть і сотні тисяч комбінацій. Щоб перебрати таким чином 1000 комбінацій, зловмисник отримає блокування акаунту 333 рази, що загалом займе 7992 години тільки щоби перечекати, коли блокування буде знято. Зазвичай, після таких спроб власнику акаунта можуть відсилати повідомлення, що до його акаунта намагаються увійти . Повідомлення буде відіслано тільки у тому випадку якщо реалізовано захист через блокування самого акаунту, до якого намагаються увійти. Якщо захист реалізовано через блокування IP адреси, то власник акаунта не дізнається, що його акаунт намагаються вкрасти.

Слід зауважити, що захист акаунта методом блокування IP адреси зловмисника не є надійним, бо в такому випадку, зловмисник може просто змінити свою IP адресу використовуючи проксі або VPN. Такий тип блоку може використовуватися з іншого боку, якщо такий блок буде виданий саме девайсу, який використовується для брут-форсу, через унікальний серійний номер. Проте, такі способи не э панацеєю, та їх можна обійти в залежності від того як спроектована система автентифікації.

Хоч взлам теоретично можливий, навіть якщо система автентифікації була захищена, проте процес взламу акаунту в такому разі для зловмисника може бути не доцільним з точки зори затраченого часу та отриманої користі. Ідеальної

системи захисту не існує, хоча на деяких сайтах через те, що логіка систем автентифікації спроектована неправильно, автентифікація може зовсім обходитися.

Під час наступного тесту буде показано один із випадків, коли невдалі спроби блокуються, але деяким способом лічильник невдалих спроб можна будуть скинути.

Тестове середовище веб-сайт другого тесту представлено на рис. 2.26

Рис. 2.26 Знімок екрану другого навчального середовища для пошуку вразливостей в системі автентифікації.

Загалом нічого особливого на новому полігоні немає. Знову основним плацдармом для атак буде вікно автентифікації (рис .2.27)

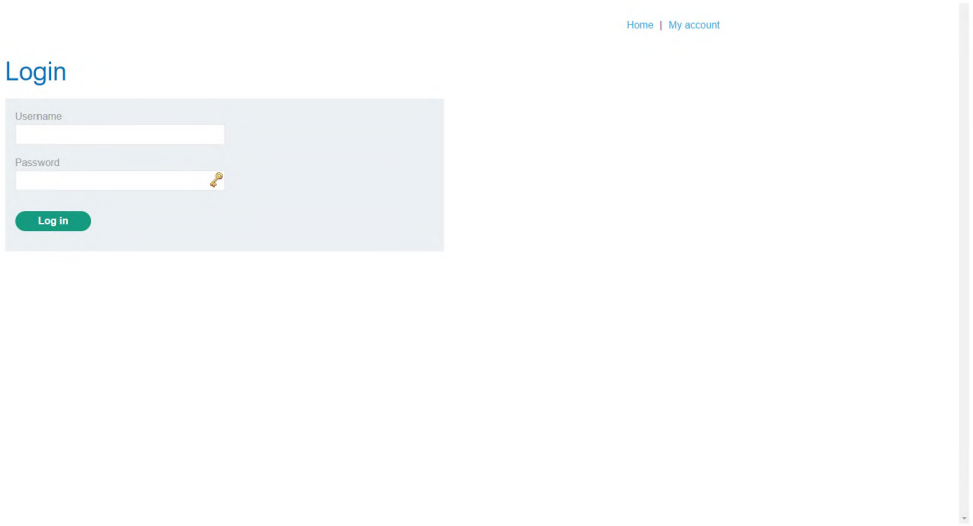


Рис. 2.27 Поле автентифікації другого тесту.

Результат спроб введення неправильних даних до форму входу зображено на рис. 2.28

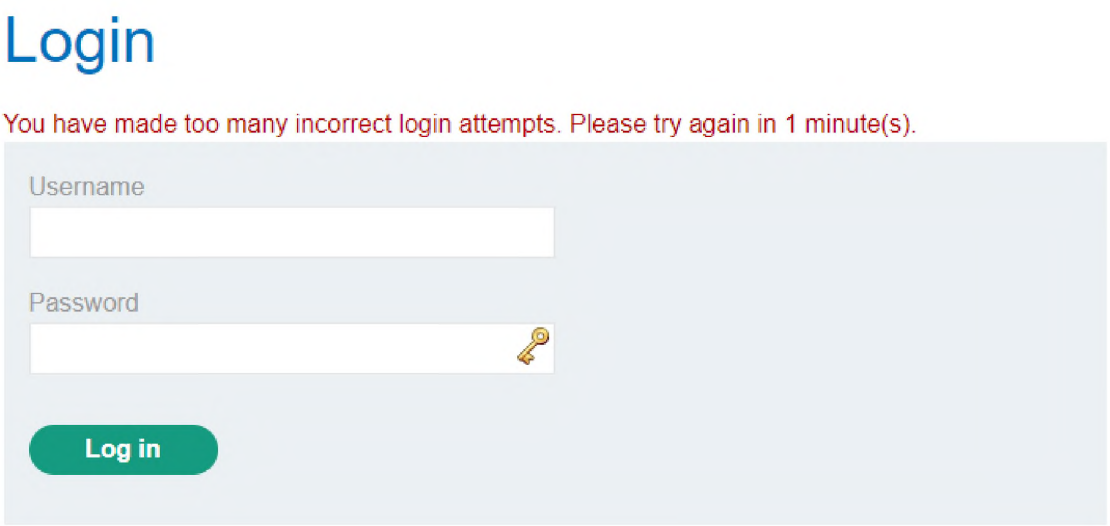


Рис. 2.28 Блокування акаунта після декількох невдалих спроб входу підряд

IP адреса була заблокована, якщо її змінити можливо продовжити брут-форс атаку. Проте, це дуже повільний та не автоматизований варіант, до того, на полігоні блокування дають лише на одну хвилину, коли на деяких веб-додатках дають до доби.

Спробуємо зайти до власного кабінету. Логін «wiener» та пароль «peter». Припустимо, що відоме ім'я користувача: «carlos», а також є словник можливих паролів. Проте, на цей раз недостатньо мати тільки логін та словник можливих паролів для брут-форсу. Треба продовжити збирати інформацію щодо конструкції системи автентифікації Відомо, що після декількох невдалих спроб IP адреса буде заблокована, але можна спробувати увійти до існуючого акаунту, до якого ми маємо доступ і подивитися як на це відреагує система.

Система відреагує наступнім чином, вона пустить нас до нашого акаунта:

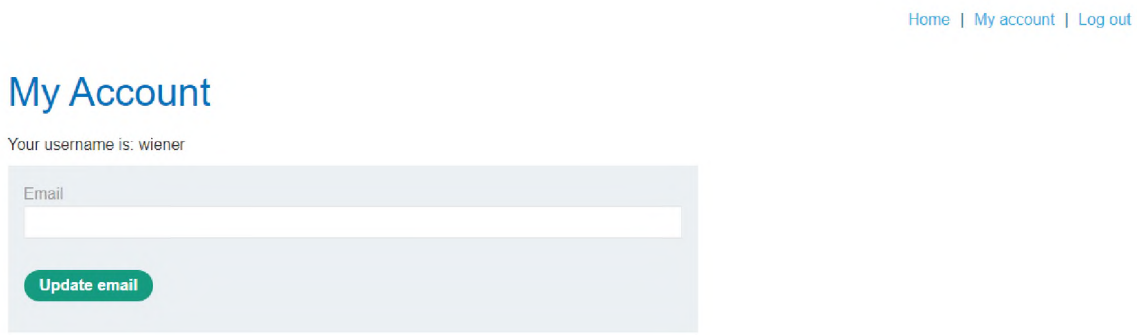
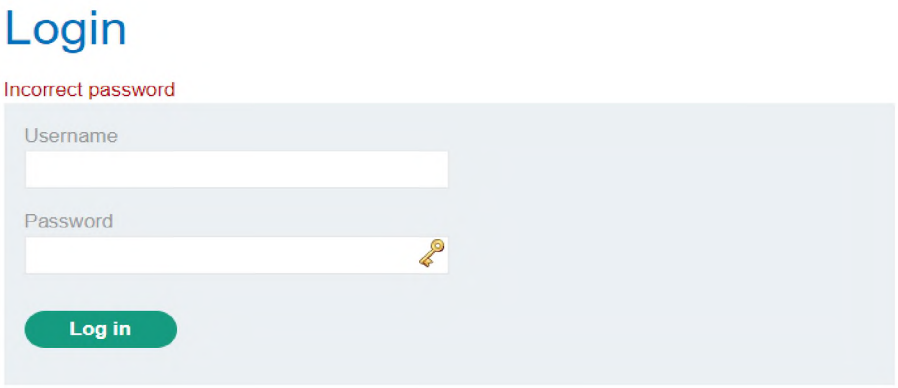


Рис. 2.29 Власний кабінет.

Доступ до валідного акаунту можемо спрацювати наступним чином: система зараховує це як успішну спробу та скидає лічильник невдалих спроб до 0, тобто,після входу до аканту лічильних повністю скидується до нуля та можна зробити ще 2-3 спроби перед блокуванням.

Отже після збирання інформації щодо нашої системи автентифікації на прикладі типового об'єкту, можна розробити план атаки на систему автентифікації, використовуючи Burp Suite.

Так як логін для входу в акаунт користувача відомий, через хибну логіку написання та проектування систем автентифікації, вона сама піказує, що логін вірний помилкою «Incorrect password», що також зображено на рис. 2.30:



2.30 Помилка «Incorrect password»

Помилка вказує саме на те, що невірний пароль до акаунту “carlos” було введено до існуючого акаунту. Ось такий результат буде, якщо ми спробуємо незареєстрований акаунт(рис. 2.31):

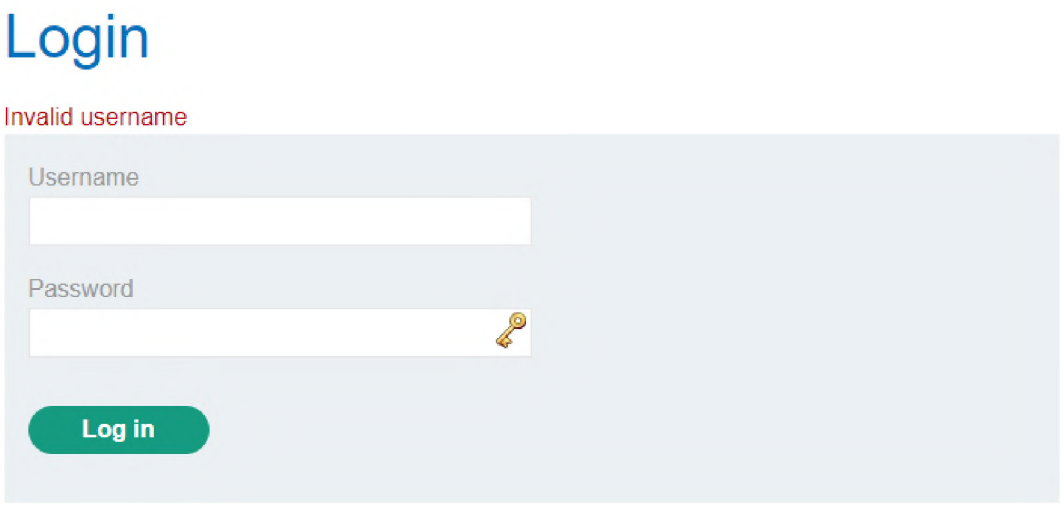


Рис. 2.31 Помилка «Invalid username».

Система нам вказує, що саме такий акаунт в системі не зареєстровано. Ось таким чином можна перевіряти в системах, чи зареєстрований акаунт в системі, для подальшого використання цієї інформації для планування атаки.

Для того, щоби виконати атаку на таку систему автентифікації потрібно розробити два словники до початку атаки. Вже є логін від акаунту до якого потрібно отримати доступ: carlos, та логін wiener, та є словник часто використовуваних паролів, який буде надано повним списком у додатку Ґ. Потрібно провести атаку так, щоби після кожної спроби входу (в незалежності від того вдала вона чи хибна), відбувався вхід до валідного акаунту для того, щоби збивати лічильник невдалих спроб до 0, що в свою чергу робиться для запобігання блокування на певний часовий період. Для цього знадобляться 2 нових словників як для логінів так і для можливих паролів. Все що потрібне, щоби після кожної спроби увійти до акаунту «carlos» с будь-яким можливим паролем із словника, а наступного разу була спроба увійти до валідного акаунту з ім'ям wiener та паролем peter. Щоби створити обидва списки, можна використати мову програмування Python для автоматизування процесу створення двох списків. Скільки паролів, стільки треба раз повторити ім'я користувачів, тобто, якщо 99 паролів, то треба повторити обидва логіни carlos та wiener 99 разів один за одним. Так само і у словнику з паролями, через один випадковий пароль, буде йти пароль від акаунту wiener. Приклади обох словників представлено у рис. 2.32 та 2.33

```
123456
peter
password
peter
12345678
peter
qwerty
peter
123456789
peter
12345
peter
1234
peter
111111
peter
1234567
```

Рис. 2.32 фрагмент з нового словника паролів

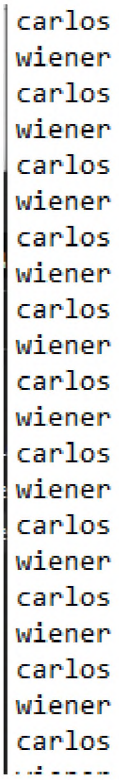


рис. 2.33 Фрагмент зі словника логінів

Повний код скрипту на мові програмування Python, можна бути знайти у додатку Б.

С отриманими словниками потрібно відкрити Burp Suite, та відіслати вкладку логіну до Burp Suite Intruder. Цього разу тип атаки треба змінити з “Sniper” на тип “PitchFork”. Відмінність цих двох атак полягає у тому, що “Sniper” використовується для підбору одного значення (або паролю або логіну), як у минулих тестах, де спочатку було знайдено валідний логін, а потім під цей логін підставлялися словники можливих паролів. Слід зауважити, що у минулому тесті не було тимчасового блокування IP адреси. Тобто “Sniper” може використовувати тільки один словник одночасно для того, щоби запустити атаку на веб-додаток, або у цьому випадку на наш типовий об’єкт. У той час Pitchfork для атаки використовує одразу два словника та підставляє перше значення с одного словника до першого значення с другого словника для чого і було

створено два нові словника. Отже, треба вибрати 2 параметри, до яких будуть додаватися словники, та для кожного додати відповідний словник. На рис 2.34 зображено нові позиції до яких додаватимуться словники:

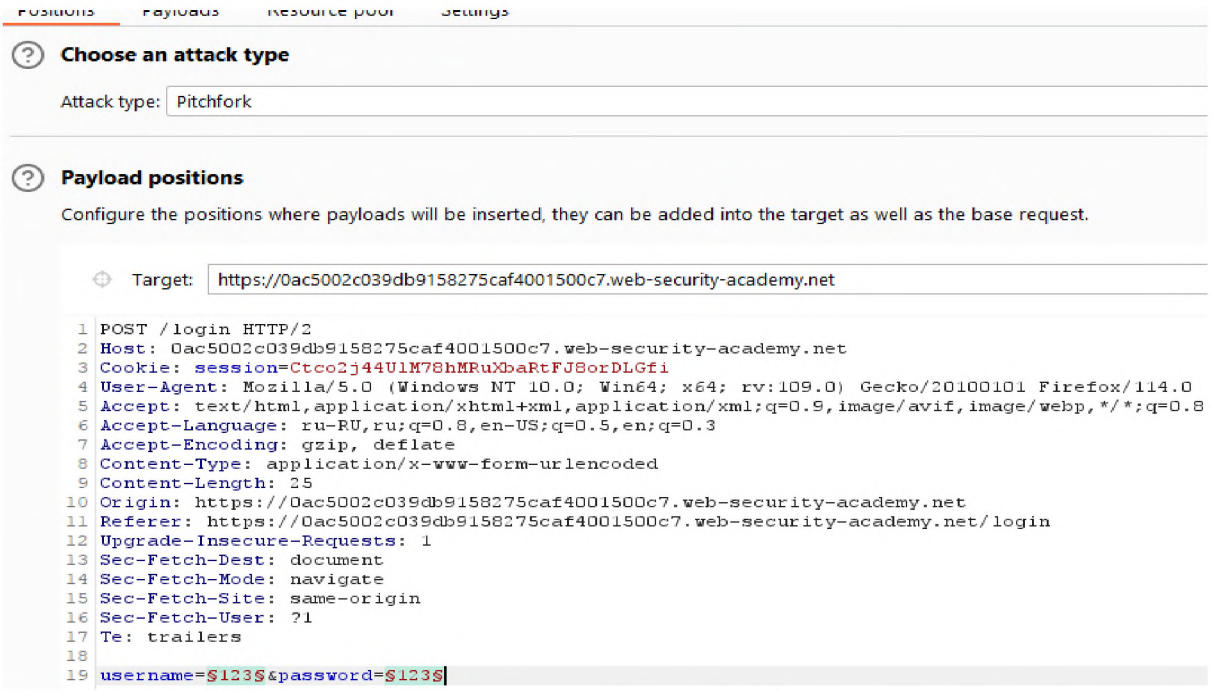


Рис. 2.34 Обрані параметри для підстановки словників

Цього разу будуть обрані обидва параметри логіну та паролю одночасно

Додаємо перший словник(рис. 2.35):

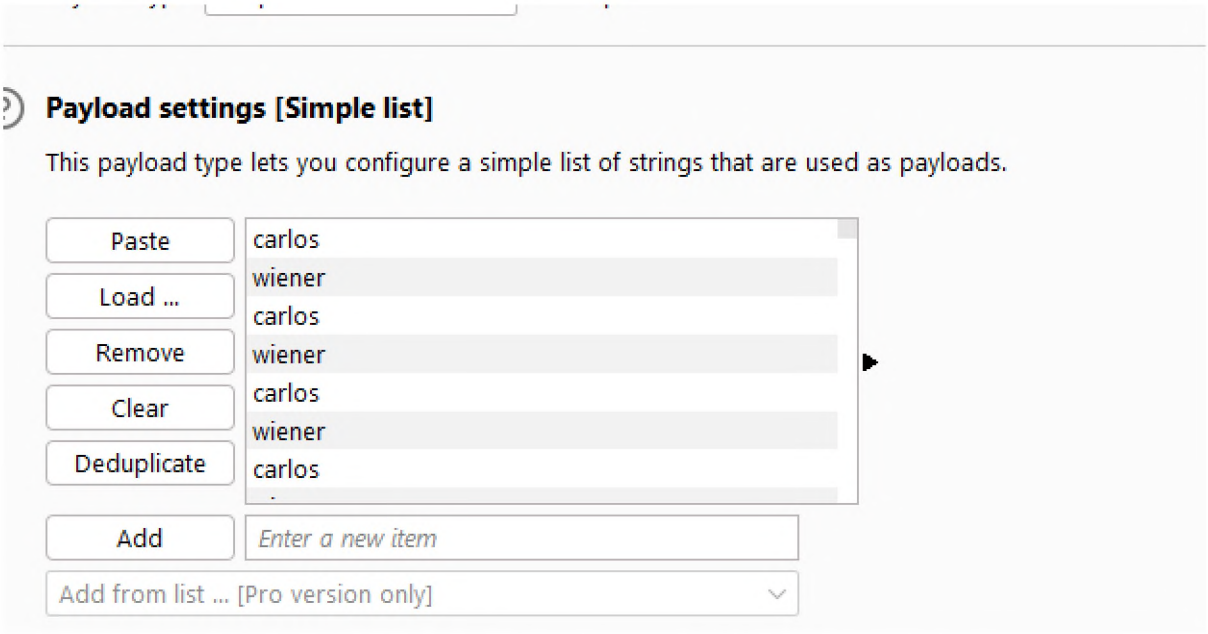


Рис. 2.35 Додавання словника логінів

Та додаємо другий словник з паролями(рис 2.36):

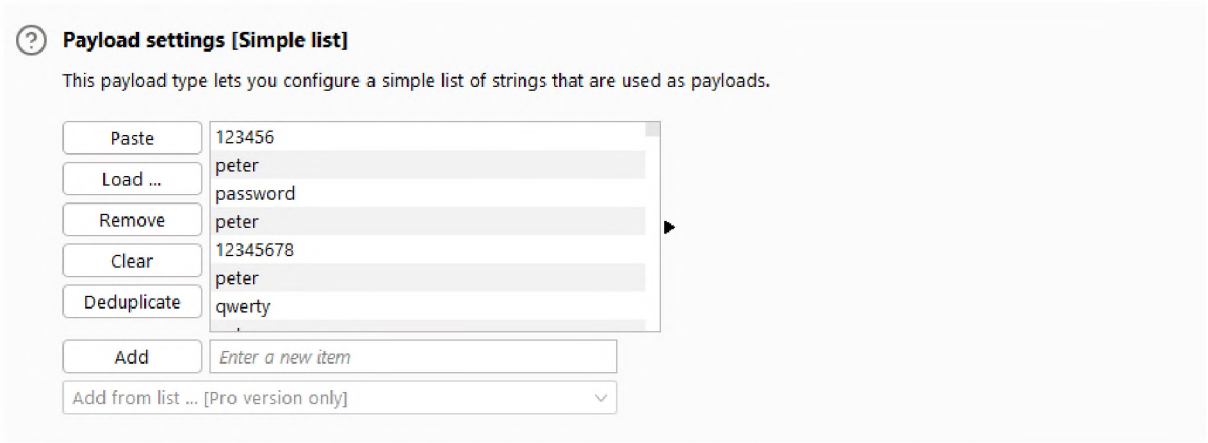


Рис. 2.36 Додавання словника з паролями

Після додавання обох словників треба почати атаку, та відстежити її результати, процес атаки зображено на рис 2.37:

Request	Payload 1	Payload 2	Status code	Error	Timeout	Length	Comment
4	wiener	peter	302			178	
5	carlos	12345678	200			3310	
6	wiener	peter	302			178	
7	carlos	qwerty	200			3310	
8	wiener	peter	302			178	
9	carlos	123456789	200			3310	
10	wiener	peter	302			178	
11	carlos	12345	200			3310	
12	wiener	peter	302			178	
13	carlos	1234	200			3310	
14	wiener	peter	302			178	
15	carlos	111111	200			3310	
16	wiener	peter	302			178	
17	carlos	1234567	200			3310	
18	wiener	peter	302			178	
19	carlos	dragon	200			3310	
20	wiener	peter	302			178	

Рис 2.37 Процес атаки.

Тут можна побачити різні довжини відповіді серверу на спроби логіну до акаунту “carlos” та “wiener”. Здебільшого представлені довжини 3310 та 178. Якщо подивитися на відповіді сервера на цих двох різних довжинах, то побачимо, що відповіді з довжиною 3310 це саме невдалі спроби увійти до акаунту, коли 178 це вдалі спроби (по більшій частині можна побачити таку довжину навпроти валідного акаунту з ім’ям “wiener”). С чого можна зробити висновок, що якщо під час атаки буде помічено довжину відповіді сервера 178 яке стосується акаунта “carlos” то це і буде потрібний пароль. Відповідь с довжиною 178 з акаунтом «carlos» можна побачити на рис 2.38

Request	Payload 1	Payload 2	Status code	Error	Timeout	Length	Comment
146	wiener	peter	302			178	
147	carlos	princess	200			3310	
148	wiener	peter	302			178	
149	carlos	joshua	200			3310	
150	wiener	peter	302			178	
151	carlos	cheese	200			3310	
152	wiener	peter	302			178	
153	carlos	amanda	200			3310	
154	wiener	peter	302			178	
155	carlos	summer	200			3310	
156	wiener	peter	302			178	
157	carlos	love	200			3310	
158	wiener	peter	302			178	
159	carlos	ashley	302			178	
160	wiener	peter	302			178	
161	carlos	nicole	200			3310	
162	wiener	peter	302			178	

2.38 Довжина відповіді 178 навпроти акаунта “carlos”

Знайдено відповідь сервера с потрібною довжиною, перейдемо до вкладки відповіді від сервера, та перевіримо, чи дійсно це те, що нам потрібно. Відповідь сервера зображено на рис. 2.39

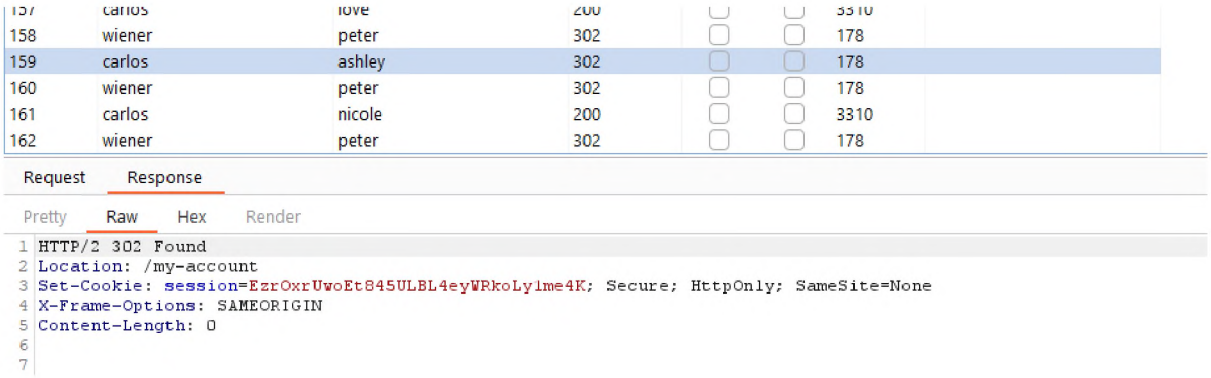


Рис. 2.39 Відповідь сервера

Можна побачити, що справді, цей запит переадресував нас до сторінки с самим акаунтом на ім’я “carlos“. На рис. 2.40 можна побачити, що до акаунту «carlos» було успішно отримано доступ.

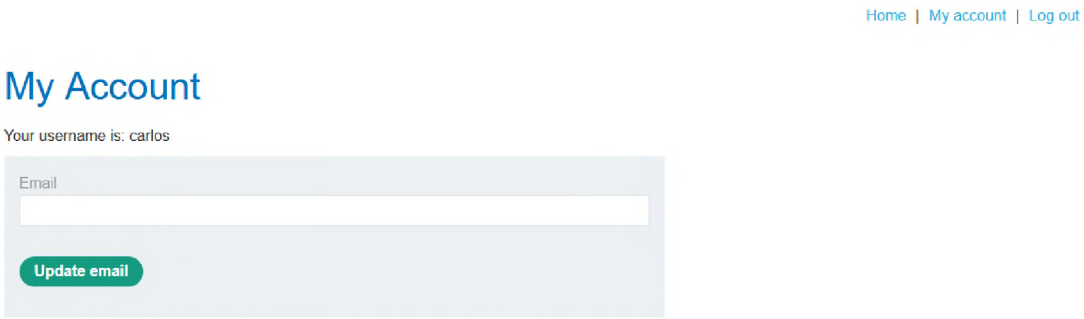


Рис. 2.40 Вдала спроба входу до акаунту “carlos”

Вразливість системи, через неправильно реалізовану логіку, була використана, несанкціонований доступ до акаунту було отримано. Є можливість змінити адресу електронної пошти цього акаунту, та перехопити повний доступ. Надалі, якщо це був би акаунт с великим рівнем доступу до внутрішніх систем

веб-додатку, можна було розгорнути нову атаку, умовно кажучи, цей акаунт можна використати як плацдарм, для нової атаки вже на самі системи веб-додатку.

2.3 Дослідження типиових вразливостей систем багатофакторної автентифікації.

Багатофакторні системи автентифікації також мають типові вразливості. Самі багатофакторні системи автентифікації у рази стійкіші до взламів на відміну від систем однофакторної автентифікації. Проте погано написана логіка може повністю зруйнувати всю стійкість системи автентифікації до того, що багатофакторність просто не буде перевірятися. Веб-додаток буде переводити до сторінки де треба буде пройти багатофакторну автентифікацію, проте іноді, в URL полі браузеру можна просто змінити параметр /login на параметр /my-account (Ці параметри можуть відрізнятися на інших веб-додатка), і можливо, що в такому разі можна буде отримати доступ до акаунту фактично без проходження автентифікації. Головна вразливість багатофакторної автентифікації є саме написання її логіки на рівні розробки.

2.3.1 Обходження багатофакторної автентифікації через хибне налаштування

Є типовий об’єкт (навчальне середовище Portswigger), який виглядає як стандартний веб-сайт блог. Є можливість входу до особистого кабінету. Для цього тесту в наявності є особистий кабінет логін:wiener та пароль:peter. Ці дані потрібні для входу до особистого кабінету, за допомогою якого в подальшому будуть виконуватися атака с несанкціонованим входом до акаунту carlos. Слід зауважити, що окрім вхідних даних від особистого кабінету wiener також надано ім’я користувача «carlos» та пароль «montoya». Неможливо увійти до цього акаунта використовуючи тільки комбінацію пароля та логіну, додатковий

верифікаціний код буде запитаний системою автентифікації, який повинен бути відправлений до поштової скриньки зв'язаною с акаунтом користувача.

Сутність цієї вразливості полягає у тому, що коли відбувається вхід до валдіного акаунту до якого ми маємо доступ, та водимо правильний автентифікаційний код від багатофакторної автентифікації, цей верифікаційний код зберігається до cookie, які в свою чергу використовуються для автентифікації користувача, та подальшої авторизації до акаунту.

Отже, щоби провести цю атаку, спочатку треба увійти до власного акаунту (wiener), та отримати автентифікаційний код для авторизації: Форма багатофакторної автентифікації зображено на рис. 2.41

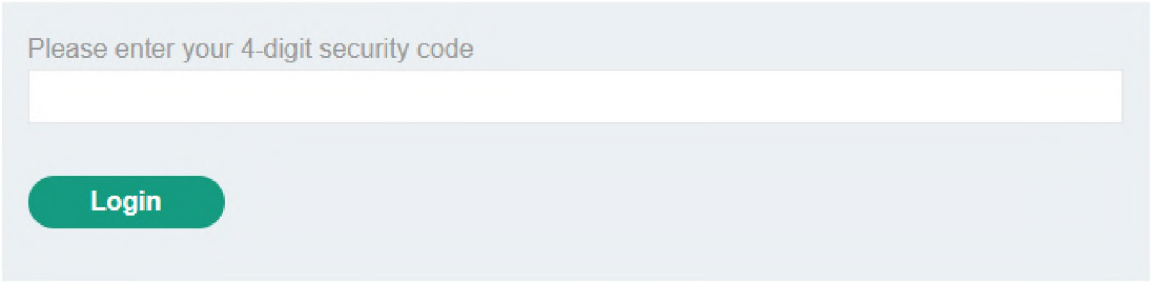


Рис. 2.41 Знімок екрану форми багатофакторної автентифікації

Тепер треба перейти до клієнту електронної пошти, та знайти верифікаційний код. Клієнт електронної пошти зображено на рис 2.42

Рис. 2.42 Клієнт електронної пошти

Увійдемо до акаунту використовуючи цей код. Коли доступ до акаунту було отримано в cookie загрузило наш верифікаційний код, який було сприйнято валідним, для поточної сесії. Тепер треба вийти з акаунту “wiener”, та спробувати увійти до акаунту “carlos”. Все одно буде запитано верифікаційний код. Проте, треба звернути увагу на URL поле браузера (рис. 2.43)

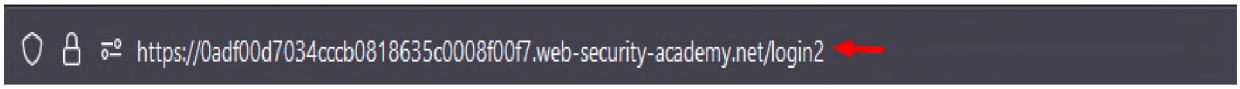


Рис. 2.43 URL поле браузера під час автентифікації.

Треба звернути увагу на останній параметр цього посилання, а саме /login2, до якого користувача після вдалої автентифікації переадресовує. Під час спроб ходу до валідного акаунту можна було побачити у URL полі параметр /my-account. Тож треба спробувати змінити параметр /login2 на параметр /my-account. Процес зображено на рис 2.44

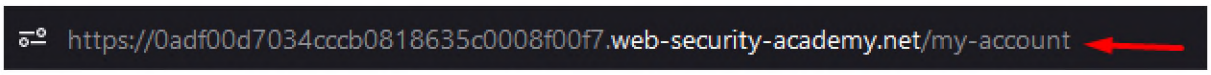


Рис. 2.44 Зміна параметру /login2 на /my-account ц

Після цієї зміни відбудеться переадресація до акаунту до якого були спроби увійти, а саме до «carlos», через те, що в cookie збереглася інформація що до верифікаційного коду, та він все ще вважається валідним. Ось результат нашої атаки(рис 2.45):

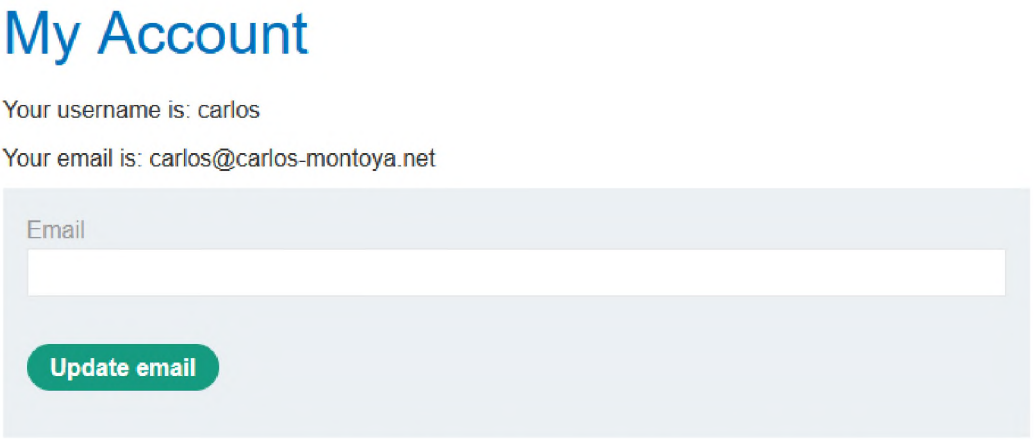


Рис. 2.45 Доступ до акаунту “Carlos”.

Тепер можна змінити пошту, та отримати повний доступ до акаунта користувача.

2.3.2 Зламана логіка багатофакторної автентифікації

У цьому тесті логіка системи автентифікації завчасно погано спроектована. Іноді до чужого акаунта можна увійти використовуючи POST та GET запити при вході до власного особистого кабінету. Таким чином можна згенерувати автентифікаційних код для чужого акаунту, який внаслідок можна бути отримати методом брут-форсу.

Спочатку треба зайти до власного акаунта. Власний акаунт має наступні облікові дані: логін - wienet, пароль - peter. Так як в цьому тесті працює багатофакторна автентифікація треба зайти до поштового клієнту та отримати автентифікаційни код.

В цьому тесті нам знову допоможе інструмент Burp Suite для виконання одразу декількох функцій. По-перше він знадобиться для перехвату POST та GET запитів до сервера, та відправки нових с модифікованими даними, по-друге, потрібно бути знайти валідний автентифікаційний код методом брут-форсу, тобто перебрати усі можливі комбінації кодів. Зрозуміти, коли ми знайдемо валідний

код наступним чином: Усі запити на сервер з невалідним автентифікаційним кодом буду мати статус 200 ОК, що означає, що запит просто був оброблений, але нікуди далі нас те не переадресує. Проте саме запит з валідним автентифікаційним кодом матиме статус відповіді 302 (що означає URL переадресування). Власне цей 302-й запит буде можливо відкрити в браузері використовуючи Burp Suite.

Через те, що дані в першому запиті на вхід до нашого особистого акаунту будуть модифіковані на дані іншого акаунту, можливо бути увійти без паролю, так як система буде вважати, що логін та пароль буде ведено правильно, та зробить запит на відправку автентифікаційного коду. В цьому випадку можна побачити, що батагофакторна автентифікація не просто не працює, а навіть руйнує автентифікацію ще на етапі комбінації логіну та паролю.

Отже, було отримано доступ до власного кабінету. Тепер треба відкрити запит GET до параметру /login2. Потрібний запит позначено на рис 2.46:

129	https://ac891f171f4cb196803f7...	GET	/login2		200	2894	HTML		2FA broken logic
130	https://ac891f171f4cb196803f7...	GET	/academyLabHeader		101	147			
131	https://ac781f8c1f7bb1b680dc7...	GET	/email		200	3915	HTML		Exploit Server: 2FA b
133	https://ac781f8c1f7bb1b680dc7...	GET	/resources/labheader/js/labHeader.js		200	851	script	js	
135	https://ac781f8c1f7bb1b680dc7...	GET	/resources/js/domPurify-2.0.15.js		200	17136	script	js	
136	https://ac781f8c1f7bb1b680dc7...	GET	/academyLabHeader		101	147			
137	https://ac781f8c1f7bb1b680dc7...	GET	/resources/labheader/images/logoAcad...		200	9134	XML	svg	
138	https://ac781f8c1f7bb1b680dc7...	GET	/resources/labheader/images/ps-lab-n...		200	897	XML	svg	
139	https://ac891f171f4cb196803f7...	POST	/login2	✓	302	178			
140	https://ac891f171f4cb196803f7...	GET	/my-account		200	3345	HTML		2FA broken logic
141	https://ac891f171f4cb196803f7...	GET	/academyLabHeader		101	147			

Рис. 2.46 GET запит до параметру /login2

Відкриємо вкладку запиту, для того, щоби на нього подивитися(рис. 2.47):

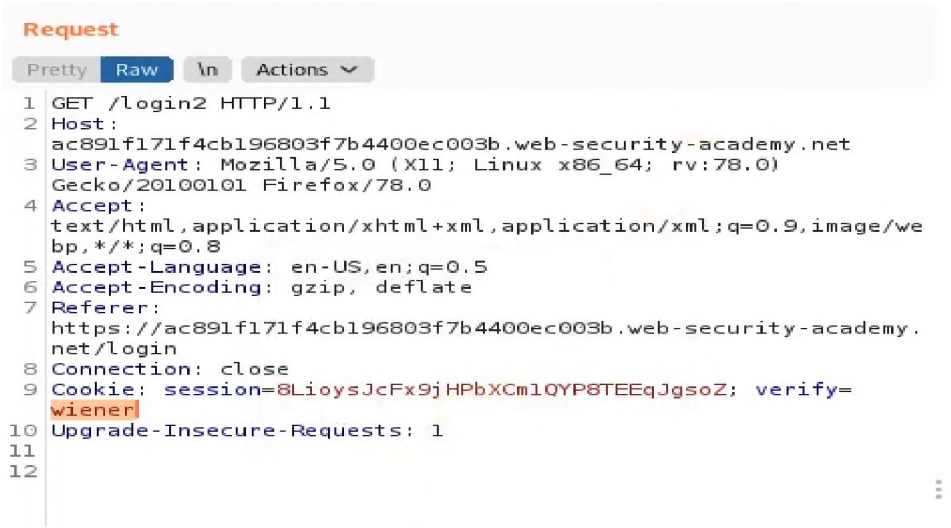


Рис. 2.47 Вкладка GET запиту до параметру /login2

Треба звернути увагу на змінну «verify=wiener», бо саме ця змінна визначає до якого акаунту буде надаватися доступ. Це GET запит треба додати до Burp Suite Repeater, для виконання подальшої атаки на автентифікаційну систему. Щоб додати цей запит до Burp Suite Repeater треба просто натиснути по цьому запиті правою кнопкою миші, та натиснути “Send to repeater”, майже так само як і додавати до Burp Suite Intruder.

Відкриймо Burp Suite Repeater та внесемо деякі зміни до запиту, а саме замість «verify=wiener», треба поставити «verify=carlos», у такому разі, вдала спроба авторизації за логіном та паролем буде переноситися з акаунта «wiener» до акаунту «carlos». Слід пам’ятати, що хоч пароль і не буде запитаний, все одно доведеться брут-форсити автентифікаційний код багатофакторної автентифікації. Проте, у цьому і у будь-якому іншому випадку, зазвичай, буде легше брут-форсити саме автентифікаційний код, тому, що переважно він складається лише з 4-6 цифр, на відміну від паролю, який може використовувати букви різних регистрів, цифри, спеціальні символи, для перебору комбінації яких можуть піти тижні, на відміну від декількох годин у випадку с числами. Зміну яку треба модифікувати зображено на рис 2.48.

Рис. 2.48 Модифікація змінної verify з нашого акаунту, на акаунт, до якого ми хочемо отримати доступ (“carlos”)

У вікні Burp Suite Repater треба натиснути кнопку “Send”, таким чином буде надіслано GET запит до серверу. Це потрібно зробити для того, щоби для акаунту “Carlos” згенерувався автентифікаційний код. Код повинен згенеруватися так як фактично відсилається запит вже с пройденим етапом автентифікації де потрібно зазначити пароль та логін. Генерація автентифікаційного коду потрібна для того, щоби була можливість його брут-форсу.

Треба ще раз увійти до власного акаунти використовуючи дані від валідного акаунту, але на етапі вводу автентифікаційного коду, вести неправильний автентифікааційний код, за для того, щоби відправити POST запит параметру /login2 до Burp Suite Intruder. POST запит параметру /login2 відображено на рис 2.49.

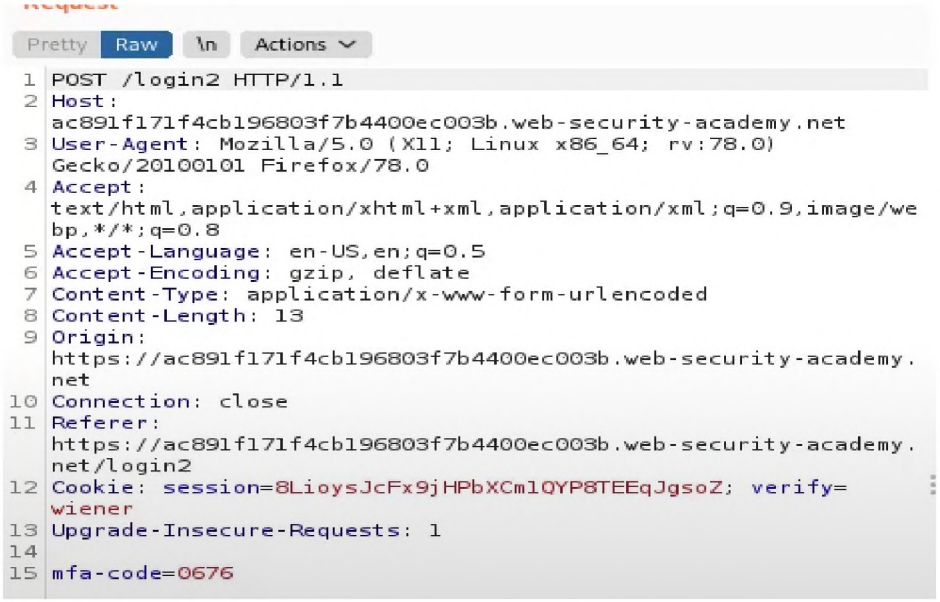


Рис. 2.49 Post запит параметру /login2

На рис. 2.49 можна побачити останню зміну “mfa-code=0676”, цю зміну у Burp Intruder треба взяти як позицію, до якої буде підставлятися значення згенеровані брут-форсером. Слід також змінити значення змінної “verify=wiener” на значення “verify=carlos”. Тип атаки, яка буде використана “Sniper”, у вкладці “Payloads”, треба змінити тип словнику зі звичайного списку на “Bruteforcer”, та вказати потрібні значення для генерації коду: Налаштування для атаки методом брут-форсеру представлені на рис. 2.50.

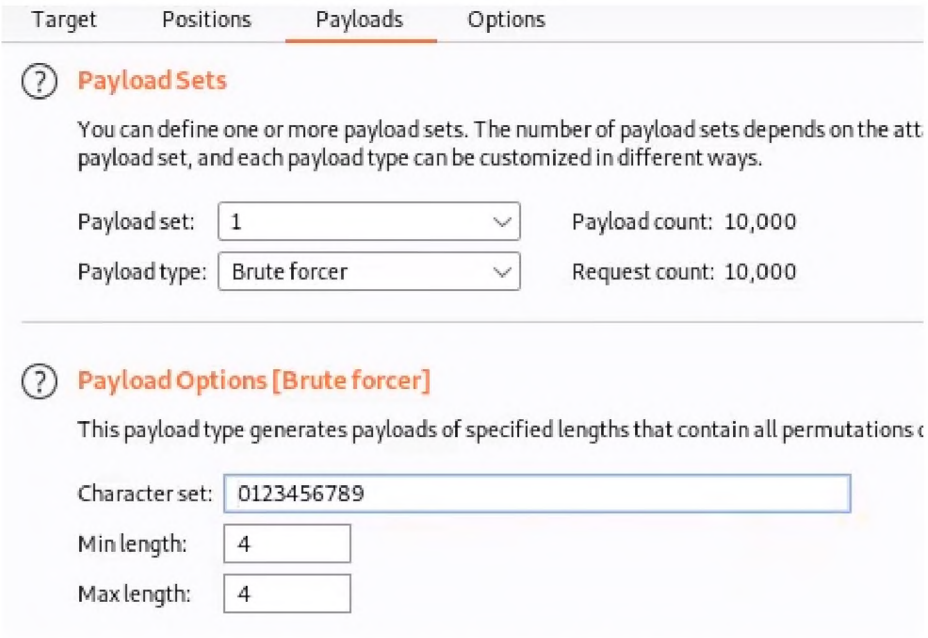


Рис. 2.50 Налаштування вкладки “Payloads” для атаки.

З такими налаштуваннями, брут-форсер буде генерувати будь-які можливі комбінації чотирьох значного коду, тому у цей раз не потрібен завчасно готовий словник можливих кодів. Процес запуску атаки представлено на рис. 2.51

Request	Payload	Status	Error	Timeout	Length
0		200			3057
1	0000	200			3057
2	1000	200			3057
3	2000	200			3057
4	3000	200			3057
5	4000	200			3057
6	5000	200			3057
7	6000	200			3057
8	7000	200			3057
9	8000	200			3057
10	9000	200			3057
11	0100	200			3057
12	1100	200			3057
13	2100	200			3057
14	3100	200			3057

Рис. 2.51 Процес атаки брут-форсером

Можна побачити, що відповіді для всіх неправильних кодів однакова - це відповідь 200 OK, що означає, що запит було оброблено, але ніякого при цьому переадресування до інших сторінок не відбулося, що може значити, що код невірний і доступу до акаунту не було отримано. На рис 2.52 можна побачити приклад потрібної відповіді від сервера.

Request	Payload	Status	Error	Timeout	Length
2961	0692	302	☐	☐	178
0		200	☐	☐	3057
1	0000	200	☐	☐	3057
2	1000	200	☐	☐	3057
3	2000	200	☐	☐	3057
4	3000	200	☐	☐	3057
5	4000	200	☐	☐	3057
6	5000	200	☐	☐	3057
7	6000	200	☐	☐	3057
8	7000	200	☐	☐	3057
9	8000	200	☐	☐	3057
10	9000	200	☐	☐	3057
11	0100	200	☐	☐	3057
12	1100	200	☐	☐	3057
13	2100	200	☐	☐	3057

Рис 2.52 Відповідь від серверу с потрібною довжиною (178 символів).

На рис 2.52 зображено, що на спробі 2961 була знайдена відповідь с автентифікаційним кодом 0692, яка має статус 302 (URL переадресація), та довжиною 178, що може свідчити про те, що необхідний код було знайдено та тепер можна увійти до акаунта користувача «carlos».

Для того щоби перевірити, чи дійсно доступ до акаунту було отримано, треба просто натиснути на цей запит, перейти до вкладку «Request», викликати контекстне меню, натиснувши правою кнопкою на мишці, та натиснути на «Show response in browser», скопіювати надане посилання, та перейти до нього відкривши нову вкладку браузера. Результат відкриття запиту у браузері зображено на рис. 2.53:

Рис. 2.53 Знімок екрану особистого кабінету користувача «carlos»

Таким чином було отримано доступ до акаунту “carlos”, не знадобився пароль від цього акаунту через те, що автентифікаційна система, а саме її частина де проходить батагофакторна автентифікація руйнує всю логіку.

2.4 Розробка методів тестування систем автентифікації на вразливості

2.4.1 Розробка інструкцій тестування систем однофакторної автентифікації

За результатами розділу 2.2 можна розробити наступні інструкції тестування систем однофакторної автентифікації:

1. Варто зробити акаунт з веб-додатком, систему автентифікації якого треба тестувати на вразливості. Власний акаунт допоможе знайти недоліки логічної структури системи автентифікації

2. При заході до власного акаунту треба звернути увагу на URL поле вашого браузеру. Якщо в цьому полі буде помітно, що останній параметр це ваше ім'я користувача, цілком можливо, що замінивши це ім'я на будь-яке інше валідне, можна буте отримати доступ до іншого акаунту без потреби вводити пароль.

3. Спробувати вести невалідне ім'я користувача, деякі системи автентифікаціїсамі можуть підказувати, чи зареєстроване це ім'я користувача або ні. Зазвичай, якщо логін не зареєстрований в системі можна побачити таку помилку "Invalid Username". Це допоможе знайти акаунт з я кого можна почати перші атаки.

4. Якщо логін було знайдено, треба спробувати використати якийсь пароль. Якщо саме логін правильний, але пароль ні, іноді системи автентифікації дають таку помилку: " Incorrect password"/ Отже залишається тільки підібрати пароль до акаунту, використовуючи будь-яку програму брут-форсу, або зробити це через Burp Suite Intruder, з використанням словників з найпопулярнішими паролями.

5. При невдалих спробах, система автентифікації може блокувати вашу IP адресу, заблокувати пристрій у своїй системі, або заблокувати акаунт, до якого були спробу отримати несанкціонований доступ, окрім цих блокувань, система може сповістити власника акаунта, щодо спроб отримання доступу до його акаунту третьої особою. У разі блокування IP адреси, можна скористатися VPN або проксі, в той час коли блокують саме девайс, або акаунт все набагато складніше, проте будь-яка з систем не ідеальна. Зі сторони веб-додатку можна зробити наступним чином: спробувати увійти до валдіного акаунту, потім вийти, та спробувати декілька разів увійти до чужого акаунту,

перед блокуванням знову увійти до свого акаунту, і спробувати ще декілька разів увійти до чужого акаунту. У разі якщо блокування не буде, можна зробити висновок, що кожну вдалу спробу входу до акаунту, лічильник невдалих спроб скидується до 0. Таким чином, якщо спочатку входити до валідного акаунту, а потім робити одну брут-форс спробу і знову увійти до валідного акаунту, можна брут-форсити акаунти користувачів без отримання блокування.

6. За допомогою Burp Suite Intruder відстежувати як сервер відповідає на будь-яку спроби вхожу до акаунтів. Можуть відрізнятися довжини відповіде, наприклад довжина 3310 символ, де якщо відкрити повну відповідь, можна побачити, що спроба не вдалася через неправильний пароль або логін, в свою чергу коли отримується відповідь 178 одну с сотні 3310 довжин, можна побачити, що відбулася переадресація до параметру веб-додатку /my-account, що свідчить про те, що доступ було отримано. Також відстежити чи було отримано доступ можна за статусом відповіді від серверу у том ж Burk Suite intruder. Якщо, припустимо, в усіх спробах можна бачити відповідь 200 OK, що свідчить про те, що сервер обробив запит, проте бачимо в результаті, що відповідь нам дала “Incorrect Password” чи “Invalid Username”, то якщо буде помічено лише один запит зі статусом 302 (URL переадресування, що зазвичай означає, що відповідь від сервера переадресовує користувача зі сторінки входу до її особистого кабінету), то це скоріше за все правильні дані для входу в акаунт.

Така загальна інструкція як можна тестувати системи однофакторної автентифікації. Слід зауважити, що існує інші способи дізнаватися логін та паролі, або отримувати доступ до акаунтів, проте ці атаки пов’язані з іншими типами вразливостей веб-додатків, такими як HTML та SQL ін’єкції, або погане

логічне налаштування бази даних. Хоч ці вразливості дозволяють увійти до чужого акаунту, вони не відносяться до вразливостей систем автентифікації, тому в даній кваліфікаційній роботі вони не розглядаються

2.4.2 Розробка інструкцій тестування багатофакторних системи автентифікації

На відміну від систем однофакторної автентифікації, не завжди потрібно знати пароль від акаунту, до якого потрібно отримати доступ, так як іноді, логіка багатофакторної автентифікації дозволяє повністю обійти кроки з вводом комбінації логіну та паролю. Проте, логін акаунта до якого треба отримати доступ, зазвичай, треба знати завчасно. Якщо немає логіну, можна скористатися пунктами 4 та 5 с розділу 2.4.1, отже, давайте перейдемо до інструкції:

1. Можна спробувати обійти багатофакторну автентифікацію наступним чином: знадобиться створити акаунт на веб-додатку для того, щоби була можливість обійти систему автентифікації. Налаштувати бататофакторну автентифікацію на цьому акаунті. Треба перевірити, чи залишається спеціальне cookie під час входу до акаунту. Слід пам’ятати, що цей спосіб потребує знання логіну та паролю від акаунту до треба увійти. Як можна дізнатися пароль та логін від акаунту для проходження однофактрної автентифікації можна дізнатися у розділі 2.4.1. Коли всі дані є, треба увійти до власного кабінету використовуючи будь-який валідний акаунт. Cookie с успішною спробою входу буде збережено. Це Cookie може використовуватися для входу до іншого акаунту без проходження багатофакторної автентифікації, так як через це cookie, вона буде вважатися пройденою. Все що залишається зробити, це вести комбінацію логіну та паролю, а коли з’явиться вікно батафакторної автентифікації, потрібно змінити параметр /login2 (на приклади типового об’єкту Portswigger) та замінити його на параметр особистого кабінету /my-account/, тоді система

дасть доступ без потреби вводити автентифікаційний код. Слід зауважити, що параметри на інших веб-дадатках можуть відрізнятися, тому попередній аналіз веб-додатку треба провести перед тим як починати атаку.

2. Якщо перший тип не спрацював треба увійти до свого аканту, який можна створити на платформі веб-додатку.
3. Проаналізувати GET запит при вході до акаунту. Тут можна побачити якусь зміну, яка відповідає за те, до якого акаунту буде надаватися доступ.
4. Якщо ця змінна була знайдена можна додати цей GET запит до Burp Suite Repeater, та змінити її значення с вашого акаунту на логін акаунту до якого треба отримати доступ. Таким чином до серверу надійде запит сгенерувати автентифікаційний код для акаунту до якого треба отримати доступ
5. Після того як код було згенеровано, треба увійти вести комбінацію пароля та логіну до дійсного акаунту, проте на моменті воду автентифікаційного коду, вести недійсний код.
6. Вод недійсного коду дає можливість отримати POST запит до серверу, де можна побачити дві ключові змінні, одна відповідає за те, до якого акаунту буде надаватися доступ, інша за верифікаційний коод
7. Цей POST запит треба додати до Burp Suite Intruder для того, щоби модифікувати змінну яка відповідає за те до якого акаунта буде надаватися доступ, на логін акаунту, до якого треба отримати доступ, а змінну яка відповідає за верифікаційний код треба взяти до позиції у Burp Suite Intruder, та розпочати брут-форс атаку, для знаходження валідного автентифіаційного кодую
8. Треба відстежувати результати роботи Burp Suite, та знайти валідний код. Зазвичай, після входу до акаунту відбувається URL переадресація до особистого кабінету (URL переадресація може мати 301 або 302 статус відповіді від серверу)
9. Відкрити відповідь серверу у браузері у результаті чого, доступ до аканту може бути отриманою. Слід зауважити, що в цьому випадку не потрібно

навіть знати пароль від акаунту до якого треба увійти. Хибна логіка автентифіккації руйнує перший етап автентифікації (комбінація логін + пароль).

2.5 Розробка рекомендацій для підвищення рівня захищеності систем автентифікації

1. Блокування акаунтів користувачів у системі веб-додатку. Блокування за IP адресою. або за девайсом не таке надійне. Окрім блокування акаунту, власника треба повідомити про причини блокування на пошти, яка стосується його акаунту.
2. Заборонити на прості паролі. Вводити підказки для людей, щодо стійкості паролю які во вводять. Створення обов'язкового мінімуму паролів за довжиною. Просити використовувати спеціальні символи для паролів Зобов'язати використовувати принаймні 2 поради з 3. Це допоможе захистити акаунти від брут-форсу. Це не ідеальний захист, проте на брут фор акаунтів с міцним паролем можуть піти тижні. С цим правилом добре буде працювати блокування акаунтів, що не дасть змогу зловмиснику пробувати багато комбінацій за одиницю часу.
3. Закликати експертів для написання логіки систем автентифікації.
4. Тестування систем автентифікації, перед тим як реалізувати веб-додаток до публічного доступу.
5. Щоби захистити веб-додаток від брут-форс атак, треба використовувати багатофакторну автентифікацію (хоча би лист до електронної пошти с верифікаційним кодом, якщо користувачі не хочуть підключати цю можливість, та отримувати автентифікаційний код до додатку)

6. Заборонити доступ до акаунту тим, хто не відправляв цей запит через форми логіну.

7. Для запобігання автоматизованих спроб автентифікації, можна додати CAPTCHA захист.
8. Чистити файли cookie після входу для акаунту, щоби не було можливостей їх використати для отримання доступу до іншого акаунту.
9. Лімітувати час за який може бути використано автентифікаційний код, щоби унеможливити, або суттєво зничити шкивдкість брут-форсу автентифікаційного коду.
10. Автентифікаційний код повиненм бути принаймні с 6 символів.

2.6 Висновок

В спеціальному розділі було охарактеризовано інструментарій, який використовувався для написана скриптів (Мова програмування Python) та для записку атак на веб-додатки (Burp Suite).

Проаналізовані типові вразливості різних систем автентифікації (однофакторна та багатофакторна системи автентифікації). За допомогою аналізу типових вразливостей було виконано практичні тести з використанням типових вразливостей.

За результатами практичного аналізу типових вразливостей систем автентифікації було розроблено декілька загальних інструкцій щодо тестування автентифікаційних систем. Завдяки отриманим результатам практичного аналізу були розроблені рекомендації, щодо підвищення рівня захищеності автентифікаційних систем.

Розділ 3 Економічний розділ.

3.1 Обґрунтування доцільності створення ефективних систем автентифікації.

Мета економічного розділу є аналіз доцільності створення ефективних систем автентифікації. Для цього потрібно зробити наступні розрахунки:

— капітальних витрат на придбання і налагодження складових систем автентифікації або витрат що пов'язані с купівлею необхідного обладнання, чи програмного забезпечення, найм спеціалістів.

—- Річних експлуатаційних витрат на утримання систем автентифікації та на їх обслуговування.

—- Річного економічного ефекту

— Освітлити можливі збитки у разі не реалізації ефективної системи автентифікації.

—- Показників економічної ефективності розробки та впровадження розробки ефективних систем автентифікації

3.2 Розрахунок (фіксованих) капітальних витрат

Капітальні інвестиції - це кошти, призначенні для створення і придбання основних фондів і нематеріальних активі, що підлягають амортизації.

До капітальних витрат належать витрати на проектування системи автентифікації

Визначити трудомісткість проектування системи автентифікації.

Трудомісткість проектування системи автентифікації визначається тривалістю кожної робочої операції, починаючи зі складання технічного завдання і закінчуючи оформленням документації (за умови роботи одного спеціаліста з інформаційної безпеки)

$$t = t_{тз} + t_в + t_а + t_{вз} + t_{озб} + t_{овр} + t_д, \text{ годин,} \quad (3.2.1)$$

де $t_{тз}$ - тривалість складання технічного завдання на розробку системи автентифікації.

$t_в$ – тривалість розробки концепції безпеки інформації для системи автентифікації;

$t_а$ – тривалість процесу аналізу ризиків;

$t_{вз}$ – тривалість визначення вимог до заходів, методів за засобів захисту;

$t_{озб}$ – тривалість вибору основних рішень з проектування системи автентифікації

$t_{овр}$ – тривалість організації виконання відновлювальних робіт і забезпечення неперервного функціонування

$t_д$ – тривалість документального оформлення політики безпеки

Визначено, що відповідно до етапів проектування системи автентифікації, тривалість операцій скаладє наступні величини: $t_{тз}$= 25 годин, $t_в$ = 40 годин, $t_а$ = 15 годин, $t_{вз}$ = 15 годин, $t_{озб}$ = 12 годин, $t_{овр}$ = 12 годин, $t_д$ = 6 годин.

Отже, t = 25 + 40 + 15 + 15 + 12 + 12 + 6 = 125 годин.

Розрахунок втрат на проектування системи автентифікації.

Витрати на проектування системи автентифікації $К_{рп}$ складаються з витрати заробітну плату спеціаліста з інформаційної безпеки $З_{зп}$ і вартості витрат машинного часу, що необхідний для розробки політики безпеки інформації Змч.

$$К_{рп} = З_{зп} + З_{мч} = 100000 + 1800 = 36500 \text{ грн} \quad (3.2.2)$$

$$З_{зп} = t * З_{пр} = 125 * 800 = 100000 \text{ грн} \quad (3.2.3)$$

де t - загальна тривалість проектування системи автентифікації в годинах

Вартість машинного часу для розробки системи автентифікації:

$$З_{мч} = t * С_{мч} = 1250 * 1.44 = 1800 \quad (3.2.4)$$

де $С_{мч}$ = вартість 1 години роботи ПК за планами тарифів за світло.

Відповідно до рекомендацій наданим найнятим спеціалістам, буде спроектована система автентифікації. За для проектування та тренуванню співробітників для проектування системи автентифікації знадобиться BuprSuite та система автентифікації буде проектуватися на базі веб-серверів Namecheap. Ціну повної роботи можна знайти наступним чином:

$$К = К_{рп} + К_{зпз} + К_{пз} + К_{аз} + К_{навч} + К_{н} \quad (3.2.5)$$

$К = К_{рп} + К_{зпз} + К_{пз} + К_{аз} + К_{навч} + К_{н} = 100000 + 2436,52 + 73000 + 5000 + 22000 + 2000 = 202636.52$ грн

де $К_{рп}$ – вартість проектування системи автентифікації

$К_{зпз}$ – вартість закупівель ліцензійного основного й додаткового програмного забезпечення, тис. грн.

$К_{пз}$ – вартість створення основного й додаткового програмного забезпечення

$К_{аз}$ – вартість закупівлі апаратного забезпечення та допоміжних матеріалів

$К_{навч}$ - витрати на навчання технічних фахівців і обслуговуючого персоналу

$К_{н}$ – витрати на встановлення обладнання та налагодження системи автентифікації

3.3 Розрахунок поточних витрат

Річні поточні витрати на функціонування системи автентифікації складають:

$$С = С_в + С_к + С_{ак} \quad (3.3.1)$$

де $С_в$ – вартість відновлення та модернізації системи

$С_к$ – витрати на керування системою автентифікації

$С_{ак}$ – витрати викликані активністю користувачів системи автентифікації ($С_{ак} = 0$)

Витрати на керування системою автентифікації ($С_к$) складають

$$С_к = С_н + С_а + С_з + С_{ел} + С_{тос} \quad (3.3.2)$$

Витрати на навчання адміністративного персоналу визначаються ($С_н$ = 30000, 3 тренінги на рік, кожен 10000 грн.)

Річний фонд заробітної плати фахівцям проекторам системи автентифікації, які будуть її обслуговувати складає

$$С_з = З_{осн} + З_{дод} \quad (3.3.3)$$

Основна заробітна плата визначається, виходячи з місячного посадового окладу, а додаткова заробітна плата в розмірі від 8% до 10% від основної заробітної плати.

Основна заробітна плата спеціаліста задіяного для проектування системи автентифікації складає 30000, додаткова заробітна плата - 10% від основної заробітної плати. Виконання роботи щодо налаштувань системи автентифікації потребу залучення спеціаліста на 0.25 ставки.

$$С_з = (30000 * 5 + 30000 * 0.1 * 5) * 0.25 = 41250 \text{ грн}$$

Ставка ЄСВ складає 22%:

$$С_{єв} = 41250 * 0,22 = 9075 \text{ грн}$$

Вартість електроенергії, що споживається апаратурою системи автентифікації протягом року (Сел):

$$С_{ел} = Р * F_р * Ц_е \quad (3.3.4)$$

де Р – встановлена потужність апаратури системи автентифікації (1 кВт)

$F_р$ – річний фонд робочого часу системи автентифікації (8760 годин)

$Ц_е$ – тариф на електроенергію ($Ц_е$ = 1,44 грн/кВт за годину)

Отже вартість енергії, що споживається апаратурою системи автентифікації становить:

$$С_{ел} = 1 * 8760 * 1,44 = 12614,4 \text{ грн}$$

Витрати на технічне й організаційне адміністрування та сервіс системи автентифікації визначаються у відсотках від вартості капітальних витрат - %1 ($С_{тос} = 202636.52 * 0.01 = 2026,3$ грн)

Витрати на керування системою автентифікації ($С_к$) визначаються:

$$С_к = 12614.4 + 9075 + 41250 + 2026,3 + 30000 = 94965,3 \text{ грн}$$

Таким чином, річні поточні витрати на функціонування системи автентифікаціїскладають 94965,3 грн

3.4 Аналіз можливих збитків у разі нереалізації міцної системи автентифікації

Аналіз буде проводитися на прикладі реальних компаній, які стали жертвами атак на системи автентифікації, внаслідок чого несли великі матеріальні та репутаційні втрати.

Інцидент Dunkin' Donuts, який стався у 2015 році. В цій крупній атаці були використані вразливості системи автентифікації, а саме був використан метод брут-форсу за для отримання облікових даних користувачів. Атака запустилася через отриману базу даних логінів, а далі брут-форсилися самі паролі від акаунтів. У разі атаки було вкрадено 19,715 акаунтів, та с акаунтів у веб-додатку було вкрадено тисячі доларів. Результатом окрім втраченої лояльності клієнтів, також стали збитки розміром 650000 доларів. Це змусило компанію вкласти кошти для удосконалення системи автентифікації, та політики паролів.

20 мільйонів акаунтів було вкрадено на веб-додатку Alibaba, хоча і не відомо які матеріальні збитки понесла компанія, проте порівнюючи с минулим прикладом, де 19000 акаунтів вийшли у більше ніж півмільйона доларів, зрозуміло, що сума може перевищувати мільярд. Атака відбулася 2016 року, як результат всіх користувачів веб-додатку попросили змінити паролі, для забезбечення безпеки акаунтів.

Слід зауважити, що було перераховано великі та відомі компанії, у яких вистачає гроші для того, щоби найняти спеціалістів та удосконалити свою систему автентифікації, проте якщо справа йде про малі бізнеси, такі атаки можуть не тільки спричинити великі матеріальні втрати, але й привести до закриття компанії.

Ticketmaster. На початку 2021 року було помічено, що один з працівників використовував свою посаду, для того щоби заходити до чужих акаунтів. Так як в нього був доступ до облікових даних від багатьох акаунтів, вони були використані для входу. Тут немає ніякого брут-форсу, системи автентифікації не було певним чином захищена, та не мала багатофактрнох автентифікації, результатом чого

Ticketmaster довелося заплатити близько 10 мільйонів доларів через цю вразливість, яка була вдало використана зі середина компанії.

Слід зауважити, що не завжди компанія переживає саме матеріальні збитки через вдалу атаку на систему автентифікації. Хакери можуть просто зібрати усю конфеденційну інформацію з акаунта та виставити її на продаж у так званому “Dark Net”, де зазвичай продається багато нелегальних речей, враховуючи вкрадені дані.

Таким чином якоїсь загальної інформації запропонувати щодо збитків неможливо, бо атаки на системи автентифікації слід розглядати у кожному випадку індивідуально. Іноді є тільки репутаційні збитки, іноді компанії встигають закрити вразливість нульового дня, іноді хакери крадуть не тільки дані але й гроші, якщо у веб-додатку вони були, або була прив’язана кредитна картка. Якщо компанія не понесе прямих матеріальних збитків від атаки, вона може отримати наслідкові збитки, коли клієнти через втрату лояльності та довіри після атаки, почнуть відмовлятися користуватися сервісом.

3.5 Розрахування ефекту від впровадження захисту систем автентифікації

Розрахувати ефект від провадження захисту систем автентифікації можна за наступною формулою:

$$E = B * R - C. \quad (3.4.1)$$

Де B це загальний збиток від атаки на систему автентифікації

R – очікувана імовірність атаки на систему автентифікації

C – щорічні витрати на експлуатацію системи автентифікації

Річні витрати на експлуатації систем автентифікації було визначено за допомогою формули 3.3.2 , та є C = 94965.3 грн.

Ефект від впровадження захисту систем автентифікації буде розібрано на прикладі Dunkin' Donuts. Виходячи з цього прикладу очікуваним збитком буде B=650000 доларів або 24020457,50 гривень, та вирогідність вдалої атаки R=5%, отже,

E = B * R – C = 24020457,50 * 0.05 – 94965,3 = 1106057,58 грн.

Завдяки отриманим розрахункам можна прорахувати коефіцієнт повернення інвестицій ROSI. Цей коефіцієнт показує скільки гривень додаткового прибутку приносить одна гривня капітальних інвестицій на впровадження захисту систем автентифікацій. Коефіцієнт рахується за допомогою наступної формули:

$$ROSI = \frac{E}{K}, \text{частки на одиниці.} \quad (3.4.2)$$

Де E – загальний ефект від впровадження захисту систем автентифікації.

K – капітальні інвестиції.

$$ROSI = \frac{E}{K} = \frac{1106057,58}{202636.52} = 5.46.$$

Також з отриманими даними можна визначити термін окупності. Термін окупності капітальних інвестицій T_0 показує за скільки років капітальні інвестиції окупляться за рахунок загального ефекту від впровадження захисту для систем автентифікації. Визначити термін окупності можна за наступною формулою:

$$T_0 = \frac{K}{E} = \frac{1}{ROSI}. \quad (3.4.3)$$

$$T_0 = \frac{202636,52}{1106057,58} = 0,18 \text{ року} = 2 \text{ місяці}$$

3.6 Висновок:

Під час виконання економічного розділу було розраховано капітальні витрати на проектування ефективної системи автентифікації, для захисту від типових вразливостей. Були розраховані витрати на персонал, обладнання, програмне забезпечення, витрати на експертів для проектування систем захисту.

Окрім капітальних витрат, були розраховані річні витрати на підтримування функціональності систем захисту та обладнання. Підготовку персоналу для роботи с цими системами, та періодичне навчання.

Були розглянуті потенціальні збитки у разі вдалої атаки на систем автентифікації на прикладі відомих компаній.

За отриманими даними було розраховано ефект від впровадження ефективних методів захисту систем автентифікації, окрім цього було визначено коефіцієнт повернення інвестицій ROSI та термін окупності впровадження ефективної системи автентифікації

Враховуючи всі розрахунки можна сказати, що є доцільним проектування ефективною та надійної системи автентифікації є доцільним, так як коефіцієнт повернення інвестицій дорівнює майже 6 одиницям за одиницю капітальних витрат, окупність проектування захисту наступить лише за 2 місяці, що є швидко порівняно с тим, які потенційні втрати може понести компанія, у разі ігнорування захисту систем автентифікації.

Висновки

Загрози автентифікації та систем автентифікації ще багато часу будуть актуальними, розробити ідеальний захист неможливо. В будь-якій системі завжди знайдеться вразливість, яка може с технічної точки зору бути використана складніше чи простіше, проте наслідки для компанії будуть одні і ті самі в разі вдалої атаки. Більше цього, кількість атак с кожним роком буде збільшуватися, через те, що багато фізичних компаній переходять до інтернет магазинів. Типові атаки на системи автентифікації не важко організувати, особливо якщо є дані про структуру, та проектування систем автентифікації. Не дивлячись на простоту атак, системи автентифікації можна ефективно захистити від типових вразливостей. Саме практичне тестування надає можливість фахівцям кібербезпеки думати як хакер, щоби в подальшому розробити на результатах практичних тестів ефективну систему захисту.

Як результат практичних тестів, було розроблені покрокові інструкції як можна тестувати системи автентифікації, та на що в першу чергу треба звернути увагу під час процесу. Разом із інструкцією тестування, були розроблені рекомендації, щодо підвищення рівня захисту систем автентифікації загалом, не тільки на прикладі типового об’єкту.

Слід пам’ятати, що проектування ефективної системи автентифікації, враховуючи типові вразливості зі створенням захисту від них, набагато дешевше та доцільніше, аніж ліквідовувати наслідки вдалої атаки методом брут-форсу. Найняти спеціалістів з кібербезпеки буде правильним рішенням, для створення надійного захисту, та дешевше аніж зустрітися с наслідками незахищеної системою автентифікації.

В економічному розділі було розглянуто питання доцільності проектування ефективних систем захисту, та за результатами розрахунків було визначено, що капітальні втрати на проектування окупляться вже за 2 місяці функціонування

систем захисту, через те, що вдасться уникнути втрати даних, чи зламу акаунтів користувачів.

Перелік посилань

1. Методичні рекомендації до виконання кваліфікаційних робіт бакалаврів спеціальності 125 кібербезпека / Упоряд.: О.В. Герасіна, Д.С. Тимофєєв, О.В. Кручінін, Ю.А. Мілінчук – Дніпро: НТУ «ДП» 2020.
2. Ідентифікація, автентифікація та авторизація — як не передати керування доступами зловмисника (Електрон. ресурс) / Ідентифікація: URL: https://www.linkedin.com/pulse/%25D1%2596%25D0%25B4%25D0%25B5%25D0%25BD%25D1%2582%25D0%25B8%25D1%2584%25D1%2596%25D0%25BA%25D0%25B0%25D1%2586%25D1%2596%25D1%258F-%25D0%25B0%25D0%25B2%25D1%2582%25D0%25B5%25D0%25BD%25D1%2582%25D0%25B8%25D1%2584%25D1%2596%25D0%25BA%25D0%25B0%25D1%2586%25D1%2596%25D1%258F-%25D1%2582%25D0%25B0-%25D0%25B0%25D0%25B2%25D1%2582%25D0%25BE%25D1%2580%25D0%25B8%25D0%25B7%25D0%25B0%25D1%2586%25D1%2596%25D1%258F-%25D1%258F%25D0%25BA-%25D0%25BD%25D0%25B5-%25D0%25BF%25D0%25B5%25D1%2580%25D0%25B5%25D0%25B4%25D0%25B0%25D1%2582%25D0%25B8-/?trackingId=ovpfKNPbjSCYfMDoxXTHLw%3D%3D
3. Portswigger (Електрон. ресурс) / Authentication vulnerabilities: URL: https://portswigger.net/web-security/authentication
4. Portswigger[1] (Електрон. ресурс) / Portswigget academy: URL: https://portswigger.net/
5. NIST SPECIAL PUBLICATION 1800-17 (Електрон. ресурс) / Multifactor Authentication for E-Commerce: URL: https://www.nccoe.nist.gov/publication/1800-17/
6. Wordpress brute-force protection documentation (Електрон. ресурс) / Brute force attacks: URL: https://wordpress.org/documentation/article/brute-force-attacks/

7. Adobe authentication systems protection documentation (Електрон. ресурс) / Above Commerce Security Best Practices: URL: https://www.adobe.com/content/dam/cc/en/trust-center/ungated/whitepapers/experience-cloud/adobe-commerce-best-practices-guide.pdf
8. Verizon (Електрон. ресурс)/ Data Breach Investigations report: URL: https://www.verizon.com/business/en-gb/resources/reports/dbir/
9. Hive Systems (Електрон. ресурс) / Password entrope: URL: https://www.hivesystems.io/blog/are-your-passwords-in-the-green
10. Strongdm (Електрон ресурс) / What is a brute force attack?: URL: https://www.strongdm.com/blog/brute-force-attack#:~:text=Examples%20of%20Brute%20Force%20Attacks,-Dunkin'%20Donuts%20pays&text=In%20a%20famous%202015%20incident,and%20ran%20brute%20force%20algorithms.
11. Makeuseof (Електрон ресурс) / 5 Times Brute Force Attacks Lead to Huge Security Breaches: URL: https://www.makeuseof.com/brute-force/
12. Ministry of Education and Science of Ukraine // The National Metallurgical Academy of Ukraine, Dnipro // Ідентифікація вразливостей процесів автентифікації
13. OBERLO (Електрон ресурс) / How Many People Shop Online?: URL: https://www.oberlo.com/statistics/how-many-people-shop-online
14. An Efficient Multifactor Authentication System[1] / researchgate (Електрон. ресурс): URL: https://www.researchgate.net/publication/371110271_An_Efficient_Multifactor_Authentication_System
15. Efficient Authentication Mechanism for Defending Against Reflection-Based Attacks on Domain Name System[2] / researchgate (Електрон. ресурс): URL: https://www.researchgate.net/publication/342654112_Efficient_Authentication_

Mechanism_for_Defending_Against_Reflection-Based_Attacks_on_Domain_Name_System

16. Security Authentication Mechanism of Spatio-Temporal Big Data Based on Blockchain[3] / researchgate (Електрон. ресурс): URL: https://www.researchgate.net/publication/371187322_Security_Authentication_Mechanism_of_Spatio-Temporal_Big_Data_Based_on_Blockchain
17. Тезаурус з кібербезпеки[4] (Електрон. ресурс) / Автентифікація та Авторизація: URL: https://yubikey.com.ua/tezaurus-z-kiberbezpeky.

Додаток А. Повний код відповіді на запит входу з неправильними даними

```
HTTP/2 200 OK
Content-Type: text/html; charset=utf-8
X-Frame-Options: SAMEORIGIN
Content-Length: 3130


<!DOCTYPE html>
<html>
    <head>
        <link href=/resources/labheader/css/academyLabHeader.css
rel=stylesheet>
        <link href=/resources/css/labs.css rel=stylesheet>
        <title>Username enumeration via different responses</title>
    </head>
    <body>
        <script src="/resources/labheader/js/labHeader.js"></script>
        <div id="academyLabHeader">
            <section class='academyLabBanner'>
                <div class=container>
                    <div class=logo></div>
                        <div class=title-container>
                            <h2>Username enumeration via different
responses</h2>
                            <a class=link-back
href='https://portswigger.net/web-security/authentication/password-
based/lab-username-enumeration-via-different-responses'>

Back to lab description 
                                <svg version=1.1 id=Layer_1
xmlns='http://www.w3.org/2000/svg'
xmlns:xlink='http://www.w3.org/1999/xlink' x=0px y=0px viewBox='0 0 28 30'
enable-background='new 0 0 28 30' xml:space=preserve title=back-arrow>
                                    <g>
                                        <polygon points='1.4,0 0,1.2
12.6,15 0,28.8 1.4,30 15.1,15'></polygon>
                                        <polygon points='14.3,0 12.9,1.2
25.6,15 12.9,28.8 14.3,30 28,15'></polygon>
                                    </g>
                                </svg>
                            </a>
                        </div>
```

```
                        <div class='widgetcontainer-lab-status is-
notsolved'>
                            <span>LAB</span>
                            <p>Not solved</p>
                            <span class=lab-status-icon></span>
                        </div>
                    </div>
                </section>
            </div>
            <div theme="">
                <section class="maincontainer">
                    <div class="container is-page">
                        <header class="navigation-header">
                            <section class="top-links">
                                <a href=/>Home</a><p>|</p>
                                <a href="/my-account">My account</a><p>|</p>
                            </section>
                        </header>
                        <header class="notification-header">
                        </header>
                        <h1>Login</h1>
                        <section>
                            <p class=is-warning>Invalid username</p>
                            <form class=login-form method=POST action="/login">
                                <label>Username</label>
                                <input required type=username name="username">
                                <label>Password</label>
                                <input required type=password name="password">
                                <button class=button type=submit> Log in
</button>
                            </form>
                        </section>
                    </div>
                </section>
                <div class="footer-wrapper">
                </div>
            </div>
        </body>
</html>
```

Додаток Б. Скрипт на Python для створення словників

```
arr_1 = **


my_password = "peter"
victim_username = "carlos"
my_username = "wiener"
file = open('file.txt', 'w')
file2 = open('file2.txt', 'w')
for i in aboba:
    file.write(i)
    file.write('\n')
    file.write(my_password)
    file.write('\n')
file.close()


file2 = open('file2.txt', 'w')
for i in range(len(arr_1)):
    file2.write(victim_username)
    file2.write('\n')
    file2.write(my_username)
    file2.write('\n')
file2.close
```

Додаток В. Відомість матеріалів кваліфікаційної роботи

№	Формат	Найменування	Кількість листів	Примітка
1	А4	Реферат	2	
2	А4	Зміст	2	
3	А4	Вступ	1	
4	А4	Розділ 1	20	
5	А4	Розділ 2	47	
6	А4	Розділ 3	9	
7	А4	Висновки	2	
8	А4	Перелік посилань	3	
9	А4	Додаток А	2	
10	А4	Додаток Б	1	
11	А4	Додаток В	1	
12	А4	Додаток Г	1	
13	А4	Додаток Д	1	
14	А4	Додаток Е	1	

Додаток Г Перелік документів на оптичному носії

1. Презентація Мірошник_АЮ_125_19_1_КВ_РОБ.pptx
2. Мірошник_АЮ_125_19_1_КВ_РОБ.docx
3. Мірошник_АЮ_125_19_1_КВ_РОБ.pdf
4. Мірошник_АЮ_125_19_1_КВ_РОБ.pdf.p7s

Додаток Д. Відгук керівника економічного розділу

Економічний розділ виконаний відповідно до вимог, які ставляться до кваліфікаційних робіт, та заслуговує на оцінку 90 б. («відмінно»).

Керівник розділу _____ Пілова Д.П.

(підписк) (прізвище, ініціали)

Додаток Е. Відгук керівника кваліфікаційної роботи

В І Д Г У К

на кваліфікаційну роботу студентки групи 125-19-1
Мірошника Артема Юрійовича
на тему: «Методи тестування вразливостей в системах автентифікації**»**

Пояснювальна записка складається зі вступу, трьох розділів і висновків, викладених на 99-ти сторінках.

Метою кваліфікаційної роботи є підвищення рівня захищеності автентифікаційних систем.

Тема кваліфікаційної роботи безпосередньо пов'язана з об'єктом діяльності бакалавра спеціальності 125 «Кібербезпека». Для досягнення поставленої мети в кваліфікаційній роботі вирішуються наступні задачі: характеристика інструментарію, який було використано для виконання роботи, тестування систем автентифікації на вразливості, розробка інструкцій тестування систем автентифікації за отриманими результатами, та розробка рекомендацій, щодо підвищення рівня захищеності систем автентифікації.

Практичне значення результатів кваліфікаційної роботи полягає у підвищені рівня захищеності систем автентифікації методом їх тестування на типові вразливості.

Оформлення пояснювальної записки до кваліфікаційної роботи виконано з незначними відхиленнями від стандартів.

За час дипломування Мірошник А.Ю. проявив себе фахівцем, здатним самостійно вирішувати поставлені задачі та заслуговує присвоєння кваліфікації бакалавра за спеціальністю 125 Кібербезпека, освітньо-професійна програма «Кібербезпека» .

Рівень запозичень у кваліфікаційній роботі не перевищує вимог "Положення про систему виявлення та запобігання плагіату".

Кваліфікаційна робота заслуговує оцінки 90 (Відмінно).

Керівник кваліфікаційної роботи **д.т.н., проф. В.І. Корнієнко**

Керівник спец. Розділу **ст. викл. Д.С Тимофєєв**