Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя
(повне найменування вищого навчального закладу)
Факультет комп'ютерно-інформаційних систем і програмної інженерії
(назва факультету)
Кафедра кібербезпеки
(повна назва кафедри)

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття освітнього ступеня

магістр
(освітній рівень)

на тему: "Дослідження застосування фреймворку Shennina для автоматизованого тестування на проникнення "

Виконав: студент (ка)
Спеціальності:
125 «Кібербезпека та захист інформації»
(шифр і назва напряму підготовки, спеціальності)
Бурмістрова Наталія Андріївна
підпис (прізвище та ініціали)

Керівник Козак Р. О.
підпис (прізвище та ініціали)
Нормоконтроль Тимощук Д. І.
підпис (прізвище та ініціали)

Завідувач кафедри Загородна Н.В.
підпис (прізвище та ініціали)

Рецензент Марценко С.В
підпис (прізвище та ініціали)

м. Тернопіль – 2024

Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя

Факультет комп'ютерно-інформаційних систем і програмної інженерії
(повна назва факультету)

Кафедра кібербезпеки
(повна назва кафедри)

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Загородна Н.В.
(підпис) (прізвище та ініціали)
«__» _____ 2024 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

на здобуття освітнього ступеня Магістр
(назва освітнього ступеня)

за спеціальністю 125 Кібербезпека та захист інформації
(шифр і назва спеціальності)

Студенту Бурмістровій Наталії Андріївні
(прізвище, ім'я, по батькові)

1. Тема роботи Дослідження застосування фреймворку Shennina для автоматизованого тестування на проникнення

Керівник роботи Козак Руслан Орестович, к.т.н., доцент
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

Затверджені наказом ректора від « 20 » 11 2024 року № 4/7-1112

2. Термін подання студентом завершеної роботи 16.12.2024

3. Вихідні дані до роботи Дослідження застосування фреймворку Shennina для автоматизованого тестування на проникнення

4. Зміст роботи (перелік питань, які потрібно розробити)
1 Проблематика та автоматизація тестування на проникнення, 1.1 Особливості та проблеми тестування на проникнення, 1.2 Автоматизація тестування безпеки інформаційних систем, 1.4 Огляд інструментів автоматизованого тестування безпеки, 1.5 Порівняння інструментів, дослідження, 2.1 Архітектура цільової системи для тестування, 2.2 Налаштування сканера вразливостей Nessus, 2.3 Алгоритм дослідження, 2.4 Розробка метрик для порівняння, 3 Аналіз результатів тестування, 3.1 Результати тестування з використанням сканера Nessus, 3.2 Аналіз виявлених вразливостей за допомогою Shennina, 3.3 Порівняльний аналіз можливостей інструментів, 3.3 Висновки щодо можливостей і обмежень фреймворку Shennina, 4 Охорона праці та безпека в надзвичайних ситуаціях, 4.1 Охорона праці, 4.2 Фактори, що впливають на функціональний стан користувачів комп'ютерів, Висновки

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень, слайдів)

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Охорона праці	Осухівська Г.М., к.т.н., доцент		
Безпека в надзвичайних ситуаціях	Клепчик В.М., проректор з адміністративно-господарської роботи та будівництва		

7. Дата видачі завдання _____

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1.	Ознайомлення з завданням до кваліфікаційної роботи	23.09-25.09	Виконано
2.	Пошук і підбір наукових джерел за темою роботи	26.09-01.10	Виконано
3.	Переклад та опрацювання наукових джерел про	02.10-08.10	Виконано
	проблематику, автоматизацію та інструменти		
4.	Огляд та порівняння інструментів автоматизованого	09.10-20.10	Виконано
	тестування безпеки		
5.	Оформлення розділу «Проблематика та автоматизація	21.10-30.10	Виконано
	тестування на проникнення»		
6.	Налаштування цільової системи для тестування	31.10-10.11	Виконано
7.	Розробка алгоритму дослідження	11.11-14.11	Виконано
8.	Оформлення розділу «Методологія дослідження»	15.11-20.11	Виконано
9.	Проведення тестування з використанням Nessus та	21.10-01.12	Виконано
	Shennina		
10.	Оформлення розділу «Аналіз результатів тестування»	02.12-11.12	Виконано
11.	Виконання завдання до підрозділу «Охорона праці та	12.12-15.12	Виконано
	безпека в надзвичайних ситуаціях»		
12.	Нормоконтроль	16.12-19.12	
13.	Перевірка на плагіат	20.12.2024	
14.	Попередній захист кваліфікаційної роботи	21.12-22.12	
15.	Захист кваліфікаційної роботи	23.12.2024	

Студент _____ Бурмістрова Н.А.
(підпис) (прізвище та ініціали)

Керівник роботи _____ Козак Р.О.
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

Дослідження застосування фреймворку Shennina для автоматизованого тестування на проникнення // ОР «Магістр» // Бурмістрова Наталія Андріївна // Тернопільський національний технічний університет імені Івана Пулюя, факультет комп'ютерно-інформаційних систем і програмної інженерії, кафедра кібербезпеки, група СБм-61 // Тернопіль, 2024 // С. 69, рис. – 6, табл. – 8 , кресл. – - , додат. – 1.

Ключові слова: АВТОМАТИЗОВАНЕ ТЕСТУВАННЯ, СКАНУВАННЯ, SHENNINA, METASPLOIT, ВИЯВЛЕННЯ ТА ЕКСПЛУАТАЦІЯ ВРАЗЛИВОСТЕЙ.

Кваліфікаційна робота присвячена дослідженню застосування фреймворку Shennina для автоматизованого тестування на проникнення інформаційних систем. У роботі проведено аналіз можливостей Shennina порівняно з популярним сканером вразливостей Nessus, зокрема щодо виявлення та реальної експлуатації вразливостей. Для практичної перевірки застосовано віртуальну машину Metasploitable 2, що містить навмисно вразливі сервіси.

У ході дослідження налаштовано середовище тестування, розроблено алгоритм дослідження, а також визначено метрики порівняння ефективності інструментів. Особливу увагу приділено можливостям Shennina в контексті автоматизованої експлуатації знайдених вразливостей і моделювання реальних сценаріїв атак.

Отримані результати підтверджують, що Shennina здатна не лише виявляти вразливості, а й успішно експлуатувати їх, адаптуючись до різних конфігураційних помилок у середовищі. Порівняльний аналіз із Nessus вказує на перспективність застосування Shennina для підвищення ефективності процесу тестування на проникнення, що може бути корисним для фахівців у сфері кібербезпеки.

ABSTRACT

Research on the Application of the Shennina Framework for Automated Penetration Testing // Thesis of educational level "Master"// Nataliia Burmistrova // Ternopil Ivan Puluj National Technical University, Faculty of Computer Information Systems and Software Engineering, Department of Cybersecurity, group СБм-61 // Ternopil, 2024 // P. 69, figs. 6, tables 8, drawings -, added. – 1.

Keywords: AUTOMATED TESTING, SCANNING, SHENNINA, METASPLOIT, VULNERABILITY DETECTION AND EXPLOITATION.

The qualification work is dedicated to researching the application of the Shennina framework for automated penetration testing of information systems. The study analyzes the capabilities of Shennina compared to the popular Nessus vulnerability scanner, particularly in terms of vulnerability detection and real exploitation. To practically verify these approaches, the Metasploitable 2 virtual machine, which contains deliberately vulnerable services, was employed.

During the research, the testing environment was configured, a research algorithm was developed, and metrics for comparing the effectiveness of the tools were defined. Special attention was paid to Shennina’s capabilities in terms of automated exploitation of discovered vulnerabilities and modeling realistic attack scenarios.

The obtained results confirm that Shennina can not only detect vulnerabilities but also successfully exploit them, adapting to various configuration errors in the environment. A comparative analysis with Nessus indicates the promise of using Shennina to enhance the efficiency of penetration testing processes, which may be beneficial for cybersecurity professionals.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

AI – Artificial Intelligence

ML – Machine Learning

RL – Reinforcement Learning

CVE – Common Vulnerabilities and Exposures

DoS – Denial of Service

vsftpd – Very Secure FTP Daemon

XSS – Cross-Site Scripting

ВСТУП

Актуальність теми. Сучасний світ стикається з постійним зростанням кількості кібератак, що ставлять під загрозу конфіденційність, цілісність і доступність даних. З огляду на швидке поширення технологій, традиційні підходи до тестування на проникнення стають менш ефективними, оскільки вони вимагають значних часових ресурсів і високої кваліфікації фахівців. Автоматизація процесів пентесту, особливо із застосуванням машинного навчання, відкриває нові можливості для забезпечення кібербезпеки. Фреймворк Shennina, що використовує машинне навчання для адаптивного виявлення та експлуатації вразливостей, є одним із новітніх рішень у цій сфері, що потребує дослідження його ефективності та можливостей.

Мета і задачі дослідження. Метою кваліфікаційної роботи є аналіз можливостей та обмежень фреймворку Shennina для автоматизованого тестування на проникнення в порівнянні зі сканером вразливостей Nessus. Дослідження спрямоване на оцінку ефективності використання Shennina для реальної експлуатації вразливостей та визначення сценаріїв, у яких цей фреймворк є найбільш доцільним.

Завданнями дослідження є проведення огляду сучасних підходів до автоматизації тестування на проникнення, налаштування середовища, проведення тестування з використанням Shennina та Nessus, аналіз та порівняння отриманих результатів, а також формулювання рекомендацій щодо використання Shennina в практичних сценаріях.

Об’єкт дослідження: Об’єктом дослідження є процес автоматизованого тестування на проникнення.

Предмет дослідження: Предметом дослідження є застосування методів та інструментів, зокрема, фреймворку Shennina, для автоматизованого виявлення та експлуатації вразливостей.

Наукова новизна одержаних результатів кваліфікаційної роботи. Наукова новизна полягає у проведенні порівняльного аналізу можливостей фреймворку

Shennina в контексті автоматизації тестування на проникнення, зокрема через зіставлення його функціональності зі сканером вразливостей Nessus, який є інструментом для широкого аудиту безпеки з орієнтацією на ідентифікацію проблем конфігурації та відомих CVE. У роботі розглянуто практичні аспекти реалізації процесів експлуатації вразливостей Shennina, визначено його переваги та обмеження у сценаріях, що моделюють реальні атаки. Отримані результати сприяють формуванню рекомендацій щодо доцільності використання Shennina в пентестах залежно від цілей тестування.

Практичне значення одержаних результатів. Результати дослідження можуть бути використані для вдосконалення процесів тестування на проникнення, зокрема шляхом адаптації методів автоматизованого виявлення та експлуатації вразливостей. Вони є корисними для інтеграції фреймворку Shennina у навчальні програми з підготовки фахівців із кібербезпеки, забезпечуючи можливість моделювання реальних атак і практичного опанування інструментів автоматизації. Порівняльний аналіз з Nessus сприяє оптимізації вибору інструментів для виконання завдань аудиту безпеки, орієнтуючи на найефективніші рішення залежно від цілей тестування та специфіки середовища.

Апробація результатів магістерської роботи. Основні результати проведених досліджень обговорювались на: XII науково-технічній конференції «Інформаційні моделі, системи та технології.

Публікації. Основні результати кваліфікаційної роботи опубліковано у працях конференції (див. Додаток А).

1 ПРОБЛЕМАТИКА ТА АВТОМАТИЗАЦІЯ ТЕСТУВАННЯ НА ПРОНИКНЕННЯ

1.1 Особливості та проблеми тестування на проникнення

Тестування на проникнення є ключовим компонентом забезпечення кібербезпеки, адже дозволяє імітувати реальні кібератаки та оцінювати безпеку системи з точки зору потенційного зловмисника. Основна мета цього процесу полягає у виявленні та усуненні вразливостей системи до того, як ними зможуть скористатися сторонні. Пентест включає декілька етапів, серед яких збирання інформації, сканування, оцінка вразливостей, експлуатація знайдених слабких місць і написання звіту [1] [2]. Послідовний підхід до аналізу системи забезпечує глибоке розуміння її стійкості до атак та можливих загроз.

Пентест має ряд унікальних особливостей, що відрізняють його від стандартних методів аналізу безпеки. По-перше, цей процес імітує дії кіберзлочинців, що дозволяє отримати більш реалістичну оцінку безпеки системи, ніж при використанні звичайних сканувань [3]. Такий підхід допомагає виявити можливі шляхи проникнення і методи експлуатації слабких місць. По-друге, документування кожного етапу процесу забезпечує можливість подальшого аналізу результатів, дозволяє оцінити ефективність застосованих методик та розробити детальні рекомендації для підвищення рівня безпеки.

Незважаючи на високу ефективність, даний метод супроводжується численними проблемами, що обмежують його застосування. Проведення тестування на проникнення є трудомістким та займає значний час, особливо коли йдеться про великі корпоративні мережі чи складні системи. Це зумовлено необхідністю ретельно опрацьовувати кожен етап тестування, аби уникнути пропусків або помилок у результатах. Висока кваліфікація фахівців також є критичною вимогою для успішного проведення пентесту. Спеціалісти мають володіти глибокими знаннями у сфері кібербезпеки, розуміти архітектуру мережевих систем, вміти застосовувати сучасні методи атак і захисту [4][5][6][7].

Іншим суттєвим обмеженням є динамічність загроз – швидке оновлення та розвиток технік і методів атак ускладнює постійне пристосування до нових викликів. Традиційні методи можуть не встигати за темпами змін у сфері кіберзагроз, через що певні вразливості можуть залишатися непоміченими.

Крім того, під час тестування існує високий ризик людських помилок через ручний характер багатьох процесів, що впливає на результати. Помилки, що виникають під час обробки даних або оцінки вразливостей, можуть призвести до невірних висновків або пропуску загроз. Традиційні підходи обмежують масштабованість тестування, що є критично важливим для великих організацій з розгалуженою IT-інфраструктурою [8] [9] [10]. У таких випадках таке тестування не завжди здатне ефективно покривати усі вузли мережі, що створює прогалини у безпеці.

Ще однією суттєвою проблемою є обмеження у сфері охоплення тестування. Часто пентест орієнтований на певні вузли або системи, через що можуть бути проігноровані важливі вразливості в інших частинах інфраструктури, зокрема, у хмарних рішеннях чи IoT-пристроях [11] [12].

Важливим аспектом також є дотримання регуляторних вимог, особливо для організацій, що працюють з конфіденційними даними. Під час пентесту необхідно враховувати численні нормативні стандарти та вимоги до конфіденційності й захисту даних, що може ускладнювати процес та додавати вимог до його організації [13].

Суб’єктивність оцінок – ще один недолік традиційного тестування. Оскільки багато в чому тестування залежить від досвіду та підходів фахівців, його результати можуть варіюватися, що впливає на точність виявлення загроз. Усі ці фактори підкреслюють важливість автоматизації тестування на проникнення. Автоматизація дозволяє знизити вплив людського фактора, підвищити точність, прискорити процес тестування, а також зробити його менш залежним від кваліфікації окремих фахівців.

Таким чином, автоматизація тестування на проникнення стає невід'ємною складовою сучасної системи забезпечення кібербезпеки. Вона не лише сприяє

оперативності й ефективності тестування, але й дозволяє значно розширити його масштаби, що особливо важливо для комплексних систем, де необхідна всеосяжна безпекова перевірка.

1.2 Автоматизація тестування безпеки інформаційних систем

Автоматизація тестування безпеки інформаційних систем відіграє ключову роль у забезпеченні надійного захисту від сучасних кіберзагроз. Зі збільшенням обсягів даних, складності систем і частоти кібератак, традиційні методи ручного тестування на проникнення стають недостатньо ефективними [14]. Відповідно, автоматизовані підходи дозволяють організаціям досягти вищого рівня безпеки, підвищуючи швидкість і точність виявлення вразливостей, знижуючи при цьому ризик людських помилок. Серед найбільш ефективних підходів до автоматизації тестування безпеки можна виділити використання сканерів вразливостей, інструментів, що працюють на основі штучного інтелекту та машинного навчання.

Сканери вразливостей є одним із базових інструментів автоматизованого тестування безпеки інформаційних систем. Їх завданням є виявлення відомих вразливостей, конфігураційних помилок та інших слабких місць, які можуть бути використані для атаки на систему. Сканери працюють на основі сигнатурного аналізу та порівняння з базами відомих вразливостей. Прикладами таких інструментів є Nessus, OpenVAS, QualysGuard та інші. Сканери забезпечують ефективне й швидке виявлення вже відомих загроз і можуть бути налаштовані на регулярне сканування, що дозволяє підтримувати системи у відповідному стані безпеки.

Окрім сканерів вразливостей, сучасна автоматизація тестування безпеки використовує штучний інтелект (AI), що дозволяє проводити більш глибокий аналіз безпеки інформаційних систем. Інструменти на базі AI здатні обробляти великі обсяги даних у режимі реального часу, виявляючи не лише відомі вразливості, але й аномалії у поведінці системи, що можуть свідчити про нові види загроз [5]. Здатність AI до аналізу складних поведінкових патернів забезпечує можливість

проактивного виявлення потенційних атак, що не завжди можуть бути визначені за допомогою традиційних сканерів.

У автоматизації тестування важливу роль відіграє машинне навчання – галузь штучного інтелекту, яка дозволяє системам самостійно навчатися на основі аналізу даних, виявляти патерни та адаптуватися до нових загроз. ML дозволяє розширити можливості автоматизованих інструментів, забезпечуючи здатність до виявлення складних шаблонів у поведінці мережі чи системи, що можуть вказувати на потенційні загрози [15]. Завдяки здатності навчатися на основі великих обсягів даних та вдосконалюватися з кожним новим циклом тестування, ML дозволяє автоматизованим системам виявляти нові загрози, покращувати точність результатів і адаптуватися до нових сценаріїв атак. Це значно підвищує ефективність тестування безпеки, адже алгоритми ML можуть автоматично знаходити аномалії та прогнозувати потенційні загрози.

Серед широкого спектру методів, що охоплює машинне навчання [16], найефективнішими для тестування безпеки є підкріплене навчання (Reinforcement Learning) та глибоке навчання (Deep Learning) [17].

Підкріплене навчання дозволяє системі приймати рішення, орієнтуючись на досягнення найкращого результату у заданому середовищі. Наприклад, інструмент, що використовує підкріплене навчання, може знаходити оптимальні шляхи для атак або вибирати найефективніші методи експлуатації вразливостей у системі. Використовуючи відгуки від попередніх дій, такі інструменти самостійно налаштовують стратегії тестування та підвищують їхню ефективність у реальному часі.

Глибоке навчання, особливо нейронні мережі, дозволяє обробляти величезні обсяги даних і виявляти складні патерни, які можуть вказувати на аномалії або підозрілу активність. Глибокі нейронні мережі особливо корисні для аналізу поведінкових патернів у мережевому трафіку або для класифікації потенційних атак [18] [19]. Це робить тестування не лише точним, але й проактивним, адже система може передбачити нові загрози на основі вивчених патернів.

Машинне навчання активно застосовується для вирішення широкого спектру завдань у сфері тестування безпеки. Одним з ключових застосувань ML є виявлення вразливостей – за допомогою моделей, навчених на даних про попередні атаки, система може ідентифікувати слабкі місця навіть у випадках обмеженої інформації про поточну конфігурацію системи. Це забезпечує ефективне виявлення вразливих ділянок у мережі або програмному забезпеченні, що дозволяє значно зменшити ризик [20].

Машинне навчання також дозволяє автоматизованим системам класифікувати атаки за типами та рівнем ризику, що спрощує оцінку загроз і пріоритизацію їхньої ліквідації. Наприклад, деякі моделі можуть ідентифікувати типи атак, класифікувати їх на базі отриманих даних і оцінювати їхню ймовірність на основі наявних патернів. Це критично важливо для великих систем, де швидкість реагування на загрозу може мати вирішальне значення.

Крім того, алгоритми кластеризації та аналізу аномалій допомагають системам виявляти аномалії, які можуть вказувати на потенційні загрози. Наприклад, ML здатне ідентифікувати підозрілу активність у мережевому трафіку або поведінці користувачів, наприклад, незвично високу активність на певному порту чи нестандартну кількість запитів до бази даних, що може свідчити про атаку [21]. Така функціональність дозволяє системам швидко реагувати на можливі атаки.

1.4 Огляд інструментів автоматизованого тестування безпеки

1.4.1 DeepExploit

DeepExploit – це інноваційний інструмент автоматизованого тестування на проникнення, який застосовує методи машинного навчання, зокрема підкріплене навчання, для підвищення точності й ефективності виявлення вразливостей у мережевих системах.Головною особливістю DeepExploit є його здатність до самонавчання та адаптації на основі отриманих результатів, що робить його особливо корисним у складних середовищах з високою динамікою загроз [22].

Архітектура DeepExploit складається з трьох основних компонентів:

– Агент на основі підкріпленого навчання – відповідає за прийняття рішень під час кожного етапу атаки. Агент аналізує стан цільової системи, зокрема відкриті порти, типи та версії програмного забезпечення, і визначає оптимальні шляхи атаки.
– Сканер вразливостей – використовує інтеграцію з платформою Metasploit для ідентифікації точок входу та потенційних вразливостей у цільовій системі. Цей сканер перевіряє відкриті порти та послуги, а також аналізує відповідність версій програмного забезпечення базі відомих експлойтів.
– Модуль для виконання експлойтів – запускає експлойти після ідентифікації вразливостей, перевіряючи реальну можливість їх експлуатації. Успішні атаки зберігаються для навчання агента на основі зворотного зв'язку, що дозволяє вдосконалювати стратегії атак у подальшому.

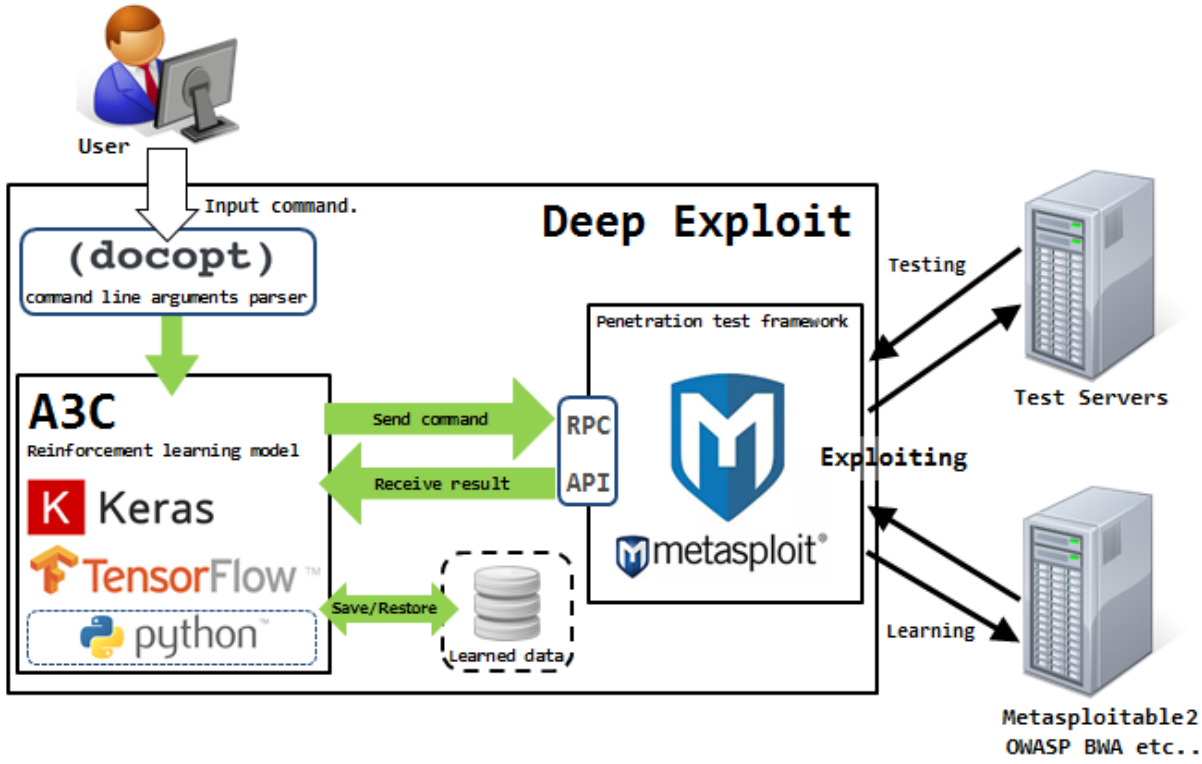

Рисунок 1.1 – Архітектура DeepExploit

Процес роботи DeepExploit включає кілька технічних етапів, що забезпечують ефективне тестування на проникнення. Перший етап – це збір

інформації про цільову систему. Інструмент виявляє відкриті порти, доступні послуги та їх версії, використовуючи дані, отримані від сканування, для оцінки вразливості системи. Таким чином, ідентифікуються компоненти, для яких доступні експлойти в базі Metasploit.

Після збору даних агент DeepExploit використовує цю інформацію для вибору експлойтів, які можуть бути застосовані до вразливих компонентів цільової системи. Агент за допомогою підкріпленого навчання оцінює ефективність кожної спроби і, у випадку невдачі, адаптує свої дії, обираючи інший метод або змінюючи параметри атаки. У разі успішної експлотації система може отримати доступ до цільової системи, що дозволяє продовжувати тестування на глибших рівнях.

Однією з ключових характеристик DeepExploit є його здатність до глибокого проникнення, що дозволяє продовжувати атаку після початкового успіху. Це означає, що при отриманні доступу до однієї частини системи інструмент автоматично намагається розширити атаку на інші внутрішні вузли або сервіси, що дозволяє більш комплексно досліджувати потенційно вразливі сегменти мережі. Це особливо важливо для великих і складних мереж, де взаємопов'язані компоненти можуть містити додаткові точки входу для потенційних атак.

DeepExploit також виділяється здатністю до адаптації та самонавчання. Підхід підкріпленого навчання дозволяє агенту вдосконалювати свої стратегії на основі винагороди за успішно виконані експлойти або розширення доступу. Таким чином, система безперервно вдосконалює свої алгоритми атак, що забезпечує вищу точність і здатність ідентифікувати нові типи загроз у складних умовах.

Рисунок 1.2 – Процес роботи DeepExploit

Важливим аспектом ефективності є його здатність динамічно налаштовувати процес тестування, що забезпечує постійну адаптацію до змін у цільовій системі або мережевому середовищі. Самонавчання дозволяє інструменту не тільки оптимізувати методи атак, але й використовувати нові стратегії для різних архітектур систем [23].

Завдяки всім цим особливостям, DeepExploit є важливим інструментом для автоматизації тестування на проникнення. Він дає можливість знизити навантаження на команду безпеки, підвищити точність виявлення загроз і зменшити залежність від людського фактора, що є критичним у контексті сучасних кіберзагроз.

1.4.2 Shennina

Shennina є передовим інструментом для автоматизованого тестування на проникнення, що ґрунтується на концепціях машинного навчання. Його основною функцією є автоматичне виявлення та експлуатація вразливостей у мережевих системах, що досягається за допомогою здатності самонавчання та адаптації. На відміну від традиційних методів тестування, що покладаються на ручну перевірку і потребують великої кількості часу, даний фреймворк здатен самостійно оцінювати стан цільової системи, обирати відповідні експлойти та швидко адаптуватися до нових сценаріїв атак, що робить його надзвичайно ефективним [24].

Функціональність Shennina спирається на використання підкріпленого навчання для забезпечення прийняття оптимальних рішень на кожному етапі тестування. RL-агент аналізує параметри цільової системи, включаючи відкриті порти, активні сервіси, типи програмного забезпечення та інші мережеві характеристики. На основі зібраних даних агент обирає найкращі шляхи атаки та адаптує свої дії, враховуючи попередні результати успішних і невдалих спроб. Це самонавчання дозволяє Shennina покращувати стратегію тестування на кожному новому етапі, зменшуючи кількість помилок і підвищуючи ефективність процесу.

Однією з важливих функцій Shennina є інтеграція з Metasploit, що дозволяє йому ефективно сканувати систему на наявність відомих вразливостей та застосовувати різноманітні експлойти для їхньої експлуатації. Інструмент проводить початкове сканування системи, виявляє доступні мережеві сервіси, їхні версії та аналізує відповідність цих даних базі експлойтів, яка постійно оновлюється. Завдяки цьому Shennina здатен виконувати автоматизоване тестування, що фокусується на реальних загрозах і мінімізує кількість помилкових спрацювань, які часто трапляються в ручних тестах.

Ключовим елементом роботи Shennina є генерація оптимального шляху атаки. Після аналізу даних та визначення слабких місць у цільовій системі Shennina формує послідовність дій, які максимально підвищують ймовірність успішного проникнення до системи. Цей маршрут зберігається у форматі *.h5 і служить своєрідною картою для подальшого проникнення в систему, що дозволяє оптимізувати ресурси та час тестування [25]. Завдяки такій структурованій побудові шляху атаки Shennina забезпечує ефективність тестування навіть у складних мережевих середовищах, де розподіл ресурсів і швидкість мають вирішальне значення.

Shennina має потужні можливості для постексплуатації, що дозволяє їй не тільки проникати в систему, але й розширювати атаку на інші внутрішні компоненти мережі. Після початкового успіху агент намагається отримати доступ до додаткових сервісів та сегментів мережі, що робить Shennina цінним інструментом для виявлення комплексних загроз у мережах з багатьма взаємопов'язаними вузлами. Ця функція постексплуатації є особливо важливою для великих організацій з розгалуженою інфраструктурою, де потрібно забезпечити комплексну перевірку всіх компонентів системи.

Фреймворк також автоматично генерує звіти після завершення кожного тестування. Ці звіти містять інформацію про IP-адреси цілей, відкриті порти, використані експлойти та результати тестування. Таке документування дає можливість командам безпеки аналізувати виявлені вразливості, краще розуміти структуру системи та розробляти ефективні заходи для покращення захисту.

Завдяки детальним звітам команди можуть легко відслідковувати етапи тестування та отримувати повний огляд ризиків безпеки у цільовій системі.

Крім того, фреймворк використовує концепції MITRE ATT&CK для створення реалістичних атак, що дозволяє Shennina охоплювати понад 40 тактик і технік атак.

Shennina спроєктований з урахуванням вимог до адаптації в реальному часі, що дозволяє йому ефективно функціонувати в різноманітних мережевих середовищах, від невеликих корпоративних мереж до масштабних розподілених систем з високою складністю. Інструмент використовує підхід до динамічної настройки алгоритмів, що базується на обробці даних у режимі реального часу, забезпечуючи постійне оновлення стратегії атак у відповідь на нові типи вразливостей та модифіковані конфігурації мережі.

Ця адаптивність досягається завдяки інтеграції підкріпленого навчання, яке дозволяє системі Shennina ефективно оцінювати і коригувати свої дії, враховуючи результати успішних і невдалих атак. Зокрема, Shennina аналізує такі мережеві параметри, як відкриті порти, типи і версії запущених сервісів, структуру мережевих з'єднань і характер мережевого трафіку. Інструмент активно використовує цю інформацію для корекції своїх стратегій, застосовуючи нові експлойти і змінюючи методику атак залежно від специфіки виявлених вразливостей.

Завдяки можливості постійного навчання та оновлення стратегій атак Shennina підходить для організацій з розгалуженою інфраструктурою, де загрози безпеки можуть швидко розвиватися та змінюватися. Наприклад, якщо під час тестування Shennina виявляє новий тип загрози, система автоматично фіксує ці дані та інтегрує їх у свої алгоритми для майбутнього використання, що дозволяє знизити час реакції на появу подібних атак у майбутньому. Такий підхід забезпечує безперервний захист мережевої інфраструктури, дозволяючи швидко адаптуватися до нових векторів загроз і реагувати на них максимально ефективно.

Здатність до поглибленого аналізу та адаптивного навчання робить Shennina важливим інструментом для сучасного тестування на проникнення. Він дозволяє значно знизити навантаження на фахівців з кібербезпеки, підвищує точність

тестування і зменшує вплив людського фактора на результати тестів. Використовуючи Shennina, організації отримують потужний інструмент, який не тільки допомагає виявляти загрози, але й дозволяє випереджати атаки, забезпечуючи проактивний захист інформаційних систем.

1.5 Порівняння інструментів автоматизованого тестування безпеки

У даному підрозділі буде проведено порівняння двох інструментів для автоматизованого тестування безпеки на основі машинного навчання – Shennina та DeepExploit. Обидва інструменти використовують підкріплене навчання, що дозволяє їм оптимізувати свої дії в реальному часі та адаптуватися до змінних умов кіберсередовища. Завдяки застосуванню машинного навчання, ці інструменти можуть самостійно навчатися на основі зібраних даних, що дає можливість підвищувати ефективність та точність тестування з кожним наступним запуском [26]. Оскільки Shennina та DeepExploit мають подібні базові принципи роботи та функціональність, їхнє порівняння є важливим для оцінки того, який інструмент краще підходить для певних сценаріїв тестування, а також для визначення їхніх сильних та слабких сторін у різних аспектах безпеки інформаційних систем.

Наведені нижче дані є оцінками, заснованими на аналізі функціональних можливостей та загальних характеристик. Оскільки прямі емпіричні дослідження для Shennina та DeepExploit відсутні, отримані значення базуються на даних їхньої архітектури та можливостей.

У таблиці буде проведено порівняння інструментів за ключовими характеристиками, які є критично важливими для автоматизованого тестування безпеки. Серед них: швидкість роботи, що визначає оперативність виявлення вразливостей; адаптивність, яка оцінює здатність інструмента динамічно змінювати стратегії відповідно до змін у середовищі; сумісність із платформою Metasploit, що впливає на ефективність використання експлойтів; можливості постексплуатації, які визначають глибину подальшого дослідження мережі після первинного проникнення.

Також розглядаються якість звітності, яка дозволяє оцінити детальність і зручність автоматично сформованих звітів для подальшого аналізу, використання фреймворку MITRE ATT&CK для реалізації реалістичних сценаріїв атак, а також придатність для роботи у великих мережах, що оцінює ефективність інструмента в складних, розподілених середовищах. Аналіз цих характеристик допоможе визначити сильні та слабкі сторони кожного інструмента, оцінити їхню придатність для різних сценаріїв тестування безпеки та обрати найефективніше рішення залежно від конкретних потреб.

Таблиця 1.1 – Порівняння інструментів на основі характеристик

Характеристика	Shennina	DeepExploit
Швидкість	Висока	Середня
Адаптивність	Висока; використовує підкріплене навчання для динамічної адаптації та корекції стратегій	Середня; адаптація можлива, але з більш повільним відгуком
Сумісність із Metasploit	Інтеграція з Metasploit для автоматизованої експлуатації	Інтеграція з Metasploit для експлуатації
Постексплуатація	Розширена; здатність продовжувати атаку на внутрішні компоненти мережі	Обмежена; потребує додаткової конфігурації
Документування та звітність	Автоматичне створення детальних звітів про результати тестування	Стандартні звіти з базовою інформацією
Використання MITRE ATT&CK	Так; базується на тактиках і техніках MITRE ATT&CK	Обмежене; підтримка окремих тактик
Придатність для великих мереж	Підходить для складних і великих мережевих середовищ	Ефективний у невеликих і середніх мережах

Швидкість сканування є важливою характеристикою, яка визначає здатність інструментів оперативно виявляти загрози. Shennina забезпечує швидкість

сканування вищу порівняно з DeepExploit, завдяки здатності обробляти великий обсяг даних з мінімальними затримками. Це робить Shennina кращим вибором для розподілених мережевих середовищ, де швидкість реагування критично важлива. DeepExploit має середню швидкість сканування, що добре підходить для невеликих або середніх мереж, проте може створювати затримки у великих інфраструктурах, де необхідна висока швидкість у реальному часі.

Адаптивність Shennina є однією з її сильних сторін. Завдяки підкріпленому навчанню інструмент може динамічно налаштовувати свої алгоритми та автоматично коригувати стратегії атак відповідно до нових загроз, які виникають у процесі тестування. Ця можливість адаптації є важливою для організацій з великими, розподіленими інфраструктурами, де швидка відповідь на зміни є критичною. У випадку з DeepExploit адаптивність присутня, але її реалізація менш динамічна, що робить його менш придатним для середовищ із швидкими змінами загроз, але достатньо ефективним для стабільніших мереж.

Сумісність з Metasploit також забезпечує обидва інструменти ефективною інтеграцією з цією платформою для автоматизованої експлуатації. Shennina використовує цю інтеграцію для автоматичного вибору та застосування відповідних експлойтів, зменшуючи необхідність ручної взаємодії. Це дає змогу швидше виконувати тестування та охоплювати більше об'єктів одночасно. DeepExploit також використовує Metasploit для експлуатації вразливостей, але його налаштування може вимагати більше ручного втручання для досягнення подібного рівня автоматизації.

Ще однією характеристикою є здатність до постексплуатації. Shennina має розширені можливості у цьому аспекті, що дозволяє їй продовжувати атаку після початкового проникнення, розширюючи доступ до інших компонентів мережі. Це є важливим для великих організацій, де комплексне охоплення вразливостей на різних рівнях системи забезпечує кращу ефективність тестування. DeepExploit підтримує обмежену постексплуатацію, тому у випадку складних мереж він потребує додаткових налаштувань або супровідних інструментів для забезпечення глибшого охоплення.

Shennina автоматично створює детальні звіти після завершення кожного тестування. Це забезпечує команди безпеки повним оглядом знайдених вразливостей, що спрощує подальший аналіз та розробку заходів для усунення ризиків. У DeepExploit цей аспект менш детально опрацьований, і його звіти надають базову інформацію, що може потребувати додаткового аналізу для отримання повної картини загроз.

Крім того, Shennina активно використовує тактики та техніки з фреймворку MITRE ATT&CK, що дозволяє формувати реалістичні сценарії атак. Завдяки цьому підходу інструмент здатний ефективніше охоплювати всі ключові етапи кіберзагроз, що є важливим для забезпечення повного циклу тестування. DeepExploit також підтримує окремі тактики з MITRE ATT&CK, але його охоплення є менш повним, що може зменшити точність і практичну користь у більш комплексних середовищах.

Загалом, Shennina показує високу ефективність у динамічних середовищах із великими обсягами даних і вимогами до швидкої адаптації та повної автоматизації процесу. DeepExploit, хоча й менш гнучкий, залишається ефективним у стабільних та передбачуваних середовищах, що робить його більш доступним для невеликих організацій або менших проєктів тестування.

2 МЕТОДОЛОГІЯ ДОСЛІДЖЕННЯ

2.1 Архітектура цільової системи для тестування

Для проведення тестування на проникнення та порівняння ефективності інструментів Shennina та Nessus було обрано віртуальну машину Metasploitable 2, розроблену компанією Rapid7. Metasploitable 2 є навмисно вразливою версією операційної системи Ubuntu Linux, спеціально створеною для навчальних цілей у сфері кібербезпеки та тестування на проникнення [27]. Ця віртуальна машина дозволяє відпрацьовувати навички виявлення та експлуатації вразливостей у безпечному та контрольованому середовищі.

Metasploitable 2 була розгорнута з використанням віртуалізатора UTM, що забезпечує сумісність з процесором Apple M1. Це рішення дозволяє створити стабільне середовище для тестування на пристроях з архітектурою ARM. Для коректного встановлення та запуску системи були виконані додаткові кроки, зокрема конвертація образу диска за допомогою команди qemu-img.

Цільова система базується на операційній системі Ubuntu 8.04, яка вже не підтримується виробником. Це означає наявність численних невиправлених вразливостей, що робить систему ідеальною для тестування інструментів безпеки [27]. Система містить широкий спектр застарілого або навмисно вразливого програмного забезпечення та сервісів, що дозволяє моделювати реальні сценарії атак.

Основні сервіси та програмне забезпечення:

- FTP-сервер vsftpd 2.3.4: Ця версія містить вразливість, яка дозволяє неавторизованим користувачам отримати доступ до системи з правами адміністратора. Використовуючи спеціально сформоване ім'я користувача, атакувальник може активувати backdoor.
- Веб-сервер Apache 2.2.8 з PHP 5.2.4: Вразливий до різних атак через застарілі модулі та відсутність оновлень безпеки. Можливі SQL-ін'єкції через вразливі веб-додатки, такі як Mutillidae та Damn Vulnerable Web Application (DVWA), які встановлені на сервері.

– SSH-сервіс OpenSSH 4.7p1: Відсутність належних обмежень та використання слабких або дефолтних паролів дозволяє виконувати атаки грубої сили та отримувати несанкціонований доступ.
– Бази даних MySQL 5.0.51a та PostgreSQL 8.3.0: Дефолтні облікові дані та слабкі паролі відкривають можливість несанкціонованого доступу до баз даних, викрадення даних та виконання довільних SQL-запитів.
– IRC-сервер UnrealIRCd 3.2.8.1: Ця версія сервера містить вбудований backdoor, який дозволяє виконувати віддалені команди з правами адміністратора.
– Сервіс Telnet: Передає дані у незашифрованому вигляді, що дозволяє перехоплювати облікові дані та іншу чутливу інформацію при аналізі мережевого трафіку.
– Apache Tomcat 5.5.25: Має дефолтні облікові дані для менеджера, що дозволяє розгорнути шкідливі веб-додатки та виконувати віддалений код на сервері.

Кожен із цих сервісів має відомі вразливості, що дозволяє детально протестувати інструменти Shennina та Nessus.

Мережеве середовище цільової системи налаштовано таким чином, що віртуальна машина отримує внутрішню IP-адресу в межах віртуальної мережі. Це дозволяє здійснювати підключення до системи та доступ до її сервісів через відкриті порти, моделюючи реальні мережеві атаки [28] [29].

У таблиці 2.1 представлено відкриті порти та сервіси, що були виявлені під час сканування віртуальної машини Metasploitable 2. Ця таблиця містить інформацію про номери портів, відповідні протоколи, сервіси, які використовують ці порти.

Таблиця 2.1 – Відкриті порти та сервіси

Порт	Протокол	Сервіс	Версія
21	TCP	FTP	vsftpd 2.3.4
22	TCP	SSH	OpenSSH 4.7p1 Debian 8ubuntu1
23	TCP	Telnet	Linux telnetd
25	TCP	SMTP	Postfix smtpd

Продовження таблиці 2.1

53	TCP	Domain	ISC BIND 9.4.2
80	TCP	HTTP	Apache httpd 2.2.8 (Ubuntu) DAV/2
111	TCP	RPCBind	2 (RPC #100000)
139	TCP	NetBIOS-SSN	Samba smbd 3.X - 4.X
445	TCP	NetBIOS-SSN	Samba smbd 3.X - 4.X
512	TCP	Exec	netkit-rsh rexecd
513	TCP	Login	
514	TCP	TCPWrapped	
1099	TCP	Java-RMI	GNU Classpath grmiregistry
1524	TCP	Bindshell	Metasploitable root shell
2049	TCP	NFS	2-4 (RPC #100003)
2121	TCP	FTP	ProFTPD 1.3.1
3306	TCP	MySQL	MySQL 5.0.51a-3ubuntu5
5432	TCP	PostgreSQL	PostgreSQL DB 8.3.0 - 8.3.7
5900	TCP	VNC	VNC (protocol 3.3)
6000	TCP	X11	(access denied)
6667	TCP	IRC	UnrealIRCd
8180	TCP	HTTP	Apache Tomcat/Coyote JSP engine 1.1

Вразливості цільової системи включають можливість анонімного доступу до FTP-сервера vsftpd 2.3.4, який має відому вразливість, що дозволяє віддалене виконання коду через бекдор. Веб-сервер Apache 2.2.8 з підтримкою PHP 5.2.4 вразливий до атак типу SQL-ін'єкцій та Cross-Site Scripting (XSS) через встановлені вразливі веб-додатки, такі як Mutillidae та Damn Vulnerable Web Application (DVWA). SSH-сервіс OpenSSH 4.7p1 може піддаватися атакам грубої сили через слабкі або дефолтні паролі.

База даних MySQL 5.0.51a налаштована з дефолтними обліковими даними, що відкриває можливість несанкціонованого доступу та виконання довільних SQL-запитів. Сервіс Telnet передає дані у відкритому вигляді, що дозволяє перехоплювати облікові дані та іншу чутливу інформацію при аналізі мережевого трафіку. IRC-сервер UnrealIRCd 3.2.8.1 містить вбудований бекдор, який дозволяє виконувати віддалені команди з правами адміністратора. Apache Tomcat 5.5.25 має

дефолтні облікові дані для менеджера, що дозволяє розгорнути шкідливі веб-додатки та виконувати віддалений код на сервері [30].

Вибір Metasploitable 2 як цільової машини обумовлений декількома причинами. По-перше, ця система містить більшу кількість вразливостей порівняно з іншими версіями Metasploitable, що дозволяє більш детально оцінити можливості інструментів Shennina та Nessus. По-друге, легкість встановлення та використання спрощує процес підготовки до тестування, що економить час та ресурси. По-третє, різноманітність сервісів та протоколів дозволяє моделювати широкий спектр атак та сценаріїв, підвищуючи цінність дослідження.

У рамках дослідження було внесено зміни до конфігурації сервісів, щоб моделювати реальні сценарії атак, які часто виникають у корпоративних мережах через людський фактор або недотримання політик безпеки. Ці модифікації спрямовані на забезпечення більш глибокого аналізу ефективності інструментів Shennina та Nessus при виявленні складних і специфічних загроз.

Зміни в конфігурації SSH були здійснені з метою створення сценарію, коли адміністратор залишає небезпечні налаштування для спрощення доступу. Було дозволено root-доступ без пароля шляхом активації параметрів PermitRootLogin yes та PermitEmptyPasswords yes у файлі конфігурації SSH. Такі помилки часто зустрічаються на етапах налагодження серверів, коли захисні заходи тимчасово ігноруються або взагалі не впроваджуються. Це створює ідеальне середовище для моделювання атак типу грубої сили, перехоплення сесій або несанкціонованого віддаленого управління сервером.

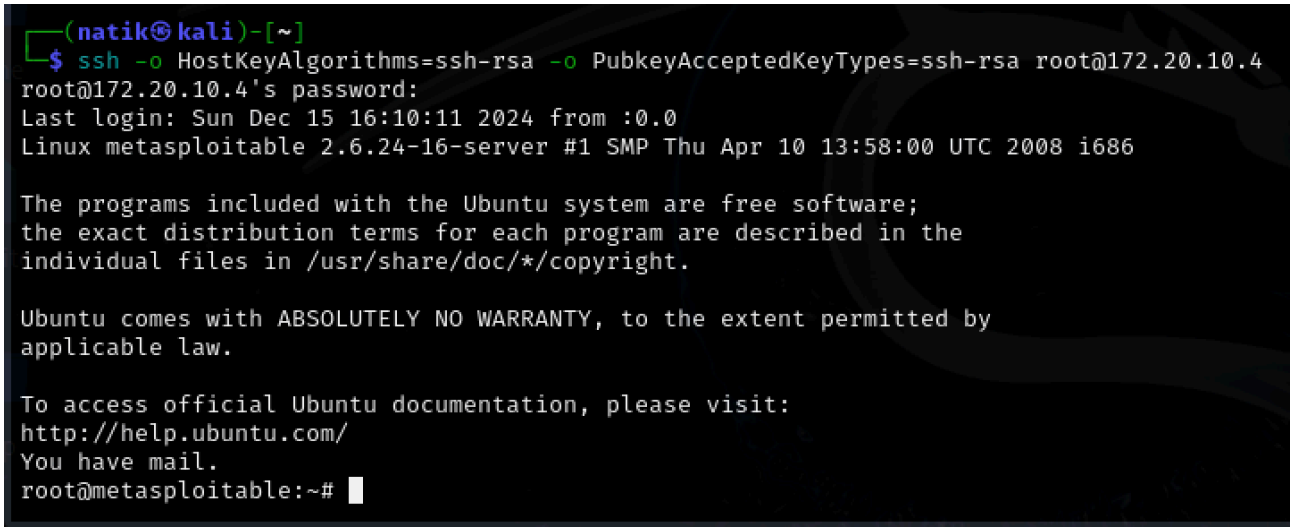

Рисунок 2.1 – Підтвердження небезпечної конфігурації SSH Підтвердження небезпечної конфігурації SSH із дозволеним root-доступом

Для створення додаткової вразливості в MySQL було використано параметр skip-grant-tables, що дозволяє обійти механізм перевірки облікових даних. Така конфігурація часто застосовується адміністраторами для усунення збоїв у роботі бази даних, але її використання без подальшого відновлення стандартних налаштувань є серйозним ризиком. Ця вразливість дозволяє моделювати сценарії, коли зловмисник отримує доступ до бази даних, виконуючи довільні SQL-запити, що може призвести до витоку конфіденційної інформації або змін у критичних даних.

Рисунок 2.2 – Підтвердження вразливості MySQL

Публічний ресурс Samba був створений шляхом налаштування доступу до мережевої папки з параметрами, що дозволяють анонімний доступ (guest ok = yes), можливість запису (writable = yes) та перегляд вмісту (browsable = yes). Ця модифікація моделює реальні ситуації, коли некоректні налаштування доступу до мережевих ресурсів дозволяють атакувальникам завантажувати шкідливі файли, викрадати дані або запускати небезпечні скрипти. Подібні помилки часто трапляються через низький рівень контролю за конфігурацією мережевих ресурсів.

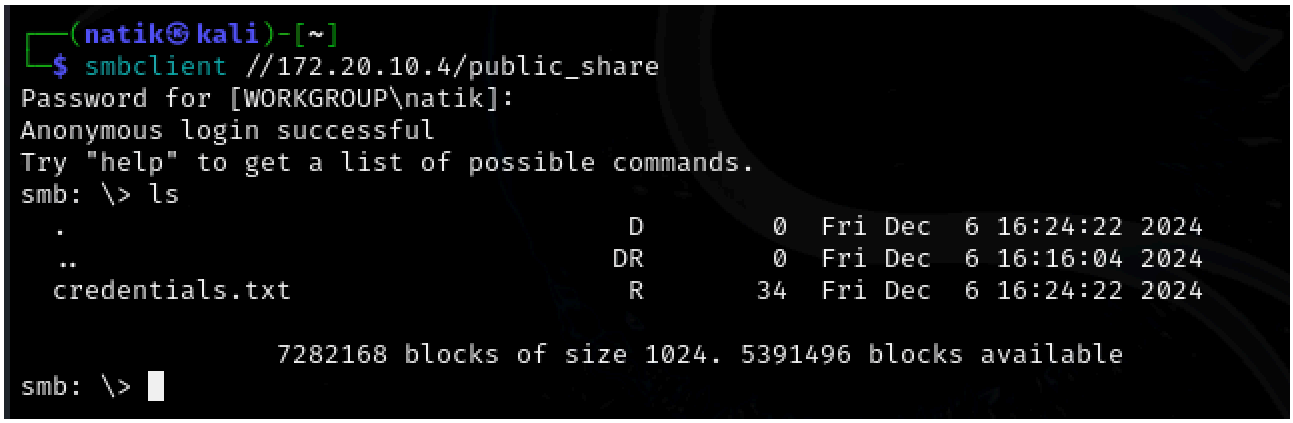

Рисунок 2.3 – Підтвердження анонімного доступу до Samba

Кожна з доданих вразливостей моделює типові проблеми корпоративних мереж, де недоліки в конфігурації або нехтування політиками безпеки створюють значні загрози для інформаційних систем. Це дозволяє оцінити, наскільки добре інструменти можуть автоматично виявляти та аналізувати ці загрози, забезпечуючи практичний внесок у покращення захисту інформаційних систем.

Проведення тестування на Metasploitable 2 сприяє поглибленню практичних навичок у галузі кібербезпеки, дозволяє відпрацьовувати методики виявлення та аналізу вразливостей, а також оцінювати ефективність різних інструментів безпеки в умовах, наближених до реальних корпоративних мереж.

Подальші кроки дослідження включають проведення повного сканування системи за допомогою Shennina та Nessus, аналіз та порівняння отриманих результатів щодо виявлених вразливостей та оцінку можливостей автоматизації процесу тестування на проникнення за допомогою Shennina.

2.2 Налаштування сканера вразливостей Nessus

Для порівняння ефективності було обрано сканер вразливостей Nessus завдяки його широкому використанню в галузі кібербезпеки, високій точності виявлення вразливостей та доступності безкоштовної версії, яка дозволяє виконувати основні завдання сканування. Nessus є одним із найпопулярніших інструментів у світі для оцінки безпеки, що забезпечує користувачам ефективні алгоритми сканування, інтуїтивний інтерфейс і можливість роботи з великим спектром цільових систем, включаючи Metasploitable 2 [31].

Його вибір для даного дослідження зумовлене наступними причинами:

– Репутація і точність: Nessus визнаний стандартом у сфері сканування вразливостей і забезпечує високу точність ідентифікації проблем безпеки.
– Гнучкість налаштування: Nessus надає широкі можливості конфігурації для адаптації до різних сценаріїв сканування.
– Безкоштовна версія: Універсальний доступ до базової функціональності дозволяє використовувати інструмент без значних фінансових витрат.
– Підтримка різних середовищ: Nessus підтримує як реальні мережі, так і лабораторні середовища, такі як Metasploitable 2.

Конфігурація Nessus є однією з найважливіших частин процесу тестування, адже вона визначає параметри сканування, поведінку інструменту та точність результатів. Коректно налаштована конфігурація дозволяє адаптувати процес сканування до специфіки цільової системи, зокрема Metasploitable 2, яка містить велику кількість уразливостей для навчання та тестування.

У цьому підрозділі детально розглянуто й обґрунтовано налаштування Nessus для сканування Metasploitable 2. Роз'яснено, як кожен пункт конфігурації впливає на процес сканування та які результати можуть бути отримані. Окрему увагу приділено таким аспектам конфігурації, як Пошук (Discovery) та Оцінка (Assessment), які є ключовими для виявлення мережевих сервісів і подальшого аналізу їх уразливостей.

Налаштування пошуку включає в себе пошук хостів, портів та сервісів. Основна функція пошуку хостів – пінгування віддалених хостів (Remote Host Ping). Завдяки цьому параметру виконується перевірка доступності хостів перед їх скануванням на вразливості, що дозволяє зменшити кількість хибнопозитивних результатів і забезпечити коректність подальшого аналізу.

Налаштування пінгування включає в себе кілька різних методів, які дозволяють встановити доступність віддалених хостів:

- TCP Ping.
- UDP Ping.
- ICMP Ping.
- ARP Ping.

Використання різних методів пінгування дозволяє компенсувати обмеження, що можуть виникати через блокування певного типу трафіку в мережі. Наприклад, якщо ICMP-пакети блокуються між хостами, Nessus може використовувати TCP або UDP Ping для визначення доступності.

Опція ICMP Ping надає можливість детальної конфігурації. Наприклад, можна увімкнути функцію "Assume ICMP unreachable from the gateway means the host is down" (Припустити, що недоступність хосту через ICMP від шлюзу означає, що хост недоступний). Однак ця функція не була включена до конфігурації, оскільки її використання може призвести до зниження точності сканування [32].

Ось кілька причин, чому цю опцію було вирішено не включати:

- Мережевий пристрій, що виконує функцію шлюзу, не завжди може відправляти ICMP-повідомлення про недоступність хосту. Такий розвиток подій може бути у випадку неправильної конфігурації, блокування ICMP на шлюзі або через певні його особливості. У такому разі, якщо ввімкнена даної функції, сканер помилково визначить хост як недоступний.
- Якщо мережевий пристрій надсилає ICMP-повідомлення про недоступність коли насправді хост активний, можливі ситуації, коли сканер пропустить такі хости під час сканування, вважаючи їх недоступними. Це може призвести до пропуску виявлення вразливостей або пропуску активних систем у мережі.

– Враховуючи ймовірність таких ситуацій, включення цієї опції може вплинути на достовірність результатів і зменшити їх надійність.

Таким чином, було прийнято рішення не використовувати цю функцію для забезпечення максимальної точності виявлення активних хостів. Додатково було налаштовано параметр "Maximum number of retries" (Максимальна кількість повторних спроб), встановивши його значення на 2, що дозволяє зменшити ймовірність помилкового визнання активного хосту недоступним.

Наступним етапом налаштування конфігурації пошуку хостів є параметр "Fragile Devices" (Крихкі пристрої). Цей параметр включає в себе кілька типів пристроїв, які потребують особливої уваги під час сканування, оскільки вони можуть бути вразливими до порушення функціональності через мережеве навантаження:

– Сканування мережевих принтерів.
– Сканування хостів Novell NetWare.
– Сканування пристроїв операційної технології (Operational Technology, OT).

Під час налаштування конфігурації не було включено опцію "Сканування мережевих принтерів", оскільки заздалегідь було відомо, що в середовищі Metasploitable 2 такі пристрої відсутні. Включення цієї опції лише збільшило б тривалість сканування без додаткової користі для отриманих результатів, тому її було вирішено виключити з конфігурації.

До налаштувань не було включено параметр "Сканування хостів Novell NetWare", оскільки даний тип хостів відсутній у цільовій системі.

Novell NetWare є старою мережевою операційною системою, яка не використовується у складі Metasploitable 2. Включення параметра було б недоцільним, оскільки це збільшило б тривалість сканування без внесення додаткової цінності до отриманих результатів [33].

Опція "Scan Operational Technology (OT) Devices" не була включена до конфігурації, оскільки Metasploitable 2 не містить пристроїв, пов'язаних із промисловими системами керування (ICS) чи системами диспетчерського управління і збору даних (SCADA). Включення цього параметра було б

недоцільним, адже воно б не вплинуло на результати сканування, але збільшило б час, необхідний для виконання сканування.

Сканування OT-пристроїв є важливим для середовищ, що включають критичну інфраструктуру, однак у цьому дослідженні зосереджено увагу на системі Metasploitable 2, яка моделює типові серверні вразливості у мережевому середовищі, де такі пристрої відсутні. Таким чином, ця опція була виключена з налаштувань для оптимізації процесу.

Наступним кроком конфігурації стало налаштування етапу "Сканування портів", який має ключове значення для виявлення активних сервісів і потенційних вразливостей. Ці параметри дозволяють детально проаналізувати стан портів, визначити відкриті, закриті чи фільтровані порти, а також ідентифікувати сервіси, що працюють на них.

Було змінено діапазон сканування портів із параметра "default" на "all", що дозволяє сканувати всі порти (від 1 до 65535). Це рішення прийнято з метою виявлення сервісів, які працюють на нестандартних портах. Metasploitable 2 містить уразливі сервіси, такі як FTP, MySQL, Samba та інші, які можуть працювати як на стандартних, так і на нетипових портах. Використання параметра "default" могло б призвести до пропуску певних портів, що зменшило б ефективність тестування.

До конфігурації було включено параметри для сканування локальних портів, які включають:

- SSH (netstat) – для перевірки доступності служби SSH.
- SNMP – для перевірки роботи Simple Network Management Protocol.
- TCP-порти, знайдені локальними нумераторами портів.

Сканування локальних портів дозволяє виявити потенційні вразливості, зокрема сервіси, які можуть бути доступними для експлуатації. Наприклад, відкриті TCP-порти, залишені через неправильне налаштування, можуть слугувати точкою входу для атак.

Було налаштовано SYN-сканування, що є одним із найефективніших методів для перевірки стану портів. SYN-сканування відправляє TCP-пакети з прапором SYN (synchronize) і очікує відповіді від цільового хоста. У разі відкритого порту

хост відповідає пакетом із прапором SYN/ACK, що дозволяє сканеру визначити, що порт доступний [34]. У випадку закритого або фільтрованого порту хост відправить пакет із прапором RST (Reset).

Використання SYN-сканування забезпечує кілька переваг:

- Процес не встановлює повного TCP-з’єднання, що робить сканування менш помітним у мережі.
- Цей метод дозволяє обійти певні фільтри чи брандмауери, що можуть блокувати інші типи сканування.
- SYN-сканування дозволяє отримати точну оцінку стану портів, включно з відкритими, закритими та фільтрованими портами.

Крім того, у конфігурації було встановлено параметр для повторних спроб у разі відсутності відповіді від хоста, щоб уникнути пропуску портів через тимчасову нестабільність мережі. Таймаут для відповіді встановлено на рівні 5 секунд, що дозволяє врахувати затримки в обробці запитів.

На етапі "Пошук сервісів" було проведено налаштування параметрів, спрямованих на виявлення максимально повної інформації про активні служби та їхні характеристики.

Першим важливим пунктом стало включення параметра "Probe all ports to find services". Це налаштування дозволяє Nessus зіставити кожен відкритий порт із відповідною службою чи додатком, які відповідають на цьому порту. Такий підхід забезпечує виявлення служб, які працюють на нестандартних портах, або служб, які не є типовими для певного типу пристрою. Це дозволяє визначити потенційно вразливі сервіси, які можуть бути експлуатовані в ході тестування.

Параметр "Search for SSL/TLS/DTLS services" не був включений до конфігурації, оскільки система Metasploitable 2 не містить служб, які використовують протоколи SSL/TLS/DTLS для захищеної передачі даних. Хоча цей параметр може бути корисним у реальних середовищах, у цьому дослідженні він не має практичного застосування. Його виключення дозволило оптимізувати час сканування.

Також не було включено параметри "Пошук SSL/TLS на всіх TCP-портах", "Пошук DTLS на всіх UDP-портах" та "Identify certificates expiring within x days", оскільки протоколи SSL/TLS/DTLS і сертифікати відсутні в системі Metasploitable 2. Ці параметри зазвичай використовуються для шифрування даних і моніторингу сертифікатів у реальних середовищах, але в цьому дослідженні вони не є релевантними. Виключення цих налаштувань дозволило оптимізувати час сканування та зосередитися на особливостях цільової системи.

Оцінка (Assessment) є наступним етапом конфігурації сканера Nessus, спрямованим на глибший аналіз вразливостей у виявлених службах та компонентах системи.

Даний етап включає кілька важливих розділів, кожен із яких відповідає за перевірку конкретного типу служб або компонентів.

Основні підрозділи налаштувань:

- Загальні налаштування ("General") – охоплюють базові перевірки, які стосуються всіх типів систем. До них входить аналіз версій програмного забезпечення, перевірка політик паролів та загальних налаштувань безпеки.
- Веб-застосунок ("Web Application") – включає перевірки безпеки веб-додатків, таких як виявлення SQL-ін’єкцій, XSS-атак, а також аналіз конфігурацій веб-серверів.
- Налаштування Віндовс ("Windows") – зосереджуються на перевірці компонентів операційної системи Windows, таких як облікові записи, налаштування реєстру, оновлення та служби.
- Бази даних ("Databases") – спрямовані на аналіз баз даних, перевірку версій серверів баз даних, їхньої конфігурації та виявлення уразливостей.

Важливим параметром конфігурації є точність ("Accuracy"), яка визначає здатність сканера розпізнавати та класифікувати дані з високим ступенем достовірності. До цього параметра належать наступні налаштування:

- Перевизначення нормальної точності ("Override normal accuracy").
- Проведення ретельного тесту ("Perform thorough tests").

Перевизначення нормальної точності дозволяє змінювати базові налаштування алгоритму для підвищення або зниження чутливості до різних типів даних. Це може бути корисним для адаптації сканера до специфіки цільового середовища. У цьому дослідженні було обрано додаткове налаштування "Avoid potential false alarm" (Уникнення можливих помилкових сигналів), щоб мінімізувати кількість помилкових спрацювань і покращити точність результатів. Інший варіант, "Show potential false alarms" (Показати можливі помилкові сигнали), не використовувався, оскільки зосередження на мінімізації помилкових класифікацій мало пріоритет у налаштуванні конфігурації.

Параметр "Perform thorough tests" (Проведення ретельного тесту) не був включений до конфігурації, оскільки такий вид сканування значно впливає на продуктивність і може створювати надмірне навантаження на мережу. Враховуючи обмеженість ресурсів у середовищі Metasploitable 2, було вирішено уникнути використання цього параметра, щоб запобігти зниженню швидкості передачі даних чи перебоям у роботі системи.

Такі рішення в налаштуванні точності дозволяють забезпечити баланс між продуктивністю сканування і точністю отриманих результатів, що є важливим для оптимізації процесу тестування.

На етапі "Веб-застосунок" було налаштовано параметри для сканування веб-додатків, які дозволяють перевірити безпеку веб-сервісів, доступних у середовищі.

Першим важливим налаштуванням було залишено дефолтне значення заголовка User-Agent: "Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 5.1; Trident/4.0)". Це значення симулює браузер Internet Explorer 8, що забезпечує сумісність із застарілими веб-додатками, які використовуються в Metasploitable 2, такими як Mutillidae і DVWA.

Параметри веб-сканера ("Web Crawler") також було налаштовано для оптимального збору інформації. Значення параметра глибини переходу ("Depth of Crawl") встановлено на 6, що дозволяє веб-сканеру переходити до шести рівнів посилань від початкової сторінки. Це значення забезпечує збір достатнього обсягу інформації, не створюючи надмірного навантаження на сервер. Максимальну

кількість сторінок для сканування ("Maximum pages to crawl") було обмежено до 1000, щоб контролювати обсяг зібраних даних і не перевантажувати процес сканування.

До конфігурації було включено параметр "Follow dynamically generated pages" (Слідувати за динамічно створеними сторінками), який дозволяє сканеру аналізувати сторінки, що генеруються на сервері, наприклад, залежно від параметрів URL або запитів користувача. Це дозволяє отримати більше інформації про варіації сторінок, що можуть містити додаткові вразливості.

У розділі "Application Test Settings" було ввімкнуто наступні налаштування:

- "Enable generic web application tests" – для виявлення вразливостей, таких як SQL-ін'єкція, міжсайтовий скриптинг (XSS) та інші.
- "Try all HTTP methods" – для перевірки потенційних проблем безпеки, пов'язаних із різними HTTP-методами (GET, POST, PUT, DELETE тощо).
- "Attempt HTTP Parameter Pollution" – для тестування можливих вразливостей, пов'язаних із перевантаженням параметрів HTTP-запитів.

Параметр "Test embedded web servers" (Тестування вбудованих веб-серверів) також було ввімкнено, оскільки цільова машина містить кілька простих вбудованих веб-додатків. Ця опція дозволяє перевіряти сервіси, які запускаються на веб-серверах, таких як Apache Tomcat.

Також було додано додаткові налаштування:

- Тестування більше одного параметра одночасно у формах.
- Не зупинятися після виявлення першої вразливості на сторінці.

Ці налаштування забезпечують глибший аналіз безпеки веб-додатків, дозволяючи виявити складніші типи вразливостей, які можуть виникати через взаємодію різних параметрів або елементів веб-сторінки.

Налаштування розділу Windows було виключено з конфігурації, оскільки середовище Metasploitable 2 побудоване на базі Linux і не містить компонентів Windows, таких як SMB-домени, Active Directory або WMI (Windows Management Instrumentation). Методи, орієнтовані на Windows-системи, такі як SAM Registry, ADSI Query, WMI Query, а також функції RID Brute Forcing та аналіз UID, не

застосовуються в Linux-системах. Виняток цих параметрів дозволив оптимізувати процес сканування, зосередивши увагу на релевантних аспектах середовища.

Розділ Бази даних залишився важливим для сканування, оскільки Metasploitable 2 включає сервіси баз даних, такі як MySQL і PostgreSQL. Налаштування параметра "Use detected SIDs", який орієнтований на ідентифікацію SID у Windows-середовищах, не було включено, оскільки він не є релевантним для Linux. Натомість конфігурація сканера була адаптована для аналізу серверів баз даних, доступних у Metasploitable 2, що дозволяє виявити їхні вразливості, такі як слабкі паролі, некоректні налаштування або уразливі версії програмного забезпечення.

Такі адаптації конфігурації забезпечили точність і ефективність сканування, дозволяючи зосередитися на ключових характеристиках системи, яка є цільовим середовищем дослідження.

2.3 Алгоритм дослідження

Для дослідження використовували два основних інструменти: Nessus і Shennina, які виконували сканування в контрольованому середовищі. Алгоритм дослідження передбачає чітко структуровані етапи, що забезпечують повноту аналізу та можливість порівняння результатів між обраними інструментами.

Алгоритм передбачає наступні етапи:

- Підготовка середовища.
- Сканування за допомогою Nessus.
- Сканування за допомогою Shennina.
- Аналіз результатів.
- Оцінка ефективності Shennina.

Для візуального представлення алгоритму дослідження була створена блок-схема, яка демонструє послідовність основних етапів і їхній взаємозв'язок.

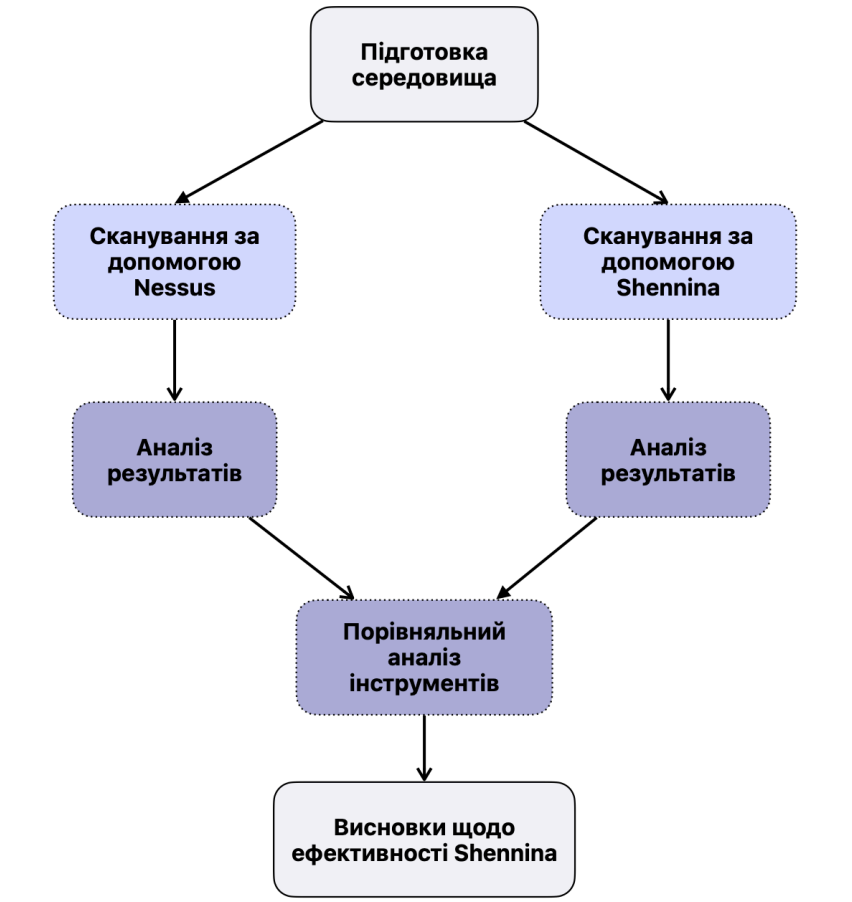

Рисунок 2.4 – Блок-схема алгоритму дослідження

На етапі підготовки середовища було створено ізольоване середовище, яке забезпечило безпеку та контрольованість умов. Основні дії включали:

– Встановлення Metasploitable 2.
– Розгортання базової інфраструктури.
– Забезпечення мережевої ізоляції.

Розгорнуто тестову систему з відомими вразливостями у вигляді віртуальної машини, яка забезпечує реалістичне середовище для моделювання атак і виявлення вразливостей. Як основну платформу використано Kali Linux, яка містить усі необхідні утиліти, включаючи мережеві аналізатори та засоби налаштування. Для мінімізації ризиків усі компоненти середовища були повністю ізольовані від реальної інфраструктури. Комунікація між хостами налаштовувалася через внутрішній віртуальний адаптер.

Для сканування за допомогою Nessus було реалізовано базовий сценарій аналізу системи Metasploitable 2. Основні кроки включали:

– Створення профілю сканування.
– Запуск сканування.
– Формування звіту.

На етапі створення профілю сканування було налаштовано такі параметри:

– Виявлення відкритих портів.
– Ідентифікація активних сервісів.
– Аналіз базових вразливостей.

Профіль був адаптований для роботи з усіма мережевими портами й сервісами, доступними у Metasploitable 2. Запуск сканування здійснювався за допомогою безкоштовної версії Nessus. Після завершення аналізу формувався детальний звіт, що включав знайдені вразливості, їхній рівень ризику та рекомендації щодо усунення.

Сканування за допомогою Shennina було ключовим етапом дослідження, спрямованим на демонстрацію можливостей інструменту. Основні кроки включали:

– Налаштування інструменту.
– Конфігурацію параметрів сканування.
– Проведення сканування.

Shennina була налаштована у віртуальному середовищі Python. Необхідні бібліотеки були встановлені через файл requirements.txt. Конфігурація параметрів охоплювала:

– Аналіз мережевих протоколів.
– Пошук уразливостей у службах.
– Оптимізацію параметрів для роботи з Metasploitable 2.

Сканування проводилося у кілька етапів для забезпечення повноти аналізу. У результаті інструмент генерував звіт із описом знайдених вразливостей і їх характеристик.

На етапі аналізу результатів проводилося порівняння даних, отриманих за допомогою Nessus і Shennina. Основні аспекти включали:

– Кількість виявлених вразливостей.
– Типи ідентифікованих проблем.
– Рівень ризику для кожної з них.

Ефективність Shennina була оцінена за ключовими параметрами, такими як точність, швидкість і покриття. Інструмент продемонстрував високу здатність виявляти складні вразливості, що може значно посилити процес аналізу безпеки. Однак час сканування був довшим у порівнянні з Nessus, що пов’язано з більш глибоким підходом Shennina.

2.4 Розробка метрик для порівняння

Для оцінки роботи Shennina та Nessus було обрано такі метрики, які дозволяють провести об'єктивне порівняння цих інструментів у контексті автоматизованого тестування безпеки:

– Кількість виявлених вразливостей.
– Час виконання сканування.
– Глибина аналізу.
– Покриття тестової системи.
– Споживання системних ресурсів.

Кількість виявлених вразливостей є основною метрикою, яка характеризує здатність інструменту ідентифікувати проблеми безпеки. Порівняння базується на загальній кількості знайдених вразливостей, розподілених за категоріями ризику: низький, середній та високий. Shennina демонструє здатність знаходити складні вразливості, які часто пропускаються іншими інструментами через інтеграцію алгоритмів машинного навчання. Nessus, зі свого боку, характеризується стабільністю та широким охопленням стандартних сценаріїв, забезпечуючи повну картину вразливостей на базовому рівні.

Час виконання сканування є важливим показником ефективності, особливо в умовах, коли швидкість аналізу відіграє критичну роль. Для Shennina цей показник може бути довшим через глибокий аналіз системи та мережі, що потребує більше часу для обробки даних. Nessus забезпечує оптимізований процес сканування завдяки ефективним алгоритмам і меншому навантаженню на процесор, що дозволяє завершувати перевірки за коротший час.

Глибина аналізу характеризує здатність інструмента виявляти вразливості, які приховані або складно визначаються звичайними методами. Shennina орієнтована на аналіз складних загроз, що дозволяє їй виявляти вразливості у специфічних налаштуваннях або неочевидних місцях. Nessus, хоч і забезпечує значну глибину аналізу, може обмежуватись типовими сценаріями, через що складніші вразливості можуть залишатись поза увагою.

Покриття тестової системи включає аналіз відкритих портів, протоколів, сервісів та інших компонентів системи. Shennina має перевагу в адаптації під специфічні середовища та сценарії, що дозволяє їй більш ефективно охоплювати комплексні інфраструктури. Nessus демонструє більш узагальнене покриття, орієнтоване на стандартні конфігурації.

Споживання системних ресурсів є критичною характеристикою, особливо для масштабованих систем. Shennina через інтеграцію алгоритмів машинного навчання може вимагати більших обчислювальних ресурсів, що обмежує її використання в умовах із низькою потужністю. Nessus, у свою чергу, забезпечує більш економічне використання ресурсів, залишаючись ефективним навіть на системах із обмеженою продуктивністю.

Ці метрики дозволяють сформувати комплексну оцінку кожного інструменту, враховуючи їхні сильні та слабкі сторони, і забезпечити об'єктивне порівняння ефективності.

3 АНАЛІЗ РЕЗУЛЬТАТІВ ТЕСТУВАННЯ

3.1 Результати тестування з використанням сканера Nessus

Сканування віртуальної машини Metasploitable 2 за допомогою інструмента Nessus дозволило виявити широкий спектр вразливостей, які варіюються за рівнями критичності. Загальний час виконання сканування становив 3 години, протягом яких було ідентифіковано 153 вразливості.

Серед виявлених вразливостей розподіл за рівнями критичності є наступним:

- Критичні (Critical): 9 вразливостей, що представляють собою найбільшу загрозу для системи та потребують негайного усунення.
- Високі (High): 12 вразливостей, які також можуть бути використані зловмисниками для компрометації системи, але потребують певних додаткових умов для успішної експлуатації.
- Середні (Medium): 41 вразливість, що потенційно можуть призвести до ускладнень у роботі системи, але вимагають більшої складності для атаки.
- Низькі (Low): 8 вразливостей, що становлять мінімальну загрозу та, як правило, не можуть бути використані для значного впливу на систему.

Під час аналізу результатів сканування було встановлено, що Nessus виявив усі очікувані відкриті порти та сервіси, наявні на цільовій системі. Серед виявлених портів були також додаткові, які не входили до початкового списку: 0, 69, 1099, 1524, 3632, 38829, 52056, 55439. Їхня наявність може бути зумовлена специфічною конфігурацією віртуальної машини та внутрішніми службами системи.

Сканування виявило кілька критичних вразливостей, які становлять значну загрозу для безпеки цільової системи. У таблиці нижче наведено результати аналізу критичних вразливостей, включаючи порти, протоколи, та назви вразливостей.

Таблиця 3.1 – Критичні вразливості

Risk	**Protocol**	**Name**	**Port**
Critical	tcp	Apache PHP-CGI Remote Code Execution	80

Продовження таблиці 3.1

Critical	tcp	Bind Shell Backdoor Detection	1524
Critical	tcp	Debian OpenSSH/OpenSSL Package Random Number Generator Weakness	22
Critical	tcp	Debian OpenSSH/OpenSSL Package Random Number Generator Weakness (SSL check)	25, 5432
Critical	tcp	SSL Version 2 and 3 Protocol Detection	25, 5432
Critical	tcp	VNC Server 'password' Password	5900
Critical	tcp	phpMyAdmin prior to 4.8.6 SQLi vulnerablity (PMASA-2019-3)	80

Крім того, Nessus визначив CVE-ідентифікатори для частини вразливостей, таких як CVE-2008-0166, CVE-2012-1823, CVE-2012-2311, CVE-2019-11768, і виявив можливість використання модулів Metasploit для експлуатації, зокрема для PHP-CGI Remote Code Execution (CVE-2012-1823) і SQL-ін'єкцій у phpMyAdmin (CVE-2019-11768).

Nessus підтвердив наявність вразливостей за допомогою таких методів:

- Порівняння з відомими вразливими версіями програмного забезпечення. Інструмент ідентифікував версії сервісів, що містять уразливості. SSLv2 і SSLv3 були виявлені через аналіз підтримуваних протоколів і використання слабких алгоритмів шифрування. Версія phpMyAdmin 3.1.1 була ідентифікована як вразлива до SQL-ін'єкцій.
- Аналіз конфігурацій та налаштувань. VNC-сервер було ідентифіковано як незахищений через використання слабкого пароля "password", що створює високий ризик несанкціонованого доступу до віддаленого робочого столу.

Використання спеціалізованих плагінів. Debian OpenSSH/OpenSSL Random Number Generator Weakness була ідентифікована шляхом перевірки криптографічних ключів на відповідність відомим слабкостям.

Nessus не виконував активну експлуатацію вразливостей, а обмежився перевіркою версій та конфігурацій програмного забезпечення.

Порівняння отриманих результатів із документацією Metasploitable 2 показало, що сканер успішно виявив більшість критичних вразливостей.

Проте аналіз виявив, що не було виявлено деякі критичні вразливості, які задокументовані для Metasploitable 2, зокрема:

– Додаткові налаштування вразливих сервісів Apache Tomcat.
– Можливі експлуатаційні сценарії для FTP-сервера vsftpd із вбудованим бекдором.

Ідентифіковано низку вразливостей високого рівня, які можуть створювати значні ризики для безпеки цільової системи. У таблиці нижче представлено перелік виявлених вразливостей із зазначенням відповідних портів, протоколів та їхніх назв.

Таблиця 3.2 – Вразливості високого рівня

Risk	Protocol	Name	Port
High	tcp	CGI Generic Remote File Inclusion	80
High	tcp	ISC BIND Service Downgrade / Reflected DoS	53
High	tcp	NFS Shares World Readable	2049
High	tcp	PHP PHP-CGI Query String Parameter Injection Arbitrary Code Execution	80
High	tcp	SSL Medium Strength Cipher Suites Supported (SWEET32)	25, 5432
High	tcp	Samba Badlock Vulnerability	445
High	tcp	TWiki 'rev' Parameter Arbitrary Command Execution	80
High	tcp	TikiWiki tiki-graph_formula.php f Parameter Arbitrary Command Execution	80

Продовження таблиці 3.2

High	tcp	phpMyAdmin Setup Script Configuration Parameters Arbitrary PHP Code Injection	80
High	tcp	rlogin Service Detection	513
High	tcp	rsh Service Detection	514
High	udp	ISC BIND Service Downgrade / Reflected DoS	53

Для вразливостей високого рівня виявлено CVE-ідентифікатори. Зокрема, CVE-2016-2183 пов'язаний із підтримкою слабких SSL-шифрів (SWEET32) на портах 25 і 5432. CVE-2012-1823 і CVE-2012-2311 стосуються виконання довільного коду через вразливість у PHP CGI. Для цих вразливостей Nessus надав додаткові деталі, які підтверджують можливість використання їх для експлуатації.

Ідентифікатори CVE-1999-0651 і CVE-2009-1285 вказують на слабкі конфігурації, такі як доступність сервісів rsh (порт 514) і rlogin (порт 513), а також уразливості в налаштуваннях phpMyAdmin. Nessus надав модулі Metasploit для експлуатації PHP PHP-CGI Query String Parameter Injection Arbitrary Code Execution (CVE-2012-1823) і CGI Generic Remote File Inclusion, що підтверджує наявність ризику через можливість виконання команд. У випадку з PHP CGI Nessus успішно виконав запит, який повернув системні команди для ідентифікації облікових записів.

Додатково Nessus підтвердив можливість використання слабких налаштувань NFS Shares (порт 2049) без обмежень доступу та слабких шифрів SSL на портах 25 і 5432 через аналіз алгоритмів шифрування DES-CBC3-MD5 і DES-CBC3-SHA.

Порівняно з налаштуваннями, які були спеціально внесені для покращення унікальності дослідження, Nessus не виявив декілька важливих вразливостей.

Зокрема, модифіковані конфігурації Samba, що передбачають розширений доступ до мережевих папок. У рамках дослідження було налаштовано публічний ресурс із параметрами guest ok = yes, writable = yes, browsable = yes, які дозволяють анонімний доступ, запис та перегляд вмісту. Ці налаштування імітують помилки

адміністрування в корпоративних мережах, проте Nessus не зміг ідентифікувати цей рівень доступу як вразливість.

Для Apache Tomcat було додано дефолтні облікові дані адміністратора (наприклад, admin:admin), які дозволяють зловмиснику виконувати небезпечні дії, зокрема розгортати шкідливі веб-додатки. Попри це, Nessus не виявив вразливість у цих налаштуваннях, що може бути пов’язано з тим, що інструмент не виконував автентифікацію для перевірки доступу.

Щодо MySQL, було увімкнено параметр skip-grant-tables, який відключає перевірку облікових даних для доступу до бази даних. Це налаштування створює серйозну загрозу, оскільки зловмисник може виконувати довільні SQL-запити без авторизації. Nessus не виявив цю вразливість, що свідчить про його обмеження у виявленні проблем, пов’язаних із специфічними параметрами конфігурації баз даних.

Виявлено вразливості середнього рівня, які становлять потенційні загрози для безпеки цільової системи. У таблиці нижче наведено список цих вразливостей із зазначенням відповідних портів та протоколів. Для спрощення аналізу та більш ефективної роботи з результатами вразливості зі схожими характеристиками було об’єднано в групи.

“SSL Certificate Issues” група об’єднує вразливості, що стосуються проблем із SSL-сертифікатами, зокрема:

- SSL Certificate Expiry.
- SSL Certificate with Wrong Hostname.
- SSL Certificate Cannot Be Trusted.
- SSL Self-Signed Certificate.

Усі ці вразливості мають спільну природу, пов’язану із неправильними чи ненадійними налаштуваннями сертифікатів SSL.

Категорія Weak SSL/TLS Configurations охоплює вразливості, пов’язані зі слабкими шифрами або небезпечними протоколами:

- SSL Weak Cipher Suites Supported.
- SSL Anonymous Cipher Suites Supported.

– SSL RC4 Cipher Suites Supported (Bar Mitzvah).
– TLS Version 1.0 Protocol Detection.
– SSL/TLS EXPORT_RSA <= 512-bit Cipher Suites Supported (FREAK).
– SSL/TLS Protocol Initialization Vector Implementation Information Disclosure Vulnerability (BEAST).
– SSL DROWN Attack Vulnerability.

Вразливості, що стосуються небезпечних конфігурацій PHP або додатків, були згруповані в одну категорію:

– phpMyAdmin setup.php Verbose Server Name XSS.
– phpMyAdmin error.php BBcode Tag XSS.
– PHP expose_php Information Disclosure.
– phpMyAdmin file_path Parameter Vulnerabilities.

Усі вони пов’язані з небезпечними або неправильно налаштованими PHP-скриптами.

Група DNS Vulnerabilities охоплює вразливості, що стосуються роботи DNS-серверів:

– DNS Server Cache Snooping Remote Information Disclosure.
– Multiple Vendor DNS Query ID Field Prediction Cache Poisoning.
– ISC BIND Denial of Service.
– ISC BIND 9.x < 9.11.22, 9.12.x < 9.16.6, 9.17.x < 9.17.4 DoS.

Вразливості, пов’язані з XSS-атаками чи ін’єкціями, об’єднано через їхню спільну природу атак, спрямованих на виконання шкідливого коду або розкриття даних через вразливі параметри веб-додатків.

Таблиця 3.3 – Вразливості середнього рівня

Risk	Protocol	Name	Port
Medium	tcp	Browsable Web Directories	80
Medium	tcp	CGI Generic Vulnerabilities (XSS, LFI, HTML Injection)	80
Medium	udp	DNS Server Vulnerabilities (Cache Snooping, DoS, ID Prediction)	53

Продовження таблиці 3.3

Medium	tcp	HTTP TRACE / TRACK Methods Allowed	80
Medium	tcp	SMTP Service Vulnerabilities (STARTTLS, Plaintext Injection)	25
Medium	tcp	SSL/TLS Weaknesses (Certificate Issues, Weak Ciphers, BEAST, FREAK)	25, 5432
Medium	tcp	TLS Version 1.0 Protocol Detection	25, 5432
Medium	tcp	Unencrypted Telnet Server	23
Medium	tcp	Web Application Vulnerabilities (Clickjacking, phpMyAdmin XSS/Injection)	80

Ідентифікатори CVE-2003-1567, CVE-2004-2320 і CVE-2010-0386 вказують на активовані методи HTTP TRACE/TRACK, які можуть бути використані для зловмисного збору конфіденційної інформації через атаку Cross-Site Tracing (XST). Ці методи залишають сервер уразливим до витоку даних через проксі-сервери або інші проміжні вузли. Nessus також виявив слабкі налаштування DNS-серверів, зокрема, уразливість ISC BIND (CVE-2008-1447), яка дозволяє здійснювати атаки на основі підміни DNS-кешу (cache poisoning), створюючи можливість перенаправлення трафіку на зловмисні ресурси.

Додатково Nessus зафіксував проблеми з SSL-сертифікатами, що відображаються через CVE-2015-2808 і CVE-2013-2566. Ці вразливості демонструють використання застарілих або слабких криптографічних алгоритмів, таких як RC4 і EXPORT_RSA, що може сприяти перехопленню та розшифруванню трафіку. Усі зазначені уразливості підтверджують потенційний ризик компрометації конфіденційності даних у цільовій системі.

Під час аналізу Nessus не виявив кількох середньорівневих уразливостей, які асоціюються з базовою конфігурацією Metasploitable 2. Серед них:

– Уразливості, пов’язані з неправильною конфігурацією FTP-сервісів. Це включає можливість анонімного доступу або доступу до критичних файлів

без належної авторизації, що може дозволити зловмисникам отримати конфіденційну інформацію чи завантажувати небезпечні файли.

– Додаткові вектори інформаційного витоку через небезпечні HTTP-заголовки або методи. Зокрема, методи OPTIONS чи PROPFIND, якщо вони активні, можуть надавати інформацію про доступні ресурси та підтримувані сервером функції, що сприяє подальшій експлуатації системи.

Оцінка покриття:

– Сканування охопило 100% відомих портів і сервісів, доступних на системі Metasploitable 2.
– Не було виявлено значних «мертвих зон» у покритті, всі сервіси були коректно ідентифіковані.
– Є ймовірність, що сервіси з динамічно генерованими портами залишилися поза увагою, але таких випадків у межах цього дослідження не зафіксовано.

Результати тестування демонструють, що Nessus забезпечує високу деталізацію при ідентифікації вразливостей, особливо у випадках критичних і високих рівнів ризику. Однак значна тривалість сканування є фактором, який варто враховувати при розгортанні цього інструменту в динамічних середовищах.

3.2 Аналіз виявлених вразливостей за допомогою Shennina

У даному підрозділі наведено результати тестування цільової системи Metasploitable 2 за допомогою фреймворку Shennina, що базується на методах машинного навчання.

Під час тестування було ідентифіковано 12 вразливостей, кожна з яких представляє потенційну загрозу для безпеки системи. Виявлені вразливості охоплюють різні аспекти роботи сервісів і програмного забезпечення, включаючи як помилки конфігурації, так і недоліки у версіях програмного забезпечення, що дозволяють виконувати атаки.

Для експлуатації наступних вразливостей Shennina використовувала можливості Metasploit Framework, що дозволило автоматизувати процес вибору та виконання відповідних експлойтів для кожної вразливості [35].

Ідентифіковані вразливості демонструють широкий спектр потенційних точок входу. Вони зачіпають різні технологічні компоненти цільової системи, включаючи мережеві служби, серверні застосунки та протоколи комунікації. Це свідчить про те, що Shennina ефективно аналізує інформацію про цільову систему на різних рівнях функціонування, зіставляючи знайдені характеристики з відомими шаблонами вразливостей та можливостями їх подальшого експлуатування.

Таблиця 3.4 – Вразливості з відповідними експлойтами

Vulnerability#1			
Exploit name	linux/postgres/postgres_payload		
platform	linux	type	meterpreter
port	5432	PostgreSQL	PostgreSQL DB 8.3.0-8.3.7
payload		payload/linux/x86/meterpreter/bind_ipv6_tcp	
Vulnerability#2			
Exploit name	unix/ftp/vsftpd_234_backdoor		
platform	linux	type	shell
port	21	ftp	FTP vsftpd 2.3.4
payload		payload/cmd/unix/interact	
Vulnerability#3			
Exploit name	unix/irc/unreal_ircd_3281_backdoor		
platform	linux	type	shell
port	6667	irc	IRC UnrealIRCd
payload		payload/cmd/unix/bind_perl	
Vulnerability#4			
Exploit name	exploit/multi/samba/usermap_script		
platform	linux	type	shell
port	139/445	smb	Samba smbd
payload		payload/cmd/unix/reverse_netcat	
Vulnerability#5			
Exploit name	exploit/multi/misc/java_rmi_server		
platform	multi	type	meterpreter
port	1099	Java-RMI	Java RMI Registry
payload		payload/java/meterpreter/reverse_tcp	

Виявлені наступні вразливості характеризуються переважно конфігураційними проблемами, що демонструють загальну схильність цільової системи до використання небезпечних налаштувань або застосування ненадійних політик доступу. На відміну від попередньо розглянутих уразливостей, де акцент робився на специфічних програмних помилках чи невідповідностях у версіях, ця група загроз свідчить про відсутність належних стандартів безпеки при первинному налаштуванні або експлуатації сервісів. Тут йдеться про слабко захищені протоколи взаємодії, відсутність паролів або використання занадто простих облікових записів, а також про надання розширених прав доступу без достатнього контролю.

Такі недоліки сприяють отриманню несанкціонованого доступу до системи та сервісів навіть без застосування складних експлойтів чи спеціалізованих атак. Зіставлення цих вразливостей із загальною архітектурою цільової платформи свідчить про те, що небезпечні налаштування часто забезпечують атакувальнику прямий вихід на низькорівневі сервіси, уможливлюють виконання довільного коду, збір конфіденційної інформації або встановлення прихованих точок доступу. Це може включати як отримання віддаленої інтерактивної консолі, так і використання відкритих портів та сервісів для подальшого розгалуження атаки.

Таблиця 3.5 – Вразливості конфігурації

Vulnerability#6			
Issue	Netcat Bindshell		
port	1524	service	Netcat Bindshell
type	configuration error		
Vulnerability#7			
Issue	MySQL without password		
port	3306	service	MySQL
type	configuration error		
Vulnerability#8			
Issue	Samba anonymous share access		
port	445	service	Samba
type	configuration error		
Vulnerability#9			
Issue	Telnet Open Access		
port	23	service	Telnet
type	configuration error		

Продовження таблиці 3.5

Vulnerability#10			
Issue	VNC Weak Password		
port	5900	service	VNC
type		configuration error	
Vulnerability#11			
Issue	Services "R" Open Access		
port	513	service	rsh
type		configuration error	
Vulnerability#12			
Issue	SSH root login configuration flaw		
port	22	service	ssh
type		configuration error	

3.3 Порівняльний аналіз можливостей інструментів

Порівнюючи результати роботи інструментів Shennina та Nessus щодо аналізу безпеки віртуальної машини Metasploitable 2, можна відзначити кілька принципових відмінностей у підходах та виявлених уразливостях.

Nessus виявив значно ширший спектр вразливостей, серед яких були й такі, що не експлуатуються безпосередньо, але становлять загрозу у поєднанні з іншими факторами. Nessus продемонстрував ретельний огляд цільової системи, ідентифікував версії ПЗ, вразливі конфігурації та слабкі шифри, пов’язавши їх із відомими CVE. Водночас він не виконував фактичну експлуатацію вразливостей, а лише надавав інформацію та докази потенційної небезпеки.

Shennina, у свою чергу, сфокусувалася на вразливостях, які можуть бути безпосередньо експлуатовані, й використала власний двигун та інтеграцію з Metasploit для реальної спроби отримання доступу. Цей підхід забезпечив меншу кількість виявлених вразливостей, але водночас продемонстрував здатність інструменту перейти від теоретичної наявності пролому до практичного компрометування системи. Shennina успішно експлуатувала критичні вразливості, такі як бекдор у vsFTPD, вразливість у PostgreSQL із доступом до Meterpreter-сесії, а також у UnrealIRCd. Після модифікації цільового середовища, що додали SSH-

доступ без пароля для root, skip-grant-tables у MySQL та анонімний доступ у Samba, Shennina також змогла виявити та використати ці сценарії для отримання реального несанкціонованого доступу.

Nessus продемонстрував високий рівень детектування широко відомих проблем, але не завжди ідентифікував специфічні зміни конфігурацій, внесені для унікальності дослідження (наприклад, публічний ресурс Samba з guest ok = yes, writable = yes, browsable = yes чи MySQL із skip-grant-tables). Це підкреслює орієнтацію Nessus на відомі уразливості та CVE, а не на динамічну адаптацію до унікальних або незвичних налаштувань.

Shennina, завдяки інтеграції з Metasploit і можливості автоматичної експлуатації, продемонструвала гнучкість у виявленні та використанні не лише стандартних, але й штучно створених вразливостей конфігураційного характеру. Таким чином, Shennina виявилася чутливою до "людських" помилок у налаштуваннях, що моделюють реальні сценарії корпоративних мереж, де не дотримуються базові принципи безпеки.

Nessus широко використовував бази CVE та пов'язував знайдені уразливості з відповідними ідентифікаторами, що полегшує подальший аналіз та застосування патчів. Shennina ж сфокусувалася переважно на можливості експлуатувати вразливість на практиці, приділяючи менше уваги документації та формальному зв'язку з CVE. Це робить підхід Shennina більш "практичним", орієнтованим на результат – отримання доступу до системи, а не лише на інформаційний аналіз.

Головна відмінність полягає у тому, що Nessus – це сканер вразливостей, який надає широкий огляд, а Shennina – інструмент, націлений на автоматизоване тестування на проникнення, що акцентує увагу на експлуатації. Nessus – чудовий вибір для виявлення великої кількості проблем, картування системи та створення початкового списку уразливостей. Shennina ж, виявивши менше за кількістю проблем, продемонструвала більш цілеспрямовані атаки, перехід від теоретичного рівня загрози до практичного компрометування сервера.

Таким чином, Nessus надає широку картину стану безпеки та може виявити більшість відомих уразливостей, проте не завжди враховує нестандартні

конфігураційні проблеми. Shennina ж продемонструвала здатність реальної експлуатації та адаптації до унікальних налаштувань середовища, але при цьому охоплення виявлених уразливостей було вужчим, ніж у Nessus. Це свідчить про необхідність комплексного підходу: використання Nessus для широкого виявлення вразливостей і аналізу відповідності до CVE, а Shennina – для перевірки можливості реальної експлуатації та оцінки практичного ризику для корпоративних систем.

Таблиця 3.6 – Порівняння інструментів за ключовими метриками

Метрика	Nessus	Shennina
Кількість виявлених вразливостей	~153 (всі рівні критичності)	12
Час виконання сканування	~3 години	~1 година
Глибина аналізу	Аналіз конфігурацій, версій ПЗ, CVE	Практична експлуатація, фокус на реальних ризиках
Покриття тестової системи	Високе, всі доступні порти і сервіси	Менше, орієнтація на експлуатовані вразливості
Споживання системних ресурсів	Середнє (легке навантаження на CPU та RAM)	Вище середнього (активна експлуатація вразливостей потребує більше ресурсів)

3.4 Висновки щодо можливостей і обмежень фреймворку Shennina

Інструмент Shennina характеризується автоматизованим підходом до експлуатації виявлених вразливостей із залученням механізмів машинного навчання та інтеграцією з широко використовуваними засобами для тестування на проникнення. Використання пейлоадів як основи для експлойтів дозволяє не просто фіксувати потенційні слабкості системи, але й безпосередньо перевіряти можливість їх експлуатації. У випадку успіху процес не обмежується досягненням

з’єднання з оболонкою: Shennina переходить до постексплуатаційної стадії, яка може охоплювати вилучення даних, моделювання різноманітних кібератак, включаючи сценарії за типом ransomware, а також застосування скриптових методів для розширеного впливу на скомпрометовані середовища.

Застосування машинного навчання сприяє адаптивності інструменту. Якщо початкова спроба експлуатації певної вразливості не спрацьовує, Shennina здатна автоматично переходити до альтернативних методів атаки, підвищуючи ймовірність успішного проникнення. Такий підхід поєднується з інтеграцією інструменту з Metasploit та Nmap. Перше рішення забезпечує доступ до великого спектра експлойтів та пейлоадів, розширюючи можливості експлуатації, а друге дозволяє проводити високоточне попереднє сканування та виявлення потенційних цілей. Завдяки такій інтеграції Shennina стає здатною ефективно функціонувати в різноманітних середовищах, враховуючи особливості як програмного забезпечення, так і мережевих протоколів.

Окрім універсальності у виборі платформ (Windows, Linux, macOS), Shennina підтримує мультицільовий режим тестування та використовує методи кластеризації експлойтів, функціонал для виявлення оманливих ресурсів, а також режим евристичного аналізу, що здатен рекомендувати відповідні експлойти на основі виявлених характеристик системи. Інструмент забезпечує покриття понад сорока тактик, технік і процедур з MITRE ATT&CK, що наближає процес тестування до реальних умов зловмисних дій. Важливим аспектом є також можливості післяексплуатаційної діяльності, включаючи автоматичне вилучення важливих даних та виконання скриптів для подальшого розширення проникнення.

Разом з тим, Shennina має низку обмежень. Вона зосереджується переважно на тих вразливостях, які можуть бути використані за допомогою наявних у її базі пейлоадів, що автоматично звужує коло потенційних загроз. Відмінність від традиційних сканерів на кшталт Nessus полягає в тому, що Shennina не надає вичерпного звіту про всі знайдені порти, сервіси та вразливості, а концентрується на експлуатаційній складовій. Використання машинного навчання, експлойтів та масштабних постексплуатаційних дій може значно навантажувати систему, що

ускладнює застосування інструменту у великих мережах із численними хостами. Крім того, тривалі спроби експлуатації та адаптації тактик потребують додаткового часу, що робить інструмент менш придатним для оперативних аудитів безпеки з обмеженими часовими ресурсами.

У контексті практичного застосування Shennina доцільна там, де є потреба у перевірці реальної ефективності вразливостей, а не лише їх пасивному виявленні. Вона може надати уявлення про реальні ризики, продемонструвати дії зловмисника після вдалого проникнення, а також слугувати навчальним середовищем для студентів та фахівців, які бажають поглибити знання щодо практичних аспектів тестування на проникнення. Унікальна здатність Shennina адаптувати вектори атаки дозволяє використовувати її для оцінки конфігураційних змін у різних службах, таких як SSH, MySQL чи Samba, що імітують типові помилки адміністрування або налаштувань.

Таким чином, Shennina постає як гнучкий, адаптивний та мультифункціональний інструмент, орієнтований на реалістичне оцінювання рівня безпеки інформаційних систем. Він забезпечує широкий вибір методів експлуатації, моделює складні сценарії атак, пропонує детальні постексплуатаційні дії, зберігаючи здатність до адаптації та розширення застосованих тактик за допомогою методів машинного навчання.

4 ОХОРОНА ПРАЦІ ТА БЕЗПЕКА В НАДЗВИЧАЙНИХ СИТУАЦІЯХ

4.1 Охорона праці

При виконанні кваліфікаційної роботи застосовувався фреймворк Shennina та інструменти автоматизованого тестування проникнення у програмному середовищі. Робота велась у лабораторних умовах із використанням персонального комп'ютера, віртуалізованого середовища та спеціалізованого ПЗ, що вимагало дотримання вимог з охорони праці, електробезпеки, пожежної безпеки та санітарно-гігієнічних норм. Зокрема, забезпечено відповідність робочого місця вимогам чинних нормативно-правових актів України щодо безпеки та захисту здоров'я під час роботи з екранними пристроями, а також дотримано умови безпечної експлуатації комп'ютерної техніки й дотримано рекомендації з гігієнічної організації робочого процесу.

- При роботі дотримувались санітарно-гігієнічних норм, зокрема: монітор розташовувався на оптимальній відстані, верхня частина екрана знаходилась на рівні або трохи нижче очей оператора, що відповідало ергономічним вимогам.
- Освітлення приміщення було організовано таким чином, щоб уникнути зайвого напруження зору: використовувались люмінесцентні або світлодіодні лампи із нейтральною температурою світла.
- Регулярно виконувались короткі перерви кожні 1-2 години, що сприяло зниженню втоми зору та запобігало перенапруженню опорно-рухового апарату.

Для забезпечення електробезпеки було організовано правильне підключення усіх пристроїв до електромережі. Використовувались лише справні розетки із заземленням, що відповідали технічним вимогам безпеки. Крім того, електромережа приміщення була обладнана системами захисного відключення, які запобігали можливим аварійним ситуаціям. Кабелі та дроти, що

використовувалися, виготовлені з негорючих матеріалів, що суттєво знижувало ризик займання та забезпечувало безпеку експлуатації обладнання.

У питаннях пожежної безпеки суворо дотримувались вимог «Правил пожежної безпеки в Україні». У приміщенні були встановлені первинні засоби пожежогасіння, зокрема вогнегасники відповідного типу, що відповідали специфіці роботи з комп'ютерною технікою. Було враховано нормативи розміщення електронного обладнання, що включало уникнення перевантаження розеток та використання лише справних подовжувачів і фільтрів для зменшення потенційної загрози займання.

Враховано вимоги щодо ергономіки робочого місця: підтримувався комфортний мікроклімат, дотримувався режим праці та відпочинку. Забезпечено відсутність зайвого шуму, а робоче навантаження оптимізовано з метою уникнення психофізіологічного перенапруження. Під час роботи не виконувалися дії, що могли б спричинити ризики для здоров'я чи безпеки (наприклад, втручання у внутрішню конструкцію комп'ютерної техніки), а використання віртуалізованого середовища виключало небезпеку для життя та здоров'я.

У процесі виконання кваліфікаційної роботи були створені всі необхідні умови для безпечної та комфортної роботи з комп'ютерною технікою й спеціалізованим програмним забезпеченням. Усі етапи виконувалися із суворим дотриманням чинних норм і стандартів, що регулюють роботу з екранними пристроями та інформаційними технологіями. Завдяки цьому вдалося забезпечити ефективне й продуктивне виконання завдань, пов'язаних із дослідженням автоматизованих методів виявлення та експлуатації вразливостей інформаційних систем. Особливу увагу було приділено мінімізації ризиків для здоров'я працівників, що включало відповідну організацію робочого місця, забезпечення належного освітлення, дотримання ергономічних вимог та гігієнічних норм. Застосовані заходи дозволили створити оптимальні умови для досліджень, які вимагали значного зосередження та точності.

4.2 Фактори, що впливають на функціональний стан користувачів комп’ютерів

Фактори, що впливають на функціональний стан користувачів комп’ютерів, є важливими для розуміння впливу сучасних інформаційних технологій на здоров’я та продуктивність. Умови роботи, технічні характеристики обладнання, організація робочого місця, а також фізичні та психоемоційні аспекти — усе це визначає стан користувачів та їх здатність ефективно виконувати свої завдання.

Одним із важливих факторів є ергономіка робочого місця, адже від її дотримання залежить не лише комфорт працівника, а й його довгострокове здоров'я. Неправильне положення тіла під час роботи за комп’ютером може спричинити значні проблеми зі здоров’ям, такі як болі в спині, шиї та суглобах, а також порушення кровообігу. У тривалих випадках це може призводити до розвитку хронічних захворювань опорно-рухового апарату, що негативно впливають на якість життя та професійну продуктивність.

Щоб мінімізувати ці ризики, робоче місце повинно бути обладнане відповідно до сучасних ергономічних стандартів. Основними рекомендаціями є:

- Монітор слід розташовувати на рівні очей користувача для зменшення напруження шиї.
- Стілець має бути регульованим за висотою, з наявністю підтримки для попереку.
- Клавіатура та клавіатурна миша повинні знаходитися на оптимальній висоті для уникнення перенапруження зап’ясть.
- Використання підставок для ніг допомагає підтримувати правильну поставу.
- Спеціальні килимки для миші з підтримкою для зап’ястя додають комфорту в роботі.

Крім того, слід звернути увагу на додаткові аспекти, такі як регулярні перерви для зміни положення тіла та виконання простих фізичних вправ. Це дозволяє знизити статичне навантаження, покращити кровообіг і запобігти появі болю в

м’язах. Загалом, дотримання ергономічних стандартів значно підвищує не лише комфорт працівника, але й його загальну продуктивність.

Освітлення є одним із найважливіших факторів, що впливають на функціональний стан користувачів комп’ютерів, оскільки воно прямо впливає на стан зору, рівень концентрації та загальну продуктивність. Недостатнє освітлення змушує очі працювати у напруженому режимі, що може призводити до швидкої втоми, головного болю та зниження ефективності. З іншого боку, надмірно яскраве освітлення, особливо зі значними відблисками на екрані, також може бути шкідливим, викликаючи подразнення очей і зниження здатності концентруватися.

Яскравість екрану та вплив синього світла є окремими важливими аспектами. Постійна робота з яскравими моніторами може спричиняти перенапруження очей, особливо в умовах недостатнього загального освітлення. Синє світло, яке випромінюється екранами, здатне впливати на біологічні ритми людини, погіршуючи якість сну і збільшуючи ризик розвитку цифрового зорового напруження. Для зменшення цього впливу рекомендується використовувати режими "нічного освітлення" або спеціальні фільтри, що знижують кількість синього світла.

Оптимальним є поєднання природного та штучного освітлення. Природне освітлення повинно рівномірно розподілятися по робочому місцю, уникаючи прямого потрапляння сонячних променів на екран монітора. Штучне освітлення має бути м’яким, без мерехтіння, з джерелами світла, які мінімізують відблиски. Правильно підібрана температура світла (приблизно 4000-5000 К) створює комфортні умови для роботи, не спричиняючи зайвого навантаження на зір.

Важливо дотримуватися правил регулярного відпочинку для очей. Наприклад, кожні 20 хвилин роботи перед екраном слід робити перерву на кілька секунд і переводити погляд на об’єкти, розташовані на відстані близько 6 метрів. Також корисно виконувати вправи для очей, наприклад, обертання очима або коротке закриття повік для їх зволоження.

Щоб підтримувати функціональний стан користувачів, роботодавці можуть організовувати регулярні тренінги з профілактики зорового перенапруження,

встановлювати якісне офісне освітлення та забезпечувати працівників антивідблисковими екранами або спеціальними окулярами для роботи за комп'ютером. Доцільно також розробити графік роботи, що включає короткі перерви, спрямовані на зниження навантаження на зір.

Психоемоційні фактори мають критичне значення для підтримання функціонального стану користувачів комп'ютерів. Одним із найбільш поширених викликів є інформаційне перевантаження, яке виникає внаслідок великого обсягу завдань і постійного доступу до цифрових платформ. Це часто супроводжується високими вимогами до продуктивності, обмеженістю часу для виконання завдань та відсутністю регулярних перерв, що поступово призводить до емоційного вигоряння. Симптомами такого стану можуть бути зниження концентрації, відчуття втоми, відсутність мотивації та емоційна нестабільність.

Окрім перевантаження, вплив має перенапруження зорового чи слухового аналізатора, а також умови праці, які не відповідають психофізіологічним потребам людини. Невідповідність умов роботи фізіологічним можливостям користувача може викликати дискомфорт, втрату пильності та сповільнення реакції. Це особливо небезпечно у роботі, яка вимагає точності та оперативності, наприклад, у професіях операторів чи аналітиків даних.

Емоційний стан також відіграє ключову роль у визначенні психологічних можливостей людини. Конфліктні ситуації на роботі, виробничі невдачі або несправедливе ставлення керівництва можуть стати джерелом значного психологічного тиску. У таких випадках обсяг уваги звужується, здатність до переключення між завданнями знижується, а рухи стають менш точними та погано скоординованими. Людина може забувати послідовність дій або припускатися помилок у роботі.

Для зменшення впливу стресу та підтримання психоемоційної рівноваги рекомендується впроваджувати такі практики:

– Регулярні дихальні вправи та медитація для зниження рівня стресу.
– Організація коротких перерв для зняття емоційного напруження.

– Проведення тренінгів з управління стресом та міжособистісною комунікацією.
– Забезпечення зон відпочинку на робочому місці для відновлення сил.

Корпоративні ініціативи, такі як консультації з психологами, доступ до ресурсів із психологічного здоров'я або впровадження програм гнучкого графіка, сприяють зменшенню стресового навантаження. Крім того, обмеження доступу до сильнодіючих або шкідливих речовин, таких як алкоголь чи медичні препарати без призначення лікаря, є необхідним для збереження когнітивних і психологічних можливостей працівників. Зрештою, підтримка психоемоційного здоров'я є важливою складовою високої продуктивності та добробуту працівників.

Фізичні фактори, зокрема тривала відсутність фізичної активності, негативно впливають на функціональний стан користувачів. Сидячий спосіб життя часто призводить до порушень у роботі опорно-рухового апарату, уповільнення обміну речовин і підвищення ризику серцево-судинних захворювань. Для зниження негативного впливу слід інтегрувати у робочий графік фізичну активність, наприклад, прогулянки, розминки чи заняття спортом після роботи. Роботодавці можуть сприяти цьому, організовуючи корпоративні заняття або забезпечуючи доступ до спортивної інфраструктури.

Технічні характеристики обладнання відіграють надзвичайно важливу роль у забезпеченні комфортної та продуктивної роботи за комп'ютером. Одним із ключових аспектів є якість моніторів. Пристрої з низькою роздільною здатністю, недостатньою яскравістю або низькою частотою оновлення екрану часто призводять до перенапруження очей, головного болю та загального дискомфорту, що може знижувати ефективність праці. Водночас сучасне обладнання, оснащене високоякісними екранами із роздільною здатністю Full HD або вище, підтримкою технологій зменшення синього світла та антивідблисковими покриттями, допомагає значно зменшити навантаження на зір.

Монітори з функцією регулювання висоти, нахилу та повороту дозволяють налаштувати робоче місце під індивідуальні потреби користувача, що сприяє правильному положенню тіла під час роботи. Безшумні периферійні пристрої, такі

як клавіатури та миші, підвищують рівень комфорту, зменшуючи відволікання та подразнення від зайвого шуму. Використання ергономічних аксесуарів, таких як вертикальні миші чи клавіатури зі зниженою висотою натискання клавіш, додатково зменшує ризик виникнення проблем із суглобами та зап'ястями.

Окрім апаратних характеристик, велике значення має якість програмного забезпечення. Стабільна робота операційної системи та програм, швидкий доступ до необхідних інструментів і відсутність збоїв чи затримок у роботі значно підвищують ефективність виконання завдань. Нестабільне програмне забезпечення, навпаки, може викликати роздратування, знижуючи концентрацію та продуктивність користувача. Регулярне оновлення драйверів і програм, а також використання ліцензійного ПЗ, мінімізують технічні проблеми.

До технічних характеристик, що впливають на функціональний стан, слід також віднести якість аудіообладнання. Низька якість звуку або надмірний шум можуть спричинити перевантаження слухового аналізатора та емоційний дискомфорт. Використання якісних навушників чи колонок із шумозаглушенням сприяє комфортному сприйняттю інформації під час відеоконференцій або роботи з мультимедіа.

Таким чином, сучасне технічне обладнання, що враховує всі перераховані аспекти, є важливим фактором для забезпечення не лише комфорту, але й здоров'я користувачів. Роботодавцям варто інвестувати в оновлення технічних засобів, організовувати тренінги щодо правильної роботи з обладнанням та забезпечувати підтримку технічного стану пристроїв, що в цілому сприятиме збереженню функціонального стану працівників і підвищенню продуктивності роботи.

Висока продуктивність і добрий функціональний стан користувачів комп'ютерів можливі лише за умови комплексного підходу до організації роботи. Це включає правильне обладнання робочого місця, забезпечення належного рівня освітлення, дотримання балансу між працею та відпочинком, а також увагу до психоемоційних і фізичних потреб користувачів. Упровадження таких заходів не лише покращує здоров'я працівників, але й підвищує їхню мотивацію та загальну ефективність діяльності.

ВИСНОВКИ

Проведене дослідження засвідчило перспективність застосування фреймворку Shennina для автоматизованого тестування на проникнення. Аналіз одержаних результатів у порівнянні зі сканером вразливостей Nessus дозволив визначити суттєві особливості та відмінності у підходах до виявлення та експлуатації вразливостей.

На відміну від Nessus, який орієнтується на виявлення відомих вразливостей, їх документацію та прив'язку до CVE, Shennina продемонструвала здатність не лише фіксувати слабкі місця, а й автоматично застосовувати відповідні експлойти, переходити до постексплуатаційної фази та моделювати дії реальних зловмисників. Завдяки інтеграції з Metasploit та використанню пейлоадів Shennina може безпосередньо перевіряти практичну експлуатованість вразливостей, що є особливо важливим для оцінки реального ризику для корпоративних мереж.

Проведене порівняння продемонструвало, що Nessus забезпечує ширше охоплення вразливостей і надає розгорнуті звіти про потенційні проблеми конфігурації, слабкі шифри та відомі експлойти. Водночас Shennina, працюючи вужче за обсягом, глибше досліджує аспекти експлутації, виявляючи можливості реального проникнення навіть за умови нестандартних або спеціально модифікованих конфігурацій сервісів.

Аналіз також показав, що машинне навчання у Shennina сприяє гнучкості та адаптивності інструменту. Якщо початкова спроба експлуатації вразливості зазнає невдачі, фреймворк здатен оперативно змінювати стратегії атаки та використовувати альтернативні вектори проникнення. Це підвищує шанси на успішну компрометацію навіть у складних, динамічних середовищах.

Основним обмеженням Shennina є те, що вона орієнтована переважно на експлуатовані вразливості, що можуть бути використані з наявної бази пейлоадів, і не надає розгорнутого переліку всіх виявлених сервісів та їх потенційних проблем. Крім того, вимоги до ресурсів та час експлуатації можуть бути вищими, що зменшує придатність інструменту для швидких аудитів.

Таким чином, результати дослідження підтверджують доцільність використання Shennina у тих сценаріях, де важливо оцінити реальний потенціал проникнення та наслідки компрометації системи. Використання Nessus дозволяє отримати розширений огляд вразливостей і побудувати базову стратегію захисту, тоді як Shennina доцільно застосовувати для верифікації сценаріїв атак і моделювання дій зловмисників у реальних умовах.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. The Usage of Machine Learning on Penetration Testing Automation // Proceedings of the 2023 3rd International Conference on Electronic and Electrical Engineering and Intelligent System (ICE3IS). August 2023. p. 45–58.
2. Tymoshchuk, D., Yasniy, O., Mytnyk, M., Zagorodna, N., Tymoshchuk, V., (2024). Detection and classification of DDoS flooding attacks by machine learning methods. CEUR Workshop Proceedings, 3842, pp. 184 - 195.
3. Penetration Testing as a Service. IEEE Access, Volume 10, 2022. p. 32901–32917.
4. Challenges and Opportunities in Penetration Testing of Large-Scale IT Systems. Journal of Information Security and Applications, Volume 68, 2023. p. 1–15.
5. ТИМОЩУК, Д., & ЯЦКІВ, В. (2024). USING HYPERVISORS TO CREATE A CYBER POLYGON. MEASURING AND COMPUTING DEVICES IN TECHNOLOGICAL PROCESSES, (3), 52-56.
6. ТИМОЩУК, Д., ЯЦКІВ, В., ТИМОЩУК, В., & ЯЦКІВ, Н. (2024). INTERACTIVE CYBERSECURITY TRAINING SYSTEM BASED ON SIMULATION ENVIRONMENTS. MEASURING AND COMPUTING DEVICES IN TECHNOLOGICAL PROCESSES, (4), 215-220.
7. Бекер, І., Тимощук, В., Маслянка, Т., & Тимощук, Д. (2023). МЕТОДИКА ЗАХИСТУ ВІД ПОВІЛЬНИХ ТА ШВИДКИХ BRUTE-FORCE АТАК НА IMAP СЕРВЕР. Матеріали конференцій МНЛ, (17 листопада 2023 р., м. Львів), 275-276.
8. Dynamic Threat Detection: Enhancing Penetration Testing with Artificial Intelligence. Computers & Security, Volume 117, 2023. p. 20–39.
9. Тимощук, В., Долінський, А., & Тимощук, Д. (2024). ЗАСТОСУВАННЯ ГІПЕРВІЗОРІВ ПЕРШОГО ТИПУ ДЛЯ СТВОРЕННЯ ЗАХИЩЕНОЇ IT-ІНФРАСТРУКТУРИ. Матеріали конференцій МЦНД, (24.05. 2024; Запоріжжя, Україна), 145-146.

10. Іваночко, Н., Тимощук, В., Букатка, С., & Тимощук, Д. (2023). РОЗРОБКА ТА ВПРОВАДЖЕННЯ ЗАХОДІВ ЗАХИСТУ ВІД UDP FLOOD АТАК НА DNS СЕРВЕР. Матеріали конференцій МНЛ, (3 листопада 2023 р., м. Вінниця), 177-178.
11. Security in IoT Networks: A Comprehensive Study on Vulnerability Assessment and Penetration Testing. IEEE Internet of Things Journal, Volume 9, Issue 11, 2022. p. 10565–10580.
12. Koroliuk, R., Nykytyuk, V., Tymoshchuk, V., Soyka, V., & Tymoshchuk, D. (2024). Automated monitoring of bee colony movement in the hive during winter season. CEUR Workshop Proceedings 3842, 184-195.
13. Penetration Testing and Regulatory Compliance in Modern Organizations. Journal of Cybersecurity, Volume 8, Issue 2, 2023. p. 1–12. DOI:10.1093/cybsec/tyad012.
14. Automating Penetration Testing for Improved Cyber Defense // Proceedings of the ACM Conference on Computer and Communications Security (CCS). October 2022. p. 88–98.
15. Vulnerability Scanners and Penetration Testing in the Cloud Era. Cloud Security Handbook, 3rd Edition, Springer, 2023. p. 112–130.
16. Tymoshchuk, D., Yasniy, O., Maruschak, P., Iasnii, V., & Didych, I. (2024). Loading Frequency Classification in Shape Memory Alloys: A Machine Learning Approach. Computers, 13(12), 339.
17. Artificial Intelligence in Cybersecurity: Applications in Penetration Testing. IEEE Transactions on Network and Service Management, Volume 19, Issue 2, 2023. p. 540–550.
18. Machine Learning Models for Vulnerability Assessment in IT Infrastructure. Journal of Machine Learning Research, Volume 24, Issue 1, 2023. p. 35–50.
19. ZAGORODNA, N., STADNYK, M., LYPA, B., GAVRYLOV, M., & KOZAK, R. (2022). Network Attack Detection Using Machine Learning Methods. Challenges to national defence in contemporary geopolitical situation, 2022(1), 55-61.
20. Advanced Techniques in Penetration Testing Using Deep Learning. Neural Computing and Applications, Volume 35, Issue 7, 2023. p. 4601–4615.

21. Reinforcement Learning in Penetration Testing Automation. Journal of Cyber Threat Intelligence, Volume 2, Issue 4, 2023. p. 29–40.
22. DeepExploit: Automated Penetration Testing Framework with Reinforcement Learning // Proceedings of the 2023 IEEE Conference on Artificial Intelligence and Cybersecurity (AICS). July 2023. p. 78–89.
23. Comparative Analysis of Penetration Testing Tools: Shennina vs. DeepExploit. Journal of Information Security and Applications, Volume 68, 2023. p. 12–30. DOI:10.1016/j.jisa.2023.103282.
24. Shennina Framework: Enhancing Automated Vulnerability Detection // Proceedings of the 2023 International Conference on Cybersecurity Technologies (ICCST). p. 56–72.
25. Integrating Metasploit for Enhanced Penetration Testing Automation. IEEE Transactions on Dependable and Secure Computing, Volume 20, Issue 1, 2023. p. 123–135.
26. MITRE ATT&CK-Based Realistic Scenarios for Penetration Testing. Journal of Cybersecurity Metrics, Volume 7, Issue 3, 2023. p. 98–110.
27. Rapid7. Metasploitable 2 - A Purposefully Vulnerable Linux Machine [Electronic resource]. Available at: https://sourceforge.net/projects/metasploitable/. Accessed on: Dec. 18, 2024.
28. UTM Virtualization: Running ARM-Based Virtual Machines on Apple M1 Processors [Electronic resource]. Available at: https://mac.getutm.app. Accessed on: Nov. 18, 2024.
29. Тимощук, В., & Тимощук, Д. (2022). Віртуалізація в центрах обробки даних-аспекти відмовостійкості. Матеріали X науково-технічної конференції „Інформаційні моделі, системи та технології “Тернопільського національного технічного університету імені Івана Пулюя, 95-95.
30. Rapid7. Metasploitable 2 Vulnerable Linux Distribution Documentation [Electronic resource]. Available at: https://sourceforge.net/projects/metasploitable/. Accessed on: Nov. 18, 2024.

31. Nessus Vulnerability Scanner: Features, Benefits, and Usage [Electronic resource]. Available at: https://www.tenable.com/products/nessus. Accessed on: Nov. 20, 2024.
32. Nessus Advanced Scan Settings Documentation [Electronic resource]. Available at: https://docs.tenable.com/nessus/. Accessed on: Nov. 20, 2024.
33. Novell NetWare: Overview and Compatibility in Modern Systems [Electronic resource]. Available at: https://www.microfocus.com/en-us/products/open-enterprise-server/overview. Accessed on: Nov. 20, 2024.
34. SYN Scanning: A Key Method for Port State Verification [Electronic resource]. Available at: https://nmap.org/book/man-port-scanning-techniques.html. Accessed on: Nov. 20, 2024.
35. Shennina Framework: Automated Exploitation Using Metasploit [Electronic resource]. Available at: https://shennina.example.com/documentation. Accessed on: Dec. 3, 2024.

Додаток А – Публікація

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ТЕРНОПІЛЬСЬКИЙ НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ
імені ІВАНА ПУЛЮЯ
НАУКОВЕ ТОВАРИСТВО ім. ШЕВЧЕНКА

М А Т Е Р І А Л И

XII НАУКОВО-ТЕХНІЧНОЇ КОНФЕРЕНЦІЇ

«ІНФОРМАЦІЙНІ МОДЕЛІ, СИСТЕМИ ТА ТЕХНОЛОГІЇ»

18-19 грудня 2024 року

ТЕРНОПІЛЬ
2024

УДК 001
М34

ПРОГРАМНИЙ КОМІТЕТ

Голова: Приймак Микола – професор кафедри комп'ютерних систем та мереж, д.т.н., професо**р.**

Співголови: Марущак Павло – проректор з наукової роботи, докт. техн. наук, професор.

Баран Ігор – канд. техн. наук, доцент, декан факультету ФІС.

Науковий секретар: Надія Крива– старший викладач.

Члени: Василь Кривень - завідувач кафедри математичних методів в інженерії д.ф.-м.н., професор; Галина Осухівська – завідувач кафедри комп'ютерних систем та мереж, к.т.н., доцент; Микола Карпінський - професор кафедри кібербезпеки, д.т.н., професор; Жанна Баб'як - завідувач кафедри української та іноземних мов, к.пед. н., доцент; Ярослав Литвиненко – професор кафедри комп'ютерних наук, д.т.н., професор; Михайло Петрик - завідувач кафедри програмної іженерії, д.ф.-м.н., професор; Наталія Загородна – завідувач кафедри кібербезпеки, к.т.н., доцент.

ОРГАНІЗАЦІЙНИЙ КОМІТЕТ

Голова: Скоренький Юрій Любомирович – канд. техн. наук, доцент кафедри фізики.

Члени: доцент кафедри комп'ютерних наук, к.т.н. В. Никитюк; доцент кафедри програмної інженерії, к.т.н. Д. Михалик; доцент кафедри кібербезпеки, к.т.н. М. Стадник; доцент кафедри комп'ютерних систем та мереж, к.т.н. Є. Тиш; ст. викладач Л. Джиджора.

М34 Матеріали XII науково-технічної конфції «Інформаційні моделі, системи та технології» Тернопільського національного технічного університету імені Івана Пулюя, (Тернопіль, 18–19 грудня 2024 р.). – Тернопіль : Тернопільський національний технічний університет імені Івана Пулюя, 2024.

Адреса оргкомітету: ТНТУ ім. І. Пулюя, м. Тернопіль, вул. Руська, 56, 46001,
тел. (0352) 52-41-33, факс (0352) 25-49-83.
E-mail: conffis2024@gmail.com

Редагування, оформлення та верстка: Надія Крива

СЕКЦІЇ КОНФЕРЕНЦІЇ, ЯКІ ПРЕДСТВЛЕНІ В ЗБІРНИКУ

- Математичне моделювання
- Інформаційні системи та технології, кібербезпека
- Комп'ютерні системи та мережі
- Програмна інженерія та моделювання складних розподілених систем
- Новітні фізико-технічні та освітні технології

В збірнику надруковано тези доповідей XII науково-технічної конференції «Інформаційні моделі, системи та технології» (Тернопіль, 18–19 грудня 2024 р.) за такими науковими напрямками: математичне моделювання; інформаційні системи та технології, кібербезпека; комп'ютерні системи та мережі; програмна інженерія та моделювання складних розподілених систем; новітні фізико-технічні та освітні технології.

Розрахований на науковців, викладачів та студентів вузів.

За зміст тез та дотримання норм академічної доброчесності відповідальність несе автор.

УДК 004.056.53
Наталія Бурмістрова, студент гр. СБм-61, Руслан Козак, к.т.н., доцент
(Тернопільський національний технічний університет імені Івана Пулюя)

ЗАСТОСУВАННЯ МАШИННОГО НАВЧАННЯ ДЛЯ ТЕСТУВАННЯ НА ПРОНИКНЕННЯ: ФРЕЙМВОРК SHENNINA

Nataliia Burmistrova, student of gr. СБм-61, Ruslan Kozak, Ph.D., Assoc. Prof.

USING MACHINE LEARNING FOR PENETRATION TESTING: SHENNINA FRAMEWORK

Автоматизація процесів пентесту із застосуванням машинного навчання відкриває нові можливості для досягнення цілей кібербезпеки. Фреймворк Shennina, що використовує машинне навчання для адаптивного виявлення та експлуатації вразливостей, є одним із новітніх рішень у цій сфері, що потребує дослідження його ефективності та можливостей. Автоматизоване тестування на проникнення не лише сприяє оперативності й ефективності тестування, але й дозволяє значно розширити його масштаби, що особливо важливо для комплексних систем, де необхідна всеосяжна безпекова перевірка [1].

Метою дослідження був аналіз можливостей та обмежень фреймворку Shennina для автоматизованого тестування на проникнення в порівнянні зі сканером вразливостей Nessus. Дослідження спрямоване на оцінку ефективності використання Shennina для реальної експлуатації вразливостей та визначення сценаріїв, у яких цей фреймворк є найбільш доцільним [2].

Для тестування та порівняння ефективності інструментів Shennina та Nessus було обрано віртуальну машину Metasploitable 2, яка є навмисно вразливою версією операційної системи Ubuntu Linux, спеціально створеною для навчальних цілей у сфері кібербезпеки та тестування на проникнення. Ця віртуальна машина дозволяє відпрацьовувати навички виявлення та експлуатації вразливостей у безпечному та контрольованому середовищі.

У процесі дослідження використовували два основних інструменти: Nessus і Shennina, які виконували сканування в контрольованому середовищі [3]. Алгоритм дослідження передбачав чітко структуровані етапи, що забезпечують повноту аналізу та можливість порівняння результатів між обраними інструментами.

З метою оцінки роботи Shennina та Nessus було сформовано метрики, які дозволяють провести об'єктивне порівняння цих інструментів: кількість виявлених вразливостей, час виконання сканування, глибина аналізу, покриття тестової системи, споживання системних ресурсів.

У результаті дослідження встановлено, що фреймворк Shennina, завдяки інтеграції з Metasploit і можливості автоматичної експлуатації, продемонстрував гнучкість у виявленні та використанні не лише стандартних, але й штучно створених вразливостей конфігураційного характеру. Однак охоплення виявлених уразливостей було вужчим, ніж сканера вразливостей Nessus. Це свідчить про необхідність комплексного підходу: використання Nessus для широкого виявлення вразливостей і аналізу відповідності до CVE, а Shennina – для перевірки можливості експлуатації та оцінки практичного ризику для інформаційних систем.

Література

1. Yaacoub J.-P. A., Noura H. N., Salman O., Chehab A. A Survey on Ethical Hacking: Issues and Challenges // A Preprint. American University of Beirut, Electrical and Computer Engineering Department, Beirut, Lebanon. 2020.
2. The Usage of Machine Learning on Penetration Testing Automation // Proceedings of the 2023 3rd International Conference on Electronic and Electrical Engineering and Intelligent System (ICE3IS). August 2023.
3. AI-Powered Penetration Testing using Shennina: From Simulation to Validation // Proceedings of the 2024 ACM Conference. July 2024.